A Defense of Pure Connectionism

Alex B. Kiefer

The Graduate Center, City University of New York

How does access to this work benefit you? Let us know!

Follow this and additional works at: https://academicworks.cuny.edu/gc_etds

Part of the Cognitive Psychology Commons, Computational Neuroscience Commons, Philosophy of Language Commons, and the Philosophy of Mind Commons

Recommended Citation

https://academicworks.cuny.edu/gc_etds/3036

This Dissertation is brought to you by CUNY Academic Works. It has been accepted for inclusion in All Dissertations, Theses, and Capstone Projects by an authorized administrator of CUNY Academic Works. For more information, please contact deposit@gc.cuny.edu.
A DEFENSE OF PURE CONNECTIONISM

by

ALEXANDER B. KIEFER

A dissertation submitted to the Graduate Faculty in Philosophy in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

2019
A Defense of Pure Connectionism

By

Alexander B. Kiefer

This manuscript has been read and accepted by the Graduate Faculty in Philosophy in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

1/22/2019

Eric Mandelbaum

Chair of Examining Committee

1/22/2019

Nickolas Pappas

Executive Officer

Supervisory Committee:

David Rosenthal

Jakob Hohwy

Eric Mandelbaum

THE CITY UNIVERSITY OF NEW YORK
ABSTRACT

A Defense of Pure Connectionism

by

Alexander B. Kiefer

Advisor: David M. Rosenthal

Connectionism is an approach to neural-networks-based cognitive modeling that encompasses the recent deep learning movement in artificial intelligence. It came of age in the 1980s, with its roots in cybernetics and earlier attempts to model the brain as a system of simple parallel processors. Connectionist models center on statistical inference within neural networks with empirically learnable parameters, which can be represented as graphical models. More recent approaches focus on learning and inference within hierarchical generative models. Contra influential and ongoing critiques, I argue in this dissertation that the connectionist approach to cognitive science possesses in principle (and, as is becoming increasingly clear, in practice) the resources to model even the most rich and distinctly human cognitive capacities, such as abstract, conceptual thought and natural language comprehension and production.

Consonant with much previous philosophical work on connectionism, I argue that a core principle—that proximal representations in a vector space have similar semantic values—is the key to a successful connectionist account of the systematicity and productivity of thought, language, and other core cognitive phenomena. My work here differs from preceding work in philosophy in several respects: (1) I compare a wide variety of connectionist responses to the systematicity challenge and isolate two main strands that are both historically important and reflected in ongoing work today: (a) vector symbolic architectures and (b) (compositional) vector space semantic models; (2) I consider very recent applications of these approaches, including
their deployment on large-scale machine learning tasks such as machine translation; (3) I argue, again on the basis mostly of recent developments, for a continuity in representation and processing across natural language, image processing and other domains; (4) I explicitly link broad, abstract features of connectionist representation to recent proposals in cognitive science similar in spirit, such as hierarchical Bayesian and free energy minimization approaches, and offer a single rebuttal of criticisms of these related paradigms; (5) I critique recent alternative proposals that argue for a hybrid Classical (i.e. serial symbolic)/statistical model of mind; (6) I argue that defending the most plausible form of a connectionist cognitive architecture requires rethinking certain distinctions that have figured prominently in the history of the philosophy of mind and language, such as that between word-, phrase-, and sentence-level semantic content, and between inference and association.
Acknowledgements

This dissertation is dedicated to my daughter Evelyn Stelmak Kiefer, who was born just before it was. I’m not sure what she would need it for, but it’s hers. Thanks to my partner Natallia Stelmak Schabner for her patience, love and inspiration during the writing of this work, and to her daughter Sophia. I am deeply grateful to my family (all of you, truly—I won’t name names here in case there’s some cousin I forgot about) for the same, and for believing in my potential since probably before I was born in many cases. The list of friends that I have to thank for accompanying me through life and sustaining my very existence is long, but I wish to thank here those who have specifically encouraged and inspired my work in philosophy over the years: Andrew, Ian, Rose, Rachel, Richard, Hakwan, and Pete. I am also grateful to the chiptune crew and other friends in music and the arts (particularly Jeff, Kira, Haeyoung, Jamie, Chris, Tamara, Josh, Dok, Amy, Pietro, and Jeremy) for being the truest of friends and for informing my work in oblique but essential ways. If I’ve forgotten anyone, get in touch and I’ll put you in the dedication when I write a book.

Thanks to my advisor David Rosenthal and the members of my dissertation committee, Jakob Hohwy, Eric Mandelbaum, Richard Brown, and Gary Ostertag, for helpful feedback on drafts of this work. I’d additionally like to thank my advisor for introducing me to cognitive science, for expert and dedicated guidance over the years, for serving as an exemplar and inspiration, and for his patience in helping me work through many iterations of these ideas (and many others). I would like to thank Jakob Hohwy for taking an early interest in my work, for encouraging me to develop its most promising aspects, and for generously sharing his insight and enthusiasm. Thanks to Richard Brown and Gary Ostertag for their support over the years, and for being involved with this project since the prospectus days. And additional thanks to Eric Mandelbaum for posing the stiffest challenges to my work in the sincere hope of improving it.
I am also grateful to teachers, friends and fellow graduate students, including Pete Mandik, Josh Weisberg, Kenneth Williford, Hakwan Lau, Graham Priest, Janet Fodor, David Pereplyotchik, Jake Quilty-Dunn, Zoe Jenkin, Grace Helton, Jona Vance, Ryan DeChant, Jesse Atencio, Kate Pendoley, Jacob Berger, Henry Shevlin, Kate Storrs, Eric Leonardis, Hudson Cooper, and the cognitive science group at the CUNY Graduate Center, for indispensable exchanges that have informed my thinking. Thanks also to Geoffrey Hinton for kindly replying to my two brash and overly familiar emails. It was more inspiring than you know. And not least to Jerry Fodor, for his love of the philosophy of mind and for goading the connectionists on back when their success seemed less than certain.
A Defense of Pure Connectionism - Contents

Chapter 1: Preliminaries on Connectionism and Systematicity

Introduction.........................................................................................................................1
1.1 Systematicity, productivity and compositionality.......................................................10
1.2 Fodor and Pylyshyn’s challenge to connectionism....................................................14
1.3 Systematicity and vector spaces..............................................................................19

Chapter 2: Vector Symbolic Architectures

2.1 Smolensky architectures............................................................................................26
2.2 Smolensky’s take on Smolensky architectures.........................................................32
2.3 Fodor’s take on Smolensky architectures..................................................................36
2.4 McLaughlin’s take on Smolensky architectures.......................................................42
2.5 Holographic Reduced Representations......................................................................48
2.6 Processing in Vector Symbolic Architectures..........................................................52
2.7 Compositional VS combinatorial systems.................................................................56
2.8 Inferential systematicity.............................................................................................62
2.9 Some problems for VSAs.........................................................................................67

Chapter 3: Neural Vector Space Models

3.1 Distributional semantics............................................................................................76
3.2 Vector space models in semantics............................................................................81
3.3 Neural language models and vector space embeddings............................................85
3.4 GloVe vectors............................................................................................................92
3.5 Collobert et al’s model: syntax from distribution.............................................. 97
3.6 Compositionality in vector space semantic models........................................... 104

Chapter 4: Parsing in Connectionist Networks

4.1 The RAAM system and autoencoders............................................................. 110
4.2 Discovering parse trees................................................................................... 117
4.3 More powerful tree-shaped models................................................................. 122
4.4 Applying computer vision techniques to language.......................................... 129
4.5 Hinton’s critique of CNNs............................................................................... 134
4.6 The capsules approach.................................................................................... 139

Chapter 5: Concepts and Translation

5.1 Concepts and propositions............................................................................... 148
5.2 Words as regions, concepts as operators......................................................... 155
5.3 The prototype theory of concepts revisited..................................................... 162
5.4 Machine translation and the language of thought.......................................... 168
5.5 Multimodal connectionist nets......................................................................... 175
5.6 Larger-scale multimodal models...................................................................... 182

Chapter 6: Reasoning

6.1 Inference, association and vector spaces....................................................... 189
6.2 Induction and semantic entailment................................................................. 197
6.3 Inductive reasoning in connectionist systems............................................... 204
6.4 Deductive reasoning in connectionist systems.............................................. 211
6.5 The PLoT hypothesis...................................................................................... 218
6.6 One-shot and zero-shot learning................................................................. 224
6.7 Biological plausibility............................................................................. 229
6.8 The representational power of graphical models..................................... 233
6.9 Serial VS parallel architectures and implementation of the LoT.............. 240
Conclusion..................................................................................................... 243

References..................................................................................................... 249
# A Defense of Pure Connectionism - List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Two types of distributed representation</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>A single role/filler tensor product binding</td>
<td>29</td>
</tr>
<tr>
<td>3</td>
<td>Illustration of the circular convolution operation</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Milokov et al’s recurrent neural network language model</td>
<td>87</td>
</tr>
<tr>
<td>5</td>
<td>Word vectors learned from the Wikipedia corpus</td>
<td>103</td>
</tr>
<tr>
<td>6</td>
<td>A recursive neural network</td>
<td>111</td>
</tr>
<tr>
<td>7</td>
<td>Two varieties of recursive autoencoder</td>
<td>114</td>
</tr>
<tr>
<td>8</td>
<td>Schematic layout of a capsules network</td>
<td>140</td>
</tr>
<tr>
<td>9</td>
<td>A combined directed/undirected generative model</td>
<td>180</td>
</tr>
<tr>
<td>10</td>
<td>An image accurately captioned by Karpathy and Fei-Fei’s model</td>
<td>184</td>
</tr>
<tr>
<td>11</td>
<td>The skip-thoughts model</td>
<td>207</td>
</tr>
<tr>
<td>12</td>
<td>Orthogonal projection using the NOT operator</td>
<td>216</td>
</tr>
</tbody>
</table>
Chapter 1: Preliminaries on Connectionism and Systematicity

Introduction

Recent cognitive science has seen a revival of interest (in somewhat altered terms) of the debate between proponents of “Classical” and “Connectionist” models\(^1\) of cognitive

\(^1\) Below, I drop the initial capital for “Connectionist” but maintain it for “Classical”, partly in deference to the practice of writers in each camp, and partly because Fodor and colleagues use the term “Classical” in a quasi-technical sense somewhat distinct from more general uses of “classical” (cf. §1.1 below). The term “connectionism” is not currently in vogue even among connectionists, but I find it a useful general-purpose term for a well-known style of artificial neural network modeling, encompassing deep learning as a special case involving large networks with many layers and empirically learned weights.
architecture (Fodor and Pylyshyn, 1988). The recent successes of deep learning\(^2\) (i.e. 21st-century connectionism) in artificial intelligence constitute a strong abductive argument in favor of connectionist models of at least many cognitive and perceptual processes. And the domain-generality of deep learning algorithms, together with their scalability\(^3\), suggests that deep learning may furnish the principles for a general artificial intelligence, and therefore for a general theory of mind, as some proponents of connectionist models have long maintained (Smolensky 1988b). Such a theory would suggest, broadly, that the mind is a statistical model, with the most conceptually promising (if not yet the most empirically successful) recent approaches focusing more narrowly on the unsupervised learning of generative models from data (Hinton and Sejnowski 1983, Hinton and Zemel 1994, Hinton et al 1995, Goodfellow et al 2014). This principle is also at the heart of recent broad theories of cortical function in part inspired by these machine learning models (Friston 2009, Hohwy 2013, Clark 2013).

Against this, critics have urged that pure deep learning (like any connectionist and therefore associationist theory) fails to provide a plausible model of crucial aspects of human psychological function, particularly in the domain of higher cognition. Joshua Tenenbaum and colleagues, for example, argue that human beings excel at learning novel concepts from just a few examples, as measured by successful generalization (Tenenbaum et al 2011, p.1279), whereas deep learning algorithms typically require large amounts of data to extract regularities (Tenenbaum et al 2011, Salakhutdinov et al 2011). This may seem to point to fundamentally

\(^2\) See (Goodfellow et al 2016) for an overview.

\(^3\) The respects in which deep learning models are “scalable” are myriad and themselves merit deeper consideration than it is to my purpose to engage in here. Briefly, I have in mind at least (a) the applicability of the same algorithm(s) to lower- and higher-dimensional input in the same domain (viz. 28 x 28 → 2800 x 2800 pixel images), (b) the iterability of the algorithms e.g. by stacking identical layers in a deep network, and (c) the fact that deep learning models of related tasks can be joined to yield larger differentiable systems that can be trained end-to-end via a single objective function, as in e.g. (Karpathy 2016). (b) is perhaps a special case of (c), since deeper layers in a network compute different transformations of the input, i.e. different representations, and both shade into the issue of domain generality, since some domains are complex and subsume simpler ones (as for example object recognition arguably does edge detection).
different learning mechanisms, and in particular, to the existence of highly abstract, structured
representations, which greatly facilitate inference from data (Tenenbaum et al 2011, Lake et al
2015). Relatedly, connectionist networks are structured as graphical models (in some cases, as
Bayesian networks\textsuperscript{4}), whose expressive power is, it is argued, fundamentally limited as
compared to more flexible symbolic representations (Goodman et al 2015, Williams 2018).

The latter point directly echoes (Fodor and Pylyshyn’s 1988)’s classic critique of
connectionism: connectionist representations lack internal constituent structure and an
attendant combinatorial semantics, which is our best bet for explaining the systematicity (if I can
think “Fred is red” and “George is green” then I can think “Fred is green” and “George is red”)\textsuperscript{5}
and productivity (i.e. the capacity to produce indefinitely many novel tokens) of thought and
language.\textsuperscript{6} Goodman, Tenenbaum and colleagues draw much the same conclusion that Fodor
and Pylyshyn did, but with a twist: we need an internal “Language of Thought” (LoT—see Fodor
1975), they claim, but rather than throwing out the statistical modeling approach to cognition\textsuperscript{7},
we can combine this approach elegantly with the LoT hypothesis, yielding the “Probabilistic
Language of Thought” (PLoT) hypothesis (Kemp and Tenenbaum 2008, Goodman et al 2015).

The essence of the PLoT hypothesis is that conceptual thought involves (a) a basic
inventory of symbols with a compositional syntactic and semantic structure, that (b) figure in
stochastic programs, i.e. programs with a branching structure whose transitions depend on the
values of random variables sampled from some distribution. It is also characteristically

\begin{itemize}
  \item Deep learning, like any approach to the mind based on statistical modeling, always operates in a broadly
Bayesian way since model parameters are estimated from data, usually iteratively via gradient descent
from some initial (possibly arbitrary or random) prior. Bayesian networks in the formal sense are acyclic,
directed graphical models whose nodes represent events or states of affairs and whose edges represent
dependencies between the events associated with connected nodes (Pearl 1988).
  \item See Evans (1982, §4.3), who discusses a type of systematicity under the description “the generality
constraint”.
  \item See also (Fodor 1997), discussed below.
  \item Or, what amounts to the same in practice, relegating its mechanism to the status of mere
“implementation base” for a symbolic system so that it can safely be ignored in theorizing about cognition,
\end{itemize}
assumed that (c) these symbolic structures, or a set of rules for constructing them, are supplied
\textit{a priori}, i.e. innately (see especially Kemp and Tenenbaum 2008, p.7). Inference and learning
consist in the induction of such programs from data (as constrained by the high-level structured
priors), which can be cast in Bayesian terms (see e.g. Kemp and Tenenbaum 2009).

In this dissertation, my goal is to argue that connectionist systems stand a very good
chance of meeting these challenges (and so remaining contenders in the arena of cognitive
architecture) by appeal to a system of representations that exhibits \textit{combinatorial} structure
without thereby exhibiting “Classical” \textit{constituent} structure in the sense defined in (Fodor and
McLaughlin 1995). The result is a view according to which connectionist networks can in
principle serve as a cognitive (representational) architecture, without simply implementing a
Classical (conventional serial symbol processing) system.

As I’ll argue, there are nonetheless reasons to suppose that a successful connectionist
general AI will end up exploiting syntactically structured representations in certain ways.
However, I think such representations will be connectionist representations. Hence, one of the
main claims of Fodor and Pylyshyn (1988), that connectionist representations as such can’t
exhibit constituent structure, must be false, as must the following claim: “What it appears you
can’t do...is have both a combinatorial representational system and a Connectionist architecture
\textit{at the cognitive level}” (Fodor and Pylyshyn 1988, p.28). In any case, it emerges that the way in
which connectionist systems would exploit syntactic structure is very different from what may
have been envisioned from a Classical perspective, and depends more centrally on the kind of
combinatorial (as opposed to constituent) structure just mentioned.

The most well-known early connectionist responses to Fodor and Pylyshyn are the
so-called “Smolensky architectures”\textsuperscript{8} or “tensor product representations” proposed in

\footnote{This is Fodor’s term in (Fodor 1997).}
(Smolensky 1990) and elsewhere, on the one hand, and the “Recursive Auto-Associative Memory” (RAAM) system, on the other (Pollack 1990). These models are the progenitors of two quite different families of connectionist systems: Vector Symbolic Architectures (VSAs) (Gayler 2003, Plate 1995) and recursive neural nets (see, e.g., Socher et al 2011a, b), respectively. Of these two model classes, the former are more closely related to Classical systems, and amount in effect to ways of reversibly encoding conventional structured symbols in connectionist nets, using a handful of fixed mappings between vectors. The latter models, which learn recursive vector composition functions from data, have proven more fruitful in practice and are at the heart of many powerful contemporary connectionist language parsers (see e.g. Socher et al 2013a).

Part of the reason for the broader uptake of the RAAM approach, as against Smolensky’s, is that it employs more variable, empirically learned mappings between representations, hewing closer to the bread and butter of connectionist learning and processing. This affords theoretical connections to other connectionist models such as autoencoders (Hinton and Salakhutdinov 2006), and to a powerful class of models in computational linguistics called Vector Space (semantic) Models (VSMs).

VSMs are themselves implementations of the “distributional” approach to semantics (see e.g. Harris 1954, 1968, Miller and Charles 1991), whose central hypothesis is that the co-occurrence statistics of expressions in natural language corpora serve as an adequate guide to their semantic properties. VSMs can be learned from this distributional data as components of generative models implemented in connectionist networks (“neural language models”—see Bengio et al 2003). The representations in these models can be combined recursively to yield

---

9 I’ll be discussing vectors *ad nauseam* below, but in the interests of making this dissertation maximally accessible: a vector is, for present purposes at least, simply an ordered list of numbers.
the kind of non-Classical combinatorial structure\textsuperscript{10} advertised above. I argue that this approach is central, explicitly or implicitly, to almost all successful large-scale deep learning systems that model higher cognitive functions, such as natural language comprehension and reasoning.

Somewhat orthogonal to my argument are other attempts, such as those in (Horgan and Tienson 1996) and (Stewart and Eliasmith 2012), to argue that connectionist networks may involve non-Classical forms of compositionality and constituent structure. I argue below that, while the features of connectionist models appealed to in these accounts are interesting and well worth drawing attention to, they don’t really amount to non-Classical forms of constituency.

If the processing steps in recursively applied networks like those just discussed were diagrammed, they would look like tree structures isomorphic to conventional syntax trees, where the representation at each node corresponds to a vector. Connectionist proposals also exist in which similar tree structures are not merely virtual, but are constructed out of the hierarchically arranged units in their hidden layers\textsuperscript{11} (Kalchbrenner et al 2014, Hinton et al 2000, Sabour et al 2017). Contrary to Fodor and Pylyshyn’s assumption (1988, §2.1.2), it makes sense in these cases to read a connectionist graph precisely as one would read a syntax tree, assigning progressively more inclusive contents to nodes higher in the tree. But this literal embodiment of tree structure in a network is, I argue, not essential to the connectionist explanation of systematicity on offer here. Models that incorporate only this sort of tree structure without also making use of a combinatorial system of distributed vector representations at each node (such

\textsuperscript{10} As discussed below, connectionist modelers tend to refer to these kinds of models as providing a “compositional semantics” for vector representations. However, Fodor and other hard-line Classicists tend to define compositionality in such a way that it depends essentially on the syntactic constituent structure of representations (see again Fodor and McLaughlin 1995). I have tried to avoid confusion by referring to this non-Classical form of compositionality as “combinatorial structure”, following (Fodor and Pylyshyn 1988, p.13).

\textsuperscript{11} I will assume basic familiarity with connectionist networks in this dissertation, but will do my best to explain any details or technical terms that have not already received wide attention in philosophy.
as that in Kalchbrenner et al 2014) are severely limited in the functions that they can perform on the basis of their representations.

Interestingly, the type of combinatorial representational system under discussion is not limited in its application to natural language processing tasks, or even to purely cognitive tasks. Some of the very same approaches to representation and processing, including those that construct literal parse trees over input nodes, have been used successfully for “scene understanding” (e.g. image segmentation and object classification—see Socher et al 2011b). Perception of syntactic and semantic properties (Azzouni 2013) in natural language processing may be thought, on this model, to be a special case of perceptual processing more generally.

Relatedly, there are perhaps unexpected commonalities between successful connectionist models of purely linguistic and of cross-modal tasks. Google’s state-of-the-art machine translation system (Johnson et al 2017, Wu et al unpublished), for example, functions identically (at an only moderately coarse grain of description) to connectionist models that map images to natural language descriptions (Karpathy and Fei-Fei 2015, Vinyals et al 2015, Karpathy 2016). Aside from some matters of detail, the one type of model can be transformed into the other simply by switching the modality of the input channel (from English, or some other natural language, to vision).¹²

It is thus at least possible and perhaps probable that something like a language of thought, in the sense of an amodal neural code, sits atop the sensory and motor hierarchies at their intersection. If connectionist models are on the right track, however, this “language”—vectorese—is also the language of sensory processing and motor control (though distinctly cognitive function might depend on a certain sophisticated dialect). Whether this

---

¹² I am not suggesting that simply removing the input channel to Google’s translation system and replacing it with a neural network for vision would result in a system that worked well—some retraining would certainly be necessary. My point is that the architecture remains constant across these two prima facie very different tasks.
perspective vindicates the Language of Thought hypothesis is perhaps a verbal question, but one sharpened by the question of the dependence of processing on Classical constituency. And verbal questions aside, the fact that psychological functions related to sensory modalities and to language alike are (on the relevant models) amenable to description in terms of a single domain-general type of processing suggests that these domains are perhaps not as fundamentally different as their phenomenology may suggest.

One respect in which I take many of the most successful connectionist approaches discussed here to be deficient as potential models of mind is in their reliance on supervised learning methods. This is particularly problematic in the case of natural language processing, where the best parsers are essentially provided tree structures a priori as part of their training. This is obviously not a psychologically realistic model of language learning. However, the corpus-based learning that informs VSMs is generally unsupervised, and more sophisticated algorithms for unsupervised learning are a key focus in contemporary connectionist research (see Goodfellow et al 2014 for a landmark paper in this area). There is thus hope that in the future, connectionist models will be developed that combine the plausibility of unsupervised learning with competitive performance, as I discuss below.

An apparent drawback of connectionist models (indeed, the apparent drawback that motivates most criticisms of them) is that they rely on statistically based adjustment of “synaptic” weights and thus on a fundamentally associationist conception of learning and representation. In addition to worries about systematicity, this poses a challenge for understanding logical, rule-based inference in such systems. I argue below that induction cannot in any case be

---

13 The distinction between supervised and unsupervised machine learning is, in my view, not deep (in particular it need not map onto any important difference between architectures or processing styles), but it is psychologically significant. Briefly, a learning method is supervised if it takes labeled data as input and learns a mapping from the data to the labels, while unsupervised methods (intimately related to generative models) learn the distribution of the data itself.
modeled as a matter of the application of purely formal rules, even within PLoT models such as that hinted at in (Goodman et al 2015), and that induction constitutes the bulk of human inference.

We are thus in good shape if we can explain how the rather special case of deductive reasoning can be accommodated in connectionist systems. I discuss several proposals for doing this below. I also argue that despite the merits of the PLoT hypothesis, it does not possess fundamental conceptual resources that put it at an advantage with respect to connectionist systems. One very challenging task, "one-shot classification" (Lake et al 2015), seems to point to a serious advantage for models that employ more structured representations, but it turns out on closer inspection that a connectionist solution exists that is, all things considered, competitive on this task (Koch 2015).

The plan for the dissertation is as follows. In the remainder of this short introductory chapter, I outline Fodor and Pylyshyn’s challenge to connectionism from the systematicity and productivity of thought, as well as their positive proposal for explaining systematicity, i.e. Classical constituency. I also discuss some preliminary moves connectionists might make in reply, which serve as the backbone of the more nuanced proposals in later chapters.

In Chapter 2, I discuss Vector Symbolic Architectures and argue that while they do offer the beginnings of a non-Classical explanation of systematicity, they are limited in ways that make them unsuitable as core models of general intelligence.

In Chapter 3, I discuss Vector Space Models in semantics and the distributional approach to semantics that motivates them, concluding with a preliminary discussion of how they might implement a form of (combinatorial, not Classical) compositionality.

Chapter 4 takes an extended look at connectionist models of natural language processing, which doubles as a survey of powerful neural network-based composition functions
for vector space representations, and explains the way in which traditional tree structures may be constructed online during processing in connectionist networks.

Chapter 5, which is in a way the heart of the dissertation, begins with some philosophical reflections on the surprising relations between concepts and propositional or “sentence-sized” representations that obtain in certain vector space representations, and defends a conception of concepts as operators on vector spaces. This provides a natural segue into a discussion of multimodal connectionist networks that exploit the “encoder-decoder” architecture (Wu et al unpublished) that is in many ways the core of a workable connectionist cognitive architecture.

In Chapter 6, I conclude by discussing various ways in which connectionist networks may implement inference. As mentioned above, induction comes naturally, but deduction is a bit of a problem case (just as induction is for Classical theories). In the latter half of this chapter, I discuss the PLoT hypothesis and argue that despite its merits, it does not amount to a rival to the connectionist paradigm.

1.1 Systematicity, productivity and compositionality

The debate over connectionism, cognitive architecture, and compositionality has been a long and somewhat tortured one. I will not waste space here on preliminaries that have been covered adequately elsewhere, leaving room instead for detailed discussion of aspects of connectionist approaches that have received comparatively little philosophical attention. Here, I touch only on those points necessary to frame the rest of my discussion, without attempting anything like an exhaustive survey even of the recent work on this topic.\(^\text{14}\) I’ll begin with the briefest of reviews of the facts that motivate Fodor and Pylyshyn’s critique of connectionism,

\(^{14}\) See for example (Hinzen et al 2012) and (Calvo and Symons 2014).
and then sketch the main commitments of the Classical compositionality thesis and how they purport to explain those facts.

The natural place to begin this discussion is with the definitions of the *systematicity* and (less importantly, in practice) *productivity* of representational systems. Systematicity is meant to be an empirically observable and *a priori* plausible feature of flexible cognitive systems, while productivity, though not empirically observable (since it hinges in the limit on an infinite range of behavior), is also plausible on certain *a priori* assumptions (including, most importantly, that of the methodological soundness of abstracting competence from performance). *Compositionality* is then proposed as the best explanation of systematicity and productivity.

I’ll let the following long-form quote from (Fodor and Lepore 1993, p.21) define systematicity and productivity, while also introducing compositionality, and fleshing out the dialectic a bit:

No doubt connectionist psychologists (and a handful of philosophers) have occasionally been moved to dispense with compositionality. But that just indicates the desperateness of their plight. For, compositionality is at the heart of some of the most striking properties that natural languages exhibit. The most obvious of these are productivity (roughly, the fact that every natural language can express an open ended set of propositions) and systematicity (roughly, the fact that any natural language that can express the proposition P will also be able to express many propositions that are semantically close to P. If, for example, a language can express the proposition that aRb, then it can express the proposition that bRa; if it can express the proposition that P → Q, then it can express the proposition that Q → P; and so forth.)

What, then, is compositionality? As it is put in Fodor and Lepore (2001, p.365):

“Compositionality says, roughly, that its syntax and its lexical constituents determine the meaning of a complex expression”. Fodor (Fodor 1975) and others have long argued that an analogous sort of compositional structure is very likely a feature of cognition15 as well as of

---

15 Fodor and others tend to talk about thoughts rather than beliefs or other propositional attitudes in this context. I take it that this is because thoughts are in a sense the most unconstrained form of mental representation. You can think, i.e. entertain, pretty much whatever you want. By contrast, you can perhaps only believe something if you find it plausible, you can only wonder whether something if you
language: thoughts, like sentences, have a semantics that depends on the semantics of their parts (i.e. concepts), and the ways in which those parts are combined.

As argued in (Fodor and Lepore 2001), the thesis of compositionality as just stated does not actually, by itself, buy you very much, and in particular it does not yield systematicity. Strictly speaking, the above only entails that there exists some function from the meanings of a set of lexical items and the syntactic properties of a complex expression in which they figure to the meaning of that complex expression. But it does not entail either (a) that those primitive expressions have stable context-independent meanings, or (b) that the meanings of the parts are constituents of the meaning of the complex.

To be concrete: it could be that there is some function from the (context-dependent) meanings of each of the terms in “John swims in the sea”, together with the syntactic structure \([([\text{John}_n]_{\text{NP}}[\text{swims}_v][[\text{in}_\text{prep}][\text{the}_\text{det}\text{sea}_n]_{\text{NP}}]_{\text{PP}}])_{\text{VP}}\), to the meaning of the sentence, without there being any overlap at all between the meaning of the whole and the meanings of the constituents (for example, ‘John’ might refer to John and yet the whole might mean that owls fly high in the sky, for all one knows a priori). And it could a priori be that the meaning of ‘John’ in the context of this syntactic structure differs from its meaning in other contexts, and thus contributes differently to the meanings of the various complexes in which it figures. In sum, it is consistent with the existence of this extremely weak notion of compositionality that the meanings of the complex expressions are all idiomatic (Fodor 1997, p.111), and that their parts have no stable meanings.

Any notion of compositionality of interest, then, will involve further assumptions. Systematicity seems to require at least that the meanings of lexical items be context-independent. As Fodor and Lepore put it, “The idea is this: compositionality says that the meaning of ‘John snores’ and of ‘John swims’ depend, inter alia, on the meaning of ‘John’.

don’t already know it, etc. Focusing on thoughts allows us to abstract from these idiosyncrasies. But compositionality is meant to be in the end a constraint on other attitudes as well.
And it’s because ‘John’ means the same in the context ‘. . .snores’ as it does in the context ‘. . .swims’ that if you know what ‘John’ means in one context you thereby know what it means in the other” (Fodor and Lepore 2001, p. 365; see also Fodor and Pylyshyn p.124).\textsuperscript{16}

Fodor and Lepore (2001, p.364) list several additional features of the semantics of complex expressions that a useful notion of compositionality should comport with, among them: (1) complex meanings are semantically evaluable (for truth or satisfaction), and (2) the meanings of complexes determine facts about co-reference across sentences. These features together furnish an abductive argument for \textit{semantic constituency}: the fact that ‘John swims’ and ‘John snores’ both have meanings involving reference to John is perhaps best explained by ‘John’ s referring to John, and semantic evaluability can be understood in terms of the semantic values of referring expressions being in the extensions of predicates, and the like.

In addition, Fodor and Lepore define “reverse compositionality” as the principle according to which “each constituent expression contributes \textit{the whole of} its meaning to its complex hosts”. The idea here is that if you can represent “dogs bark” you \textit{ipso facto} have the resources to represent “dogs” and “bark” as well (Fodor and Lepore 2001, p.366). The fact that both compositionality and reverse compositionality hold, and as a matter of empirical fact go hand-in-hand, is best explained (they argue) by appeal to semantic constituency.

So, a serviceable notion of compositionality (with respect to explaining systematicity and the other facts adduced above) boils down, not just to the idea that the meanings of semantically complex expressions are functions of the meanings of their parts, but to the claim that the (stable, context-independent) meanings of contributing expressions are constituents of the meanings of the syntactically and semantically complex expressions in which they occur.

\textsuperscript{16} If the distinct senses of an ambiguous expression are a function of the contexts in which they occur (see e.g. Harris 1954), meaning may still be supposed to be context-invariant for each sense.
Note that the preceding definitions of compositionality and reverse compositionality depend on both syntactic and semantic constituency. In later work (see for example Fodor 1997 and Fodor and McLaughlin 1995), Fodor and colleagues argue that the hypothesis really needed to tie all this together nicely is that semantic constituency is to be explained in terms of a necessary co-tokening relation obtaining between semantically complex expressions and their semantic parts. The most obvious candidate relation of this sort is the one suggested by natural language, namely syntactic constituency.

The idea that concepts stand in syntactic constituency relations to thoughts, together with the assumption of a compositional semantics, yields the notion of compositionality that Fodor et al take to be definitive of Classical cognitive architectures (i.e. those that embody the “Language of Thought” (LoT) hypothesis - Fodor 1975): thoughts have a compositional syntax and semantics. Indeed, Fodor and McLaughlin (1995) define Classical constituency in terms of the co-tokening relation: “for a pair of expression types E1, E2, the first is a Classical constituent of the second only if the first is tokened whenever the second is tokened” (Fodor and McLaughlin 1995, p.201).

1.2 Fodor and Pylyshyn’s challenge to connectionism

Next, I very briefly review the challenge to connectionism posed by Fodor and Pylyshyn (1988) on the basis of the preceding considerations. The core of Fodor and Pylyshyn’s critique is the claim that connectionist systems lack structured representations, and that they therefore lack a mechanism for combining representations to form complexes so as to yield a systematic and productive set of representations. From this (alleged) fact about connectionist

---

17 This idea, as it figures in the Language of Thought hypothesis, is what Ken Aizawa refers to as ‘concatenative LOT’ (Aizawa 1997, p.120).
representations, it follows that processing in connectionist systems cannot be sensitive to the structure of the representations over which such processes are defined.

None of this is to say that connectionist systems can't be good models of something or other—in particular, they may be excellent (if rather simplified) models of neural networks. But they can't, according to Fodor and Pylyshyn, model what's psychologically interesting, viz. the relations among internal representations in terms of which cognitive science aims to explain intelligent human behavior. Famously, Fodor and Pylyshyn stress that connectionist networks may very well implement Classical (e.g. serial, symbolic) systems, but the representations of interest would then be higher-level structures whose "horizontal" relations to one another, rather than their "vertical" relations to connectionist systems, would figure in cognitive theories.

To this challenge, the most obvious reply on the part of connectionists is to appeal to distributed representations (Hinton 1981, 1986): representations that consist of the states of collections of nodes in a connectionist network, rather than individual nodes. Since distributed representations have parts, it may be that they have syntactic parts with semantic interpretations, and thus a compositional syntax and semantics.

There are two quite different ways in which this might work. First, the states of the nodes in a given layer or other functionally integrated subregion of a network (i.e. vectors of activities across those nodes) might be taken as distributed representations. Second, by selecting causally related subsets of the nodes in multiple layers of a multilayer connectionist

---

18 I will not bother with a very precise definition of serial symbol processing, as the issues discussed here turn on the specific requirement, already stated, that mental processes be sensitive to the constituent structure of representations. The relevant notion of symbol systems is expounded in (Newell 1980).

19 McLaughlin (2014) insists that, strictly speaking, vectors, which are purely abstract mathematical entities, represent the activities of connectionist units, and should not be confused with them. He thinks, further, that neglecting this distinction has led to confusion. I argue in §2.4 below that it hasn't led to at least some of the specific confusions he envisions. But more generally, the potential for confusion is limited by the fact that almost all connectionist systems exist only as computer simulations, in which the activity of a (virtual) node just is the vector component that "represents" it.
network\textsuperscript{20}, one can build something that looks exactly like a Classical parse tree (see Fig. 1). “Distributed representation” in the connectionist literature almost always refers to the first of these possibilities, but I will later discuss proposals that appeal to the second as well.\textsuperscript{21}

Fodor and Pylyshyn acknowledge these possibilities, but argue that the idea that distributed representations can be leveraged to accommodate syntactic constituency is a confusion. They offer two sets of arguments, one that pertains to the first idea just mentioned (that the components of vector representations are semantically interpretable) and one that pertains to the second (about syntax trees). I’ll discuss each, briefly, in turn.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{Two types of distributed representation. \textbf{Main graphic:} Schematic illustration of a connectionist network with three layers, $L_1$-$L_3$, in which local connections between groups of nodes (bounded in the diagram by boxes) are shown. Shaded circles indicate active units (e.g. activity values $> 0$), versus empty circles for inactive units, and bolded lines indicate large positive connections. \textbf{Inset:} Abstraction of the network state depicted in the main diagram, exhibiting two types of distributed representation: (1) each of the boxes labeled $R_1$, $R_2$, ..., $R_6$ is a representation distributed over a local pool of nodes. (2) the collection of causally connected representations $R_1$-$R_6$ is itself a distributed tree-structured representation $R_T$.}
\end{figure}

\textsuperscript{20} Or causally related groups of nodes across time, in a recurrent setting—see below, esp. Chapter 4.

\textsuperscript{21} Note that in proposals that combine both approaches, the nodes in the tree structure will themselves correspond to distributed representations. See Fig. 1 below, as well as Figs. 6 and 7 in §4.1.
Hinton (1981) outlines the type of proposal that Fodor and Pylyshyn (1988) reject in §2.1.4 of their paper: that connectionist distributed representations might be distributed not just over neurons in the brain but over semantically significant “microfeatures”. Here, Hinton points out that “the ‘direct content’ of a concept (its set of microfeatures) interacts in interesting ways with its ‘associative content’ (its links to other concepts). The reason for this interaction, of course, is that the associative content is caused by the direct content” (Hinton 1981, p.205). Hinton is here advocating something that is quite Classical at its heart: that the causal powers of connectionist representations depend on their internal structure. In this case the internal structure in question is a set of microfeatures, i.e. a feature vector in a vector space.

Fodor and Pylyshyn claim quite flatly that “syntactic/semantic structure is not defined for the sorts of representations that Connectionist models acknowledge” (1988, p.32). But what they really mean to be arguing, I take it, is that no workable notion of syntactic constituency has been proposed. Fodor and Pylyshyn criticize the kind of proposal just mentioned on the grounds that it is just confused: the analysis of concepts into microfeatures is simply not the same thing as syntactic constituency, because semantic containment relations among predicates are not the same thing as structural constituency relations among symbols. Criticizing the idea that the concept CUP’s being in part (semantically) constituted by the feature has-a-handle is at all relevant to compositionality, they say: “The expression ‘has-a-handle’ isn’t part of the expression CUP any more than the English phrase ‘is an unmarried man’ is part of the English phrase ‘is a bachelor’ (Fodor and Pylyshyn 1988, pp.21-22).

I suspect that this criticism rests on a certain failure of imagination brought about by a steadfast commitment to the idea that thought is language-like: while it’s certainly true that the pseudo-English phrases ‘has-a-handle’ and ‘CUP’ bear no syntactic constituency relations to one another, we’re supposed to be talking about concepts as distributed connectionist
representations. The vehicle for the concept CUP is not, according to the connectionist, the symbol ‘CUP’, but a vector representation. And the microfeatural representation whose content is captured in English by ‘has-a-handle’ would, by stipulation, be one component (hence, a literal constituent) of the vector representation corresponding to the concept CUP. Thus, there is a sense in which distributed connectionist representations might support a very conventional form of constituency, indeed Classical constituency (since it involves co-tokening)—a proposal that Fodor and Pylyshyn took themselves to have addressed, but really did not.

Turning to the second proposal: §2.1.2 of (Fodor and Pylyshyn 1988) is devoted entirely to critical discussion of the idea that connectionist graphs might be read (in some cases at least) as hierarchical structures akin to syntax trees. In essence, their argument is that the edges in connectionist graphs signify causal connections between nodes, and therefore not grammatical or constituency relations: “the links in Connectionist diagrams are not generalized pointers that can be made to take on different functional significance by an independent interpreter, but are confined to meaning something like “sends activation to” (pp.18-19).

I find this argument puzzling. Why couldn’t the links in connectionist graphs signify both causal connections between nodes (weighted “synaptic” lines, in fact) and grammatical or constituency relations? Surely this idea shouldn’t be precluded as a way of taking complex structures to be represented in connectionist nets simply on the grounds that the constituent representations would be causally related to one another. Suppose, connectionism aside, that I hear the English phrase ‘running horse’ and token the complex concept RUNNING HORSE as a result, and that as a matter of fact, the tokening of RUNNING has a hand in bringing it about that I token HORSE and thus RUNNING HORSE. It’s hard to see why this etiology should compromise the status of the complex concept as a genuine structured representation.
Perhaps what Fodor and Pylyshyn have in mind is that, since connectionists explicitly build into their models that the edges in graphical depictions of networks should be read as causal connections, the (“independent”) interpretation of them as signifying grammatical relations, etc, would be external to the theory proper. But this seems to tie the connectionist’s hands: surely connectionists should be allowed to define a notion of constituency if their machinery gives them the resources to do so.

The latter point is of course the pivotal one. It’s true that the fact that a connectionist graph is interpretable as a syntax tree doesn’t suffice to make it one. However, if it can be shown that such graphical structures function as syntax trees (while at the same time functioning as components of connectionist networks, i.e. according to the “causal connection” reading of the graph, in such a way that the two functions are systematically related), Fodor and Pylyshyn’s claim will be proven false. Much later on, I’ll look at connectionist models in which parse-tree-like structures are constructed during processing. For now, I move on to canvass some preliminary reasons to suppose that, whether or not they can exhibit Classical constituency, connectionist distributed representations may possess the resources to ground a type of systematicity (and productivity). Later sections spell this idea out in much more detail.

1.3 Systematicity and vector spaces

There is a type of systematicity that connectionist networks may, by their very nature, be expected to exhibit, and it is perhaps best explained by appeal to the idea of a vector space. As is well known, one can think of an activation pattern over the nodes in a given network or layer as a vector, where the value of each node specifies the magnitude of the vector along a corresponding dimension in the vector space.
Clearly, this affords at least a generic sort of systematicity: let’s just assume for the moment that each vector represents something or other. Then, the hardware of the network guarantees that if you can represent whatever is represented by one of the vectors, you can represent many other things as well (the things represented by the other possible vectors), provided that the network is configured so as to make many points in its vector space accessible. The portions of the “activity space” that a given network is actually able (or, in a stochastic network, likely) to visit depends on the network’s weights. So, if you consider the weight values to be part of the architecture, and you want to take advantage of this sort of systematicity, you’d need to design the weights appropriately.

In general, of course, especially in larger modern networks, the weights are not hand-coded but are learned from experience, in precisely the associationist way Fodor and Pylyshyn (1988) lament. But even in a network whose weights are learned (let us grant at least for the moment that this learning is purely associationist), generalization to unseen examples can occur, and indeed, is the main goal of training and the point of the whole enterprise. The weights are learned so that novel inputs drive the network into an appropriate region of its state space (in a typical feedforward network, the relevant state space is that of its “output nodes”).

Here’s Geoff Hinton on this theme, from a paper that is in part a direct reply to Fodor and Pylyshyn. He draws an analogy between layers in a connectionist network and computer chips designed to carry out mathematical functions (Hinton 1990, p.50):

To characterize a multiplier chip as simply associating the input with the correct output is misleading. A chip that multiplies two N-bit numbers to produce a 2N-bit number is able to use far less than the O(2N) components required by a table look-up scheme because it captures a lot of regularities of the task in the hardware. Of course, we can always render it useless by using bigger numbers, but this does not mean that it has failed to capture the structure of multiplication. A computer designer would be ill-advised to leave out hardware multipliers just because they cannot cope with all numbers. Similarly, a theoretical psychologist would be ill-advised to leave out parallel modules that perform fast intuitive inference just because such modules are not the whole story.
Hinton is here suggesting that connectionist networks can exhibit the same kind of systematicity that a hard-coded multiplying device in a conventional digital computer exhibits: within a range fixed by the capabilities of the hardware, any product can be computed. It’s not necessary for the machine to have “seen” a particular number pair before—as soon as the thing is built, it’s capable of multiplying all the numbers within its range. The same is true of a typical connectionist network, with respect to computing functions more generally, once its weights have been learned appropriately from data.

Hinton’s apparently concessive remark at the end about fast parallel modules not being “the whole story” deserves brief commentary. In the same paper, Hinton distinguishes between two types of inference that might be implemented in a connectionist net. On the one hand there is “intuitive” inference, which involves the system’s “settling” into a stable state subsequent to input (as in, for example, the energy minimization processes in Hopfield nets (Hopfield 1982) and related models). On the other hand, there are inferences, which Hinton calls “rational” inferences (p.50), that are distinguished from the former in that they involve changing the mapping from represented entities to states of the network after each step. Rational inferences in this sense are necessarily serial processes, but intuitive inferences may be so as well (even in the course of a single “settling” upon a conclusion given a set of premises as input, several intermediary “conclusions” may be reached). Hinton supposes that conscious, deliberate reasoning may be implemented by a series of intuitive inferences.

It is perhaps worth noting, before moving on, that Hinton’s remarks here needn’t really be seen as concessive with respect to the present dispute: if Fodor and Pylyshyn can appeal, as they do, to the idea that the class of possible Classical systems includes variants that operate in parallel (Fodor and Pylyshyn 1988, pp.55-56), surely a connectionist can appeal to
the idea that the same parallel machinery can be used in several different ways in sequence.\textsuperscript{22} And rational inferences in Hinton’s sense do not \textit{ipso facto} have anything to do with constituent structure.

It has been argued, for example in (Horgan and Tienson 1996) and (Horgan 2012), that the proximity of states to one another in the state space of a network amounts to a kind of “non-Classical compositionality”. The etymology of “compositional”, it is argued, reveals that the essence of composition is being proximally situated. In a well-trained (or, more generally, well-designed) connectionist net, states that are semantically related are also “syntactically” related in that they are adjacent to one another in the network’s activation space. This type of “constituency” does exhibit one of the key features of Classical constituency: it allows purely “syntactic” processing to accomplish semantically defined tasks (e.g. reasoning), at least in principle, because the syntax tracks the semantics. It is the syntactic structure of the \textit{whole network} that gives rise to systematicity on this view, not that of discrete symbols.

This principle—that representations proximal in a vector space are similar in content—is the centerpiece of the successful approaches to modeling natural language processing (and other cognitively sophisticated functions) that I’ll discuss in far greater detail below, and underlies many types of connectionist processing more generally.\textsuperscript{23} However, I think the description of this principle in terms of a non-Classical form of syntax is apt to be misleading.

\begin{footnotesize}
\begin{enumerate}
\item Important, the distinction between intuitive and rational inference applies to conventional computers just as well as it does to parallel computers: “Intuitive inferences correspond, roughly, to single machine instructions and rational inferences correspond to sequences of machine instructions that typically involve changes in the way in which parts of the task are mapped into the CPU.” (Hinton 1981, p.50). It’s just that in a connectionist net (or other parallel system), very much more can be accomplished in a single “intuitive” step, before re-mapping is required. I will have a bit more to say about the distinction between serial and parallel systems much later, in §6.9.
\item It is directly related to the idea of a “content-addressable memory”—see (Hopfield 1982)—as well as to the “semantic hashing” approach to memory retrieval, which is capable of accessing semantically similar documents in a collection, given a cue, in a time that is independent of the size of the search space (Salakhutdinov and Hinton 2009b). It is worth noting that the converse (that similar items have similar representations) does not necessarily hold in the latter system (Salakhutdinov and Hinton 2009b, p.977).
\end{enumerate}
\end{footnotesize}
In the first place, the notion of non-Classical constituency at issue is, given the right set of distributed representations, equivalent to Classical constituency. Assume for the moment that we have a set of vectors in a 3-dimensional space, semantically interpreted in such a way that compositionality and reverse compositionality are satisfied (for example, the interpretation of \([1, 0, 2]\) depends systematically on the interpretation of each of the component numbers, the semantic contribution of the leftmost “1” to \([1, 0, 1]\) and to \([1, 1, 2]\) is the same, the interpretation of a “1” is more or less constant across positions, etc). In this case, the value of each vector component functions as a Classical constituent, and also corresponds to the bounded 2D region of the space delineated by holding that component fixed while varying the others. Thus each Classical constituent would correspond to a non-Classical syntactic feature of the space.

Second, we so far lack an account of precisely how non-Classical syntactic properties figure systematically in processing. Suppose that tokening a thought with content “John loves to swim” disposes me to transition into a state with content “Someone loves swimming”. The fact that the two states would thus be proximal in activation space is the “dispositionally realized” “syntactic” fact to which Horgan and Tienson advert. But here, we must assume the mapping between positions in vector space and positions in “semantic space” in order to explain the inference. It is not yet clear why (unless we assume that the vector components are Classically related to the vectors in which they occur, as discussed above) this mapping should obtain.

By contrast, Classical compositionality does offer an explanation of the Classical analogue of this mapping, for the case of complex representations, at least: given that we’ve got a symbol that refers to John, and one that refers to Mary, and one that refers to the \(x\) loves \(y\) relation, and an inventory of complex symbols that can be formed using these elements, together with a fixed mapping from the semantics of simples to the semantics of complexes, we
can explain (in terms of part-whole constituent structure) how it is that syntax manages to track semantics. What parallel (no pun intended) story do connectionists have to tell?

An adequate connectionist explanation will, it seems to me, be grounded in the fact that proximity in state space corresponds to intrinsic physical similarity between states of the network (see O’Brien and Opie 2004)\(^{24}\), and that similar physical vehicles may (with important caveats)\(^{25}\) be expected to have similar causal effects. Together with the principle that nearby vectors have similar contents, this suggests a way in which the structure of connectionist representations may systematically influence processing even if that structure is not syntactic. The causal efficacy of Classical constituents in processing, appealed to by (Fodor and Pylyshyn 1988), (Fodor and McLaughlin 1995), and elsewhere, is perhaps a special case of this.

Later on, I will return to connectionist proposals that explicitly exploit the principles just articulated. First, in the next chapter, I begin with what may seem a long digression: I consider a set of connectionist networks inspired by the “tensor product” system introduced by Paul Smolensky (Smolensky 1990) that make representational assumptions somewhat at odds with those just discussed. What unifies these approaches (sometimes called “Vector Symbolic Architectures” (VSAs), as noted in the introduction—see Gayler 2003)\(^{26}\) is that they attempt to at least approximately implement structured symbols in connectionist systems.

---

\(^{24}\) That is, assuming a fixed, monotonic mapping between the magnitude of a vector component and some magnitude in the physical medium (see Maley 2011), changing a vector component amounts to both a move in activation space and a proportional physical alteration.

\(^{25}\) There are crucial limitations to this principle. For one, in most useful connectionist nets, vector representations have non-linear effects on those in other layers. Thus, a small difference between two vectors may sometimes correspond to a large difference in downstream effect. But even in the case of discontinuity, the principle may hold: e.g. one group of similar input vectors may yield outputs below the activation threshold of an output unit, while another group of similar inputs exceeds it. A second possibility is that some of the outgoing weights from a layer are much larger than the others, so that units connected to the larger weights dominate the downstream effects of the layer and varying the other units has negligible impact. But in this case, the representation would effectively be distributed only over the units connected to the larger weights.

\(^{26}\) In a slight departure from (Gayler 2003)’s usage, and in order to cut down on acronyms, I use the term “VSA” to cover both Smolensky’s system and its conceptual descendants, rather than only the latter.
VSAs are of interest for several reasons. First, they provide answers to the systematicity challenge that are different in at least some matters of detail from other sorts of answers connectionists might offer. Second, a consideration of how it is that these systems manage to emulate certain features of Classical systems without precisely being Classical systems reveals principles that are also at work in less Classically-inspired systems, and that, in tandem with the principles just discussed, do stand a good chance at accounting for systematicity. As I hope to show, the large (and growing) literature on “compositionality” in vector-based systems of representation picks out a natural kind of sorts distinct from Classical compositional systems—what I have called a combinatory system of representations.
Chapter 2: Vector Symbolic Architectures

2.1 Smolensky architectures

As is well known, Paul Smolensky, soon after the publication of Fodor and Pylyshyn’s critique of connectionism, introduced a connectionist module, the “tensor product representation”, that provides a way of systematically encoding syntactic relations in patterns of activity on units in a connectionist network. This system was designed specifically to answer Fodor and Pylyshyn’s challenge by showing how Classical behavior could be approximated, to arbitrary precision, within a connectionist system. In more recent work, Smolensky and Legendre (2006) develop this idea in greater depth.

In this section I review Smolensky’s proposal. Before proceeding to the technical details, I’ll discuss the conceptual background and address a pervasive ambiguity in discussions of this proposal, including Smolensky’s own. While one might assume that Smolensky’s goal would be to show that systems of connectionist representations (i.e. nodes or groups of them) can be constructed in such a way that they exhibit systematicity, Smolensky frames his project rather as one of showing how “the representation in connectionist systems of symbolic structures” (which themselves exhibit systematicity) is possible (Smolensky 1990, p.165, my italics).

This suggests that Smolensky’s proposal is best thought of primarily as a tool for the systematic encoding and subsequent extraction of arbitrary structures in patterns of activity on connectionist units, without immediate regard for the idea that these structures are specifically symbolic, i.e. have semantic properties. Some of Smolensky’s examples of the use of tensor product representations concern symbols (for example, he talks about cases in which tree
structures and strings can be represented within the system), while other examples involve the representation directly of extra-symbolic entities, such as energy and time.

One way of thinking about the tensor product system, at least in the use to which it has most commonly been put, is as enabling the representation of a set of symbolic structures \( S \), and thereby potentially the representation of whatever those symbol structures can represent, just as a physical symbol system in Alan Newell's sense (see Newell 1980) can, by representing a universal Turing machine, itself gain the representational powers of a universal computer. That is: a connectionist network might be made to simulate a Classical symbol system. Indeed, Fodor's most trenchant criticism of this model is that, insofar as it works, it is entirely parasitic on Classical explanations. More recently, (McLaughlin 2014) aims the same sort of criticism at Smolensky's more recent, and more comprehensive, efforts in the same vein.

However, Smolensky's system, despite allowing for the systematic encoding and decoding of syntactic constituents, is not simply an implementation of a Classical system in the sense we've been discussing. This is possible because a set of symbols \( S \) may be represented in more and less fine-grained ways. In particular, a Smolensky architecture may represent \( S \) only in terms of relevant input/output relations, i.e. functional relations among its members, without representing the internal constituent structure of each member in virtue of which it stands in those relations (relatedly, a symbol system SS may simulate a Turing machine TM by incorporating symbol structures isomorphic to the internal processes of TM—that is, SS may emulate TM—or it may simulate TM by reproducing its input/output profile via some other mechanism (Newell 1980). Moreover, as I've noted, Smolensky's system can be used to represent things other than specifically symbolic structures.

The tensor product representation works as follows. First, suppose we have a set of \( n \) “filler” nodes and \( m \) “role” nodes, as well as a set of \( n \times m \) “binding” nodes (Smolensky 1990,
The activities of the filler nodes are representations corresponding to constituents of whatever complex structure we aim to encode—they may represent words in a sentence, amino acids in a chain, or whatever. The activities of the role nodes are representations of distinct roles that the fillers may play in some system—for instance, *subject, object, left position, right position, first, second, third*, etc. The activities of the binding nodes are representations that correspond to semantic complexes composed by combining fillers in various roles. The representational system thus includes provisions for representing (a) arbitrary entities, (b) the positions that they may occupy in some broader structure, and (c) those structures themselves, i.e. combinations of filler-role pairs. In this section and in this chapter more generally, I focus only on how primitive role and filler representations can be used to construct representations corresponding to more complex entities (such as structured sentences or propositions), not on the question of how the primitive representations get their content (for a discussion of how “primitive” vector representations may be assigned content in a principled way, see Chapter 3).

In Smolensky’s system, complex representations are derived from simples as follows. First, a filler is “bound” to a role via the tensor product operation, a generalization of the outer product operation on vectors. For present purposes, we can think of the tensor product $f \otimes r$ of filler vector $f$ and role vector $r$, where $f$ is $n$-dimensional and $r$ is $m$-dimensional, as an $(n \times m)$-dimensional vector $b$, whose indices correspond to those of an $n \times m$ matrix $B$. The elements $b_{ij}$ of $B$ are the products of the elements $i$ of $f$ and $j$ of $r$, where $i$ and $j$ index rows and columns respectively in the case of the matrix. So, for example, if $f = [1, 0.5, 9]$ and $r = [0, -2]$, $b$

---

27 The relationship between tensors, on the one hand, and matrices, vectors, and scalars, on the other, is somewhat complex. For the purposes of this chapter, it suffices to think of tensors of rank $n$ as the $n$-dimensional volumes of numbers that could be used to represent them (scalars, tensors of rank 0, are single numbers akin to points; vectors, tensors of rank 1, are lists of numbers akin to lines; 2D matrices, tensors of rank 2, are grids of numbers akin to planes; 3D matrices, tensors of rank 3, are cubes of numbers, and so on to higher dimensions). This isn’t quite right because, for one thing, the same rank-2 tensor can be represented by distinct matrices. But it is more than enough at present. Here, I follow the convention of using lower-case bold letters for vectors and upper-case bold letters for matrices.
\[ f \otimes r = [0, -2, 0, -1, 0, -18], \text{ with corresponding matrix (for perspicuity) } B = [[0,-2],[0,-1],[0,-18]]: \]

\( b \) is just the list of rows of \( B \) with inner parentheses removed. This operation is illustrated schematically in Fig. 2 below.

![Figure 2](After Smolensky 1990, Figs. 3 and 9).
A single role/filler tensor product binding.
Grey level indicates activation level.

The result of the binding operation is a role-specific representation of the filler, implemented in the activities \( b \) of the “binding” nodes. Since structures of any interest involve multiple roles, an entire structure can be represented by \textit{superposing} the bound filler-role vectors using simple element-wise addition, to yield a new vector. Thus, the full representation of a structure, \( S \), involving three roles, \( r_1, r_2 \) and \( r_3 \), bound to three fillers \( f_1, f_2 \) and \( f_3 \), is \( S = \sum_{i}^{3} [f_i \otimes r_i] \). Here, the role vectors must be distinct to indicate the different role types that the fillers play, but more than one filler vector may be identical, since the same entity may play multiple
roles. Note that there is no guarantee, using this representation, that vectors for constituents appear in the vector for a complex symbol.

What about cases in which a role is repeated? Consider, for example, an important case: the encoding of binary branching tree structures. The roles left element and right element are exhaustive at each branch, and it is possible to construct an entire tree by recursive use of these two categories (for example, using a bracket notation for trees: $T = [a [b c]]$ is a well-defined tree, in which $a$ is the left child of the root node, $[b c]$, itself a binary tree, is the right child of the root node, and $b$ and $c$ are the left and right children of $[b c]$). This can be captured in the tensor product notation by recursive application of the binding and superposition operations: $T = [a \otimes l] + [(b \otimes l) + (c \otimes r)] \otimes r$, where $l$ and $r$ are the left and right role vectors, respectively.

As Smolensky points out (as do some of his critics—see McLaughlin 2014), this expression is in fact ill-formed, because the dimensionality of the vectors increases with each tensor product operation, and vectors can only be added if they have the same number of dimensions. This raises a whole host of difficulties for the tensor product proposal, none of which are difficult to overcome technically, but the modifications required to overcome them diminish somewhat the neural plausibility of the scheme (see Stewart and Eliasmith 2012), as well as, potentially, its claim to be different from a Classical symbol-processor. These issues are discussed at length in (McLaughlin 2014, especially §7 and §10), and will be touched on in §2.4 below, but I will soon switch focus in any case to a closely related proposal, that of (Plate 1995), that skirts this issue entirely.

Importantly, the connectionist representations employed in this system are typically distributed, as opposed to “localist”, representations. I say “typically” because strictly speaking, local encodings of a sort could be used with the tensor product system: entities could be
represented by the binary states of individual nodes in the “filler” bank. For now, I will sidestep this issue via the following brief argument. Either two such putatively localist representations can be fed into the tensor product system at a time, or not. If so, then the overall vector is in fact a distributed representation of a more complex object that subsumes the two locally represented ones (the representation is “distributed over microfeatures”). If not, then the vector is still a distributed representation because objects can be represented only by those vectors in which one node is active at a time (this is a “one-hot” or “one-of-N” encoding), and so the representation depends on all the vector components.

I will discuss one further feature of the system before turning to some commentary: importantly, once representations have been bound to roles and superposed, they can be extracted again by “cueing”, i.e. operating on the derived representation of the total structure using dedicated “unbinding vectors”, which can be computed automatically using connectionist networks (Smolensky 1990, §3.4.2). If one is willing to put up with a bit of distortion in the recovered vectors, which can be removed using a distinct “cleanup memory” module28, the role vectors themselves can be used to do the recovery. Certain mathematical properties of the set of role vectors must hold to enable perfect recovery (or varying degrees of imperfect recovery) of embedded fillers: if the role vectors are linearly independent, the unbinding vectors can be used to recover fillers perfectly, and if they are orthogonal, they themselves can be used to recover the filler vectors with minimal distortion. It goes without saying, I trust, that all of the operations in this system, not just the computation of the unbinding vectors, can be implemented straightforwardly in connectionist networks.

28 See (Plate 1995) and (Stewart et al 2011).
I turn now to philosophical commentary on the tensor product system, first from Smolensky himself, and then from Fodor and critics in subsequent sections. According to Smolensky, the tensor product system provides a way of understanding how connectionist systems could, in addition to doing all of the things they could already do (back in 1990), represent linguistic structures. Smolensky also suggests that the processing of such explicitly represented linguistic structures could lead to connectionist models of “conscious, serial, rule-guided behavior. This behavior can be modeled as explicit (connectionist) retrieval and interpretation of linguistically structured rules” (Smolensky 1990, p.166).

The “connectionist” in parentheses in the last quoted sentence is important. A constant theme in Smolensky’s reflections on the technical apparatus he presents is that, despite the fact that the tensor product system is designed specifically to encode and manipulate things like tree structures, the ways in which connectionist systems deploy these structured representations will differ from the ways in which structured symbols would be deployed in a Classical architecture.

Here is one of many quotes from Smolensky on the matter (Smolensky 1990, p.167):

Of course, exploiting connectionist representations of the sort of symbolic structures used in symbolic AI by no means commits one to a full connectionist implementation of symbolic AI, which...would miss most of the point of the connectionist approach. The semantic processing of a connectionist representation of a parse tree should not be performed by a connectionist implementation of serially applied symbolic rules that manipulate the tree; rather, the processing should be of the usual connectionist sort: massively parallel satisfaction of multiple soft constraints involving the micro-elements forming the distributed representation of the parse tree.

In the connectionist system Smolensky envisages, talk of symbol structures, while providing a “useful description” of how the system operates at a high level of abstraction, is only

---

For more on “soft constraints”, see (Smolensky 1988, p.18).
approximate, in that “there is no complete, precise formal account of the construction of composites or of mental processes in general that can be stated solely in terms of context-independent semantically interpretable constituents” (Smolensky 1995a, p.184). Elsewhere, Smolensky claims that connectionist processing is structure-sensitive but that computation in VSAs does not operate on the constituents of the encoded symbolic representations (Smolensky 1995b, p.242).

As McLaughlin argues, it is not clear that this combination of positions can be made to fly: if (distributed) connectionist representations are to earn their keep as symbols, they must play functional roles in the systems in which they operate that licenses describing them as such (McLaughlin 2014, p.63). Granted, the basic operations upon representations of any type in a connectionist network, whether they represent structured symbols or not, are at the “micro” level of node activations and weighted summations—in short, linear algebra. These sorts of processing algorithms do look very different from those that one might see written in a high-level programming language for a serial system. Still, these low-level algorithms must implement processing algorithms defined over symbols if those symbols are to function as such. What is the point of encoding structured, language-like symbols in patterns of neural activity if those patterns are not to be processed, if only at a high level of abstraction, as structured symbols?

Smolensky makes much of the fact that the implementation of symbols in his VSA is only approximate, while implementation in the sense relevant to computer science requires exactness. When, for example, a high-level programming language is interpreted and converted to a set of machine-language instructions, there is a determinate mapping from instructions in the high-level program to machine instructions, as is of course necessary in order for the high-level language to allow the user to program the computer. It can’t be that a sequence of symbolically described steps is sometimes mapped to one functionally distinct set
of machine instructions, and sometimes to another (leaving aside any dependence on “indexical” elements like the system clock).

Yet, this is exactly what happens in a VSA, at least under certain conditions: as the capacity of the vector space used to store representational elements is saturated, it “gracefully” fails, so that stored constituents can’t be extracted with perfect accuracy. How inaccurate the decoding is will depend on how many other elements are stored in the same set of nodes (i.e. the same vector space), as well as on the properties of the vectors (as discussed above).

Because the constituent vectors are stored in a potentially lossy way, and are sometimes only approximately recoverable from the whole representation, Smolensky (as well as Stewart and Eliasmith 2012) claim that these systems do not operate Classically. In (Eliasmith 2005)'s terms, the constituents are “blurred” upon being encoded into the overall vector.

Why isn’t this just a performance-limited implementation of a Classical system? (Stewart and Eliasmith 2012) argue that because noise and limitations are “built in” to the way such systems operate, i.e. because theoretical limits on recoverability exist independently of considering any specific psychological implementation of the VSA scheme, it’s an essential feature of the system itself, not a performance limitation. But I think this misconstrues the level of description at which the performance/competence distinction is relevant to the question at hand. Granted, there may be additional performance limitations that turn up when we attempt to implement a VSA in neurons (though Eliasmith himself shows that doing so shouldn’t raise any particular problems—see Eliasmith 2005). But what’s relevant at present is not the relation between processing in VSAs and real, physical cognitive systems, but that between Classical symbol processing and such cognitive systems.

The point can perhaps be best made by looking at Smolensky’s remarks on the subject. As he claims, using larger vectors allows for a closer approximation to a perfect implementation
of a Classical Language of Thought. In the infinite-dimensional limit, the implementation would
be perfect. This suggests that, even short of that limit, the VSA can be viewed as an
approximate implementation of a LoT, insofar as such a system would be functionally equivalent
to one. The fact that it is not a symbol processor whose representations literally exhibit
syntactic compositional structure explains the observed performance constraints. That is: in
posing the VSA, we are already letting our research be constrained by considerations of
biological plausibility. But clearly (and as the creators of such systems explicitly claim), the goal
is to enable connectionist nets to work with representations having constituent structure.30

Here’s a different perspective on this argument, inspired by Eliasmith’s “blurring”
comment. Suppose that there is a Classical language of thought, involving literal syntactic
constituents that are tokened whenever a containing expression is tokened, but subject to the
following limitation: the “words” of a Mentalese “sentence” must all be printed within a finite
(and, let’s suppose, rather small) bounding box, using a fixed-size font. At some point, the
printed words would begin to overlap one another, and in the limit you’d end up with an
indecipherable mess of ink. This sounds exactly like the kind of resource limitation that would
enforce a competence/performance distinction. “We could think indefinitely, indeed infinitely,
complex thoughts—if only the Mentalese terms didn’t have to all be written inside that box!”

Well, that’s more or less what’s going on in VSAs. The vectors that encode constituents,
both relatively simple and complex, all lie in the same vector space. If you try to encode too
many constituents, they begin to “overlap” (given imprecise recall). It’s possible to “clean up”
the recovered vectors by storing a list of the ones that have been encoded, and comparing the
noisy extractions to this list, at least up to certain capacity limits, just as it’s possible to make out
which words are written in a crowded box provided one knows the language, and therefore what

---

30 This is true whether or not the structural features of these representations are taken to be causally
implicated in processing.
the words look like in their uncorrupted form. And the trick used to increase the reliability of
recovery—making the stored vectors orthogonal to (or at least linearly independent of) one
another, amounts to “spacing the words out” in the box so that there’s minimal overlap.

For the preceding reasons, I conclude that at least this argument doesn’t save VSAs
from being mere implementations of a Classical LoT. Yet, I don’t think they do implement one,
for the fundamental reason that (in the general case) the semantically complex representations
simply don’t contain constituent representations whose semantic values contribute to that of the
whole. If a Classical system is defined in terms of Classical constituency, then a VSA is not an
implementation of one even if it is functionally equivalent. The gist of Fodor’s criticism of
Smolensky’s proposal, however (to which I now turn) is that if a VSA doesn’t implement a LoT
there is little reason to expect it to be functionally equivalent to one.

2.3 Fodor’s take on Smolensky architectures

Advocates of Classicism have remained unconvinced by Smolensky’s proposal,
generally maintaining that the tensor product system is either just a thinly veiled implementation
of a Classical system, as just discussed (see also McLaughlin 2014), or fails actually to deliver
an explanation of the facts about systematicity and the like (Fodor and McLaughlin 1995, Fodor
1997). For several reasons, I agree with the critics that the tensor product system is probably
not a sufficient basis on which to build a connectionist alternative to a Classical cognitive
architecture. Nonetheless, a consideration of its merits and defects helps sharpen the terms of
the debate, as I hope to show over the next several sections.

The gist of Fodor’s reply to Smolensky (to which I now turn) is that if a VSA doesn’t
implement a LoT there is little reason to expect it to be functionally equivalent to one. According
to Fodor, deriving a system of vector representations guaranteed to correspond 1-to-1 to a set of symbolic representations doesn’t amount to an explanation of systematicity, insofar as the processing in the system is not Classical. More specifically (Fodor 1997): (A) the tensor product algorithm for encoding syntax trees couldn’t possess psychological reality unless the mind already supported trees such that they could be encoded, in which case these trees, and not connectionist algorithms, would serve to explain systematicity; (B) since (encodings of) tree-constituents in Smolensky’s architecture needn’t be tokened when (encodings of) entire trees are, Smolensky can’t appeal to the necessary co-tokening relation to explain systematicity, or the sensitivity of processing to constituent structure, in the way that Classical architectures can. Moreover, relatedly, the sense in which such structures could figure in processing without being causally efficacious is obscure.

I suspect that (B) evinces a bit of a misunderstanding on Fodor’s part about what Smolensky is up to, though an understandable one, given Smolensky’s talk of representing and encoding structured symbols in connectionist nets, rather than incorporating structured symbols into such networks (see §2.1 above). But I think Smolensky can be interpreted as (i) allowing that some cognitive processes, such as the representation of language and some forms of action planning, likely do involve tree structures, and (ii) being concerned to find a way in which such tree structures can be implemented (though only approximately) in connectionist systems, such that they can help explain more sophisticated cognitive functions, while interfacing naturally with other processes (such as, perhaps, pattern recognition in low-level sensory systems) that connectionism already models well. Granted, the talk of “acausal explanations” disparaged in (Fodor 1997), and Smolensky’s insistence on the lack of a precise mapping between symbol-processing and node-level connectionist algorithms, tend to make this interpretation less available. In any case my primary goal in this chapter is not to defend the
consistency of Smolensky’s thinking, but to show that the technical apparatus he provides can plausibly serve as the basis for a cognitive architecture that naturally exhibits systematicity.

Fodor’s point (B) goes to the heart of what is really interesting about the tensor product proposal. Fodor frames this issue nicely by noting that there is a level of abstraction at which Classical architectures and Smolensky architectures are committed to the same theses about concepts (I quote here from Fodor 1997, p.110):

1. (possession conditions) If a concept C bears R to a concept C*, then having C* requires having C; and

2. (semantics) If a concept C bears R to a concept C*, then the content of C* is determined (at least inter alia) by the content of C.

We’ll see presently that what distinguishes S-architectures from Classical architectures is what relation they say R is.

Relation R, according to Classical theories, is of course the *constituency* relation, interpreted as (roughly) a part/whole relation, and (precisely) a co-tokening relation (Fodor 1997, p.111). This secures (1) since one can’t token a concept one doesn’t possess, and taking R to be co-tokening or constituency also serves to enlist (2) as “a natural explication of the informal idea that mental representations are compositional: What makes them compositional is that the content of structurally complex mental symbols is inherited from the contents of their less structurally complex parts” (Fodor 1997, p.111).

Fodor takes relation R in Smolensky’s account to be a “derived constituency” relation wholly parasitic on (real, Classical, co-tokening-ensuring) constituency: “C is a *derived constituent of* vector V iff V (uniquely) encodes C* and C is a constituent of C*” (Fodor 1997, p.113). Aside from parasitism on Classical architectures, the main problem here, as already mentioned, is that derived constituents are not Classical constituents of the vectors in terms of
which they’re defined, so they can’t play causal roles in processing when the complexes are
tokened, and thus can’t help explain the facts that motivate the compositionality thesis.

Among the facts in question are those like the following (Fodor 1997, p.111):

There seems to be a strong pretheoretic intuition that to think the content brown
cow is ipso facto to think brown. This amounts to considerably more than a
truism; for example, it isn’t explained just by the necessity of the inference from
brown cow to brown. (Notice that the inference from two to prime is necessary
too, but it is not intuitively plausible that you can’t think two without thinking prime.)

Fodor claims that Smolensky architectures are unable to explain such facts, since in those
architectures, one could token a representation whose content is BROWN COW (the encoding
of the relevant tree structure) without tokening any representation whose content is BROWN.

I will spend a moment on this example, because I think attending carefully to it reveals
that it doesn’t get you what Fodor thinks it does. First, note that Fodor does not employ his
usual notation for referring to concepts (e.g. BROWN, BROWN COW) in stating this example.
This, I speculate, is probably because doing so would beg the question insofar as it licenses
inference to constituent mental representations whose contents are just BROWN, COW, etc,
whereas the question at issue with respect to compositionality is precisely whether thoughts are
language-like in this way and accordingly decomposable into meaningful syntactic constituents.

But if we are not talking about syntactic constituents from the start, it’s not clear what this
intuition amounts to. What Fodor says, again, is that anyone who can think brown and cow can
also think brown cow, and the like. But what does the phrase ‘think brown’ amount to, if not to
‘think BROWN’? And what exactly would the latter amount to? I take the phrase ‘think brown’
to be, strictly speaking, ill-formed, and ‘think BROWN’ to be hostage to something like the LoT
hypothesis (in effect, it would mean “token the concept BROWN”).

Since Fodor means to be discussing semantic constituency without (I have supposed)
begging questions about the syntax of thought, a charitable interpretation is that Fodor is
making the following point: if I am thinking about brown cows, then (necessarily) I am thinking also about brown things, i.e. I am tokening some representation of brown things. Similarly for possession conditions: if I can think about brown cows (if I possess this cognitive capacity), then I can think about brown things too, and if I can think about brown things and about cows, then I can think about brown cows.

But recasting things in this way so as to focus on semantics while being agnostic about syntax, it is clear that these intuitions furnish no argument at all for a Classical architecture (construed narrowly in terms of explicit syntactic constituency). In fact, it's not clear how any cognitive system could be constructed so as to violate the constraint that when I think about brown cows I think about brown things, given only that the brownness of brown cows figure in the semantics of representations of them.

In addition to semantic intuitions like these, however, Fodor argues (much more persuasively, I think) that the Classical constituents of a complex representation can in part explain its role in mental processes. It would seem that Classical constituency is needed to drive, for example, an inference procedure sensitive to the kind of structure captured in first-order logic (see also §2.8 and §§6.1—6.2 below). In (Fodor and McLaughlin 1995), Smolensky's talk of "representing" structured symbols, mentioned in §2.1, is critically exploited to make this point: “It’s not...in doubt that tensor products can represent constituent structure. The relevant question is whether tensor product representations have constituent structure” (in the Classical sense) (Fodor and McLaughlin 1995 pp.214-215).

Smolensky's theory does invoke a specific relation between the representations that correspond to constituents and those that correspond to entire trees: namely, the latter are computed from the former by means of the tensor product binding and superposition functions. This, I think, is really what Fodor's relation $R$ amounts to in VSAs (see the discussion in §2.7
below). It is clear, however, that this relation does not entail co-tokening: in general, the vectors computed for trees needn’t have any components in common with those used to represent their constituents, nor would they be expected to, except perhaps occasionally by accident.\footnote{McLaughlin (2014) thinks otherwise, and claims that Smolensky’s representations are, or could easily be chosen so as to be, essentially Classical. I just flag this problem for now (see also footnote 19 above), and will return to it.}

Smolensky claims that this computational relation amounts to a kind of constituency, but given how different it is from Classical constituency, I am not sure what turns on the label. Fodor and McLaughlin raise the following worry about such a computational notion of constituency in any case: there are, for real-valued nodes, \textit{infinitely many} ways of decomposing a given vector into components, so unless some reason is found for privileging some of them, this does not seem a very useful notion of constituency.

Constituency aside, Smolensky has at least shown that a connectionist system can be set up such that, for every Classically structured representation, there is a corresponding distributed vector representation, and that the representations corresponding to complexes can be systematically (and productively, up to a certain limit) derived from a finite stock of simples. This may go a great distance toward an explanation of systematicity, as I’ll argue in §§2.6—2.7 below. First, I consider one more objection from Fodor.

In response to Smolensky’s work on tensor products, Fodor and McLaughlin introduced what is in effect a new, more difficult systematicity challenge for connectionism. Like the second, more powerful version of Saint Anselm’s Ontological Argument (Anselm 1965), this argument trades on necessity, not mere existence. Fodor and McLaughlin point out that even if a connectionist architecture \textit{could} be set up so as to respect systematicity, nothing in the models under consideration \textit{requires} this. But this is the wrong modality: to capture the
psychological necessity of such facts as that thinking *brown cow* requires that one thinks *brown*, systematicity should follow necessarily from the architecture of the system. It doesn’t in VSAs, since they don’t necessitate relevant co-tokening relations.

We’ve seen that these semantic intuitions really cut no ice with respect to constituency. Any system that can represent brown cows *ipso facto* (and necessarily) represents brown things, which is all that Fodor’s claim amounts to, unless question-begging. However, the necessitarian argument may be motivated in other ways. Systematicity’s following, if it does, from a LoT architecture, together with the pervasiveness of systematicity in cognitive systems (taken as an empirical fact), perhaps constitutes an abductive argument in favor of Classicism, given that connectionists would not predict systematicity without further premises. This argument will be addressed in the remaining sections of this chapter (see also §4.1 below).

2.4 McLaughlin’s take on Smolensky architectures

I take the foregoing criticisms of VSAs to be a mixed bag. First, I want to make explicit the reasons that I agree with Smolensky and with Fodor, and disagree with McLaughlin (2014), about Smolensky’s proposal’s involving Classical constituency. I think Fodor and McLaughlin are right: it doesn’t. So how does McLaughlin argue that it does? The argument is quite complex, and comes in two parts.

The first exploits the technical difficulty about the representation of constituents at different depths in a tree structure, mentioned in §2.1 above. McLaughlin discusses two of Smolensky’s proposed solutions to this problem: one may (a) use the direct sum between vector spaces instead of simple addition to combine vectors, which in effect results in a representation of each complex symbol as a set of vectors of different dimensionalities, one for
each level of tree depth (McLaughlin 2014 p.60), or (b) one may “pad” the tensor product representations for shallower constituents with “dummy’ vectors that don’t interfere with the semantically relevant processing, such that all of the representations end up having the same (presumably high) dimensionality.

Either way, McLaughlin claims, the conclusion that representations of complexes involve literal representations of their constituents is facilitated. In the first case it is facilitated because, if one “stratifies” the representation of tree structures in this way, each level of depth in a tree structure gets represented by its own dedicated pool of units. Thus, there is no interference of representations at different depths with one another. In the second case, it is facilitated for similar reasons: by adding extra dimensions to the vectors that correspond to shallower tree elements, so as to bring all the vector representations to the same high dimensionality (or, equivalently in this context, high tensor rank), one in effect creates space in the representation for each constituent to be represented without overlap.

The reason representation of constituents is nonetheless only facilitated is that whether the opportunity to explicitly represent constituents is exploited depends on how one chooses the role vectors used for the binding operation. If a set of orthonormal basis vectors are chosen (for example, [1, 0, 0], [0, 1, 0], and [0, 0, 1] in three dimensions), the multiplication operations will in effect select one column of the resulting matrix for representation of each bound constituent. It may be helpful to look at an example: suppose we have the three role vectors just mentioned, and filler vectors $f_1 = [a, b]$, $f_2 = [c, d]$ and $f_3 = [e, f]$. The complex structure $S = f_1 ⊗ [1, 0, 0] + f_2 ⊗ [0, 1, 0] + f_3 ⊗ [0, 0, 1]$, i.e. the conjunction of each of the fillers in distinct roles, then results in the matrix $S = [[a, b],[c,d],[e, f]]^T$, which in vector form would simply be a concatenation of the representations of the constituents: $[a, b, c, d, e, f]$. 
However, a different set of basis vectors, say \([1, 2, 4], [3, 5, 7], \) and \([4, 5, 6], \) may also be linearly independent without leading to transparent representation of the constituents (and thus, without leading to contiguous co-tokening of the constituents in the final representation). For example, the set of role vectors just mentioned would lead to the following algebraically inelegant vector representation for \(S: \) \([(a + 3c + 4e), (b + 3d + 4f), (2a + 5c + 5e), (2b + 5d + 5f), (4a + 7c + 6e), (4b + 7d + 6f)].\)

This brings us to the second part of the argument. McLaughlin (2014, p.63) says:

From a mathematical point of view, it makes no difference which linearly independent base vectors we use [as role vectors]. But it can make a difference from an explanatory point of view along the epistemic dimension of explanation… There is an explanatory value...to using normal base vectors for the role vectors since, then, the vectors to which trees are mapped will display the activation values of the units in the distributed activation states that are the atomic symbols in the tree.

And shortly thereafter, "if an ICS\(^{32}\) architecture contains concatenated constituents, it does so whatever linearly independent vectors we assign to the role vectors. If we do not use the base vectors for the role space, all that follows is that the vectors that represent trees will not display the atomic symbols that are constituents of the trees, though they will of course represent them" (pp.63-64; footnote mine).

I am confused on several counts by this final step in the argument. First, I am not sure what McLaughlin means by the claim that the various choices of linearly independent role vectors are mathematically equivalent. Obviously, the vectors \([1, 0]\) and \([0, 1]\) are not equivalent to the vectors \([1, 2]\) and \([3, 5]\), although various equivalence relations hold between these pairs. But there are all sorts of reasons one might prefer the first representation to the second that depend not on the human interpretability of the system but on the fact that some

---

\(^{32}\) McLaughlin uses the term “ICS (Integrated Connectionist / Symbolic) architecture” in place of Smolensky’s “ICA” and my “VSA”.\)
sets of numbers facilitate efficient computation. This aside, it seems clear that concatenated constituents arise only on certain choices of role vectors.

Part of what is going on in McLaughlin’s reasoning is that as noted in footnote 19 above, he distinguishes sharply, unlike Smolensky (and no doubt many other connectionist modelers), between patterns of activation in a connectionist model and the “activity vectors” that specify them. Activity vectors represent activation-patterns, McLaughlin suggests, just as the vector space representation of a dynamical system as a point in a high-dimensional state space represents the state of the entire system without implying that the system itself is just a point.

This may be the crux of the issue: because McLaughlin distinguishes activity patterns from vectors, he perhaps thinks it’s arbitrary which set of vectors you use to represent the activity patterns, i.e. the symbols in this case. He apparently thinks it’s just a matter of adopting a convention for describing the connectionist network that makes what’s going on within the system more readily observable: if normal basis vectors are used for the roles, then vectors encoding trees contain, as it were, iconic representations of the atomic representations that contribute to their meanings (“display”), rather than more or less arbitrary pointers.

But this is incorrect. First of all, the activity vectors in connectionist models are generally chosen so as to be as isomorphic to relevant neural states as possible, insofar as this facilitates computation. For example, one way of justifying the choice of a binary threshold activation function is that it provides a rough model of the way firing rates depend on membrane potentials in real neurons (Hopfield 1982), and the “rectified linear” activation function (which just takes the minimum of 0 and the summed, weighed input to a unit) might be justified by appeal to the fact that firing rates are always 0 or greater. There is, of course, room for arbitrary conventions that map numbers to neural activation states in different ways, but those conventions are never actually employed for obvious reasons.
In any case, choosing a set of role vectors to use in a VSA is not the same thing as choosing among conventions for representing unit activities, or firing rates in neurons. Generally, the distributed patterns of activity in connectionist networks, and the ways in which they depend on one another and on inputs and weights, are a result of the activation functions chosen for the units, which constrain the values the units may take as well as how they are determined by their inputs. These activation functions, as just discussed, are modeling choices motivated in the same way as other such choices (by desired function and considerations of biological plausibility, at least). The mapping from numerical vectors to properties of real neurons would be determined prior to, or at least along with, the choice of role vectors. Put simply: granting McLaughlin’s distinction between unit activations and the activity vectors that represent them, there is a real choice as to which unit activities are used to represent the roles, and it’s this choice on which the issue of Classical constituency turns.

Moreover, a realistic implementation of a Smolensky architecture would have to provide a biologically plausible way of arriving at a set of linearly independent role vectors (if the system is required to function at the level of precision that requires such independence). It would be ideal, for analysis of the system (along the “epistemic dimension of explanation”, perhaps), if the role vectors selected on the basis of these constraints turned out to be orthonormal. Such beautiful simplicity is often to be found in successful scientific theories, so one might hope that under idealization, at least, the role vectors that satisfy all these constraints would be normal basis vectors. This would vindicate the claim that VSAs involve Classical constituency.

But whether this is how things would actually fall out is of course an empirical question. More to the point: suppose we could design a system in which the role vectors are normal basis vectors as well as functioning in desirable ways within the overall model. In this system, computing a vector for a tree from the vectors for atomic symbols will amount to concatenation.
But this would be a *different* system from one that doesn’t enable concatenation, and this doesn’t show that the *tensor product representation* as such is just an implementation of Classicism. It shows that one of them *might* be.

This, I take it, is just one of the points made against VSAs in (Fodor and McLaughlin 1995): the architecture itself doesn’t guarantee a co-tokening relation. But even if concatenation like that discussed above, and hence Classical constituency, happened to fall out of independently motivated constraints on the selection of role vectors, we would still need a reason to think that concatenation is explanatorily relevant—that is, not just “along the epistemic dimension” in terms of making what’s going on in the system easier to grasp, but in terms of playing a causal role in the sort of processing that leads to systematicity.

About that, I will have more to say in the next section. But to sum up my criticism of McLaughlin’s criticism of Smolensky: First, it is misleading to suggest that choosing a certain basis for the role space only affects the *epistemic* dimension of explanation. Rather, it affects the explanation itself in its essentials, because it affects *which cognitive architecture* is being specified (at a fine grain). As I remarked above in §2.1, there is a world of difference between representing a tree in a way that makes its constituent structure explicit, and representing it in some other way. The *tokening of syntactic constituents* as parts of a syntactically complex representation is part of the very explanation of systematicity that Classicism has to offer. The claim that the representations of tree structures in VSAs would have constituents, in the sense to which concatenation is relevant, regardless of the choice of role vectors, is simply false. Without the assumption of normal role vectors, the most one can say is that explicit representations of constituents could be derived as needed.

The following argument seems to me somewhat decisive. McLaughlin’s claim that the choice of role vectors doesn’t matter (except in some innocuous mathematical way or
epistemically, in terms of the intelligibility of the resulting explanation) is correct only if this choice has no functional consequences within the system. But Classical compositionality is defined in terms of a necessary co-tokening relation between constituents and the complexes in which they occur. Some choices of role vectors yield co-tokening, and others don’t. Therefore, on McLaughin’s assumption of functional equivalence, Smolensky must have provided a non-Classical explanation of systematicity.

2.5 Holographic Reduced Representations

One reason to think that VSAs can explain at least some aspects of the systematicity of cognition without appeal to Classical constituency is that such systems have been implemented and shown to work on small-scale toy problems. Smolensky mentions (toward the end of Smolensky 1990) that the tensor product representation has been used to implement production systems, and he devotes some effort to showing that the representations used in a variety of preexisting connectionist models can be interpreted within his framework. Furthermore, a slew of proposals since Smolensky’s have replicated its essential features while improving on it substantially in certain ways. These systems have been used to build effective connectionist models, though they have not, so far as I know, been used in any applications at scale.

Like the tensor product system, these newer models bind values to variables using a multiplicative function and superpose them using addition in order to represent complex structures. But crucially, they avoid the problem of exploding dimensionality of the vectors that encode more complex structures, by constraining output vectors to have the same dimensionality as the inputs. Among the most interesting proposals of this sort is Tony Plate’s “Holographic Reduced Representation” (HRR) system (Plate 1995).
I indulge in a brief digression on HRRs before returning to the main question of how useful processing in VSAs may be possible. In the HRR system, the vectors for atomic representations as well as for various molecular ones all have the same dimensionality, and this without any ad hoc padding or stratification tricks of the kind discussed above. HRRs, moreover, were motivated not simply by the need to explain systematicity, but (relatedly) by more specific concerns of Geoff Hinton’s (Hinton 1990) about how processing of hierarchical structures within connectionist systems may be possible.

The way in which HRRs achieve parity in the dimensions of the vectors is to use a different mathematical operation in place of the tensor product, called *circular convolution*. This operation, for input vectors of dimension $n$, yields an output vector of dimension $n$ as well, via the following equation: 

$$t_j = \sum_{k=0}^{n-1} c_k x_{j-k}, \text{ for } j = 0 \text{ to } n - 1 \text{ and for subscripts modulo-} n \text{ (e.g. where a value of 0 for } j \text{ is treated as } n),$$

and where $t$ is the output vector and $c$ and $x$ are the convolved vectors. This equation takes some examining to get the hang of, but as illustrated in Figure 3 below, it can be regarded as performing a compressed outer product on $c$ and $x$.\(^{33}\)

---

\(^{33}\) See (Plate 1995, p.3). Here, Plate is thinking of $x$ as the “filler” vector and $c$ as the “cue” vector. This makes sense since, as discussed above, role vectors can be used as cues to retrieve their bound fillers in Smolensky’s system.
Figure 3 (After Figure 4 in Plate 1995).

Illustration of the circular convolution operation, applied to 5-dimensional input vectors \( c \) and \( x \) to produce a 5-dimensional output vector \( t \). The node at row \( i \), column \( j \) in the grid corresponds to the product of element \( i \) of \( x \) and element \( j \) of \( c \) (so, for example, the value at the bottom-right-most node is \( x_4^*c_4 \)). Thus, the grid depicts the outer product of \( x \) and \( c \). In circular convolution, the values of the nodes along the arrows whose heads are labeled \( t_0 \) through \( t_4 \) are summed to yield the corresponding elements of \( t \) (in higher dimensions, a similar pattern occurs, in which the \((n-1)\)th element of \( t \) is a sum along the diagonal and the preceding elements of \( t \) are sums along “circular” paths around this diagonal. Thus, \( t \) is a (lossy) compressed representation of the outer product.

Moreover, apart from the compression, this system has the essential properties of the tensor product representation. In particular, fillers bound to roles can be combined via addition, just as in Smolensky’s system. Further, an operation called circular correlation is the approximate inverse of circular convolution, and can be used to retrieve embedded vector \( x \) from a representation in which it is superposed, together with \( x \)’s “cue” or role vector \( c \): \( y = x = c \ (\#) \ t \), where \((\#)\) is circular correlation (see Figure 5 of the paper and associated equation). Accurate retrieval depends, as in the case of Smolensky’s system, on certain assumptions about the vectors. In this case, both “cue” and “filler” vectors must be drawn independently from
a distribution with mean 0 and variance $1/n$ (for example a normal distribution), where $n$ is the dimensionality of the vectors (Plate 1995, p.4).

The capacity of a system of HRRs, if each vector $t$ is viewed as a memory trace, is, in terms of the size of the stored structures (e.g. the number of role/filler bindings involved) roughly linear in the dimensionality of the vectors (see Plate 1995, Section IX). Like Smolensky, Plate details a wide variety of mathematical properties of his system, and shows that its retrieval accuracy when using a cue degrades slightly (but not catastrophically) for more deeply embedded structures. Moreover, HRRs have been deployed in functioning systems: Plate (Plate 1992) discusses an implementation of HRRs in a recurrent neural network for sequence learning, and (Eliasmith 2005) uses HRRs to implement a simple model of the Wason selection task (Wason 1968) that reproduces the effect of context on performance.

An important downside (though not a major one) of convolution-based memories like the HRR system is that because of the compression, the exact stored vectors cannot in general be retrieved by the “unbinding” procedure, even under ideal conditions. However, under the appropriate constraints, the differences between the retrieved vectors will be large enough to allow “recognition” of whether a particular item has been stored, which can be converted into full retrieval by the use of a cleanup memory, as discussed earlier, which stores a copy of each vector previously input to the system.

I have detailed the HRR system, apart from its intrinsic interest, in order to show that VSAs exist in which some of the most counterintuitive consequences of the tensor product system (such as the exponential explosion in number of units needed for increasing depth of tree structures) can be avoided. I have also done so in order to drive home the point that VSAs exist that (a) are of use in real connectionist models of tasks traditionally addressed by Classical architectures, such as the processing of sequences in language and action, and (b) definitely do
not require or even allow the co-tokening relation. There is no question, in an HRR system, of
the vectors for the constituents of encoded trees occurring as parts of the vectors for the entire
tree. The input vectors (which are random variables in the first place) are hopelessly scrambled
in the lossy encoding process, and the matched dimensionality of input and output vectors
means that transparent “display” of more than one constituent (input) vector is impossible.

2.6 Processing in Vector Symbolic Architectures

Note that VSAs do not in general exhibit the feature, discussed above in §1.3, that
similar vectors represent similar things, so they cannot explain systematicity by appeal to the
idea that semantic relations are captured in the structure of a vector space (Horgan and Tienson
1996). Nor do they appeal to Classical constituency, as just argued. So, how exactly do they
explain the facts of systematicity, productivity and the like, or function as components of
explanations of cognitive capacities more broadly, if they do? Fodor, taking Smolensky at his
word, supposes that the representations of constituents in VSAs function only in some kind of
“imaginary”, “derived”, or otherwise virtual way, and thus that Smolensky architectures simply
don’t offer any explanation of systematicity (see §2.3 above).

It seems to me that these forms of representation do command resources sufficient to
explain at least some aspects of systematicity, though one can see this, I argue, only by
attending to the way in which such systems actually function and speculating a bit about their
further implementation and use, and not so much by heeding Smolensky’s remarks concerning
the non-causal role of symbols in processing, and the like. In particular, I take the extent to
which constituent structure is causally inert in processing within VSAs to have been somewhat
exaggerated (whether this is Fodor’s fault, Smolensky’s, or both, I will not here consider further).
One important clue is the unbinding operation. Fodor et al do not so much as mention this operation, but it is fairly easy to see how it could be used, along with its inverse, the binding operation, to process tree structures in a way that respects the constraints imposed by their syntactic structure. Moreover, I will argue that the reasonable use of these operations in practice would secure the feature of a conceptual system that Fodor really cares about, namely that possessing concept C* entails possessing C just in case C bears R to C*.

First, consider encoding. Although the tensor product representation corresponding to a given tree structure, assuming a set of atomic “filler” vectors (and role vectors), is well-defined whether or not it is ever computed, it’s reasonable to assume that representations of complexes would actually be derived from the representations of constituents during processing. Such an encoding process would begin with the explicit representations of roles and fillers, and progressively superpose these representations on the binding units, in Smolensky’s architecture. (Plate 1995) shows that analogous processes for HRRs can be implemented straightforwardly in standard connectionist nets using weight matrices.

It is easy to imagine applications of this kind of process in any domain that requires hierarchical, tree-like structures, such as natural language parsing or the construction of plans for action.

Second, the emphasis placed in (Smolensky 1990), (Plate 1995) and elsewhere on unbinding procedures and the recoverability of vectors under various conditions is hard to understand unless these procedures are meant to be used during processing. Moreover, it is easy to imagine practical uses for the binding and unbinding operations, ones that help overcome many of Fodor et al’s objections to VSAs. For instance, the fact that the

---

34 This is possible thanks to an equivalence between circular convolution and a combination of element-wise vector multiplication with Fast Fourier Transforms—see (Plate 1995, Section VIII.B, as well as Eliasmith 2005).

35 Fodor (1997) alleges that these theoretical guarantees are meant to provide only some kind of security blanket, a proof that vector representations can be mapped one-one to syntax trees. But the uses to which VSAs have been put belie this idea.
representation of a syntactically complex object can be “cued” with a role vector to extract its constituent shows that, even though VSAs do not employ the co-tokening relation, such constituents can be recovered at will, as noted in §2.4. This has obvious applications for things like reasoning in ways sensitive to constituent structure.

A distributed representation of a sentence such as ‘Sam swims before dawn’ could be cued with a subject or agent role vector to recover the vector for ‘Sam’, with a verb cue to recover ‘swims’, etc. Obviously, this general idea is consistent with many different precise ways of defining the roles. It is easy to imagine how this kind of mechanism could be used to enable a system implementing a VSA to respond rationally to queries like “Who swam?” and “What did Sam do?”, or to infer ‘Sam swims’ from the sentence in question, assuming that associative mechanisms of the right sort could call up the appropriate role vectors. This mechanism could be tested for empirically. It predicts, for example, that ceteris paribus at least, inferences involving more deeply syntactically embedded terms would take longer, since more deeply embedded structures would take a chain of unbinding operations to retrieve.

Why ceteris paribus? As discussed by (Horgan and Tienson 1996), Smolensky points out that an entire chain of syntactic derivations can in principle be accomplished by the application of a single weight matrix to the input vector. There will in many cases be no need to token representations of the intermediate steps. This is perhaps part of Smolensky’s reason for denying that syntactic constituents should be conceived of as causally efficacious in processing.

---

36 Lest there be any doubt about this: when I say “the vector for ‘Sam’”, I mean the vector that represents Sam. These contents would be assigned, I assume, on the basis of functional role, and in particular on the basis of a structural-representation relation, as in the “interpretational semantics” advocated by Robert Cummins (1991). I will not defend that assumption here, but see (Kiefer and Hohwy 2017). As it happens, since the ‘Sam’ vector will play the same functional role as the Classical constituent ‘Sam’, it could serve as a representation of the latter as well, on this assumption.

37 There are also probably neat explanations here for garden path effects and related phenomena in natural language processing, which I will not explore here.
There are good reasons for this view, as I’ll discuss presently, but it certainly muddies the dialectical waters.

First, consider that the basic operation in a connectionist network is the mapping of one vector space to another via matrix multiplication (where the vectors in question are the representations in adjacent layers of the network and the matrix is a collection of “synaptic” weights). In particular there are matrices $W_L$ and $W_R$ that can be applied to a vector representation corresponding to a binary branching tree, and used to extract its left and right components, and matrices can then be used to perform further operations on these vectors, including further binding operations. But since matrix multiplication can be iterated just as can scalar multiplication\(^{38}\), we can just construct a single matrix $W$ that combines all these steps into one. Thus, any computation over trees, however complex, can be performed in a VSA by a single application of some matrix, without computing representations of the tree’s constituents.

I don’t think this threatens the picture of pseudo-Classical processing within VSAs that I’ve been painting, however. The $W$ in question was arrived at analytically, by examining the kind of step-by-step, constituent-sensitive procedure that would occur in a Classical system. Though it might be useful to derive such composite matrices in the course of some learning process to speed up habitually applied transformations, this explanation is available only if the system originally performed the sequence of operations in terms of which the composite $W$ was defined, and in this process, the vector representations of constituents would have been computed as intermediate results.

Smolensky’s observation that all of these individual transformations could be compressed into a single vector representation is potentially related to a proposal discussed in theories of motor control. In the vocabulary of reinforcement learning (see e.g. Friesen and Rao

---

\(^{38}\) Provided the matrices have the right dimensions—the rule is that an $n \times m$ matrix multiplied by an $m \times o$ matrix yields an $n \times o$ matrix, and the result is undefined if the dimensions don’t match up in this way.
2010 and Botvinick 2008, Box 2), a set of hierarchically related decisions in the unfolding of an action plan can, in overlearned cases, be replaced by a single “option” that “chunks” the actions that achieve several routinely associated sub-goals into a single step. Something similar may occur in syntactic processing. Thus, VSAs seem to combine the Classical virtue of allowing for constituent-sensitive operations with the additional flexibility of being able to skip the tokening of constituents when relevant mappings have once been well-learned.\(^{39}\)

### 2.7 Compositional VS combinatorial systems

I think the foregoing discussion of VSAs suggests that they can, after all, play the kinds of roles in mental processes that Fodor and Pylyshyn (1988) rightly take to be diagnostic for genuinely cognitive representations. However, it could stand to be spelled out in more detail how exactly (and indeed whether) they meet the systematicity challenge, and how they might manage to do so without appeal to Classical constituency.

I will begin by offering a characterization of what Fodor’s relation \(R\) amounts to in Smolensky’s tensor product architecture (and, in fact, in all of the VSAs I’ve discussed) that I take to be far more accurate than the one Fodor gives. Recall that Fodor identified relation \(R\) in Smolensky’s theory with “a kind of derivation relation” (Fodor 1997), specifically, the “derived constituent” relation. I also take \(R\) in these systems to be a derivation relation, but rather a system-internal one: C bears \(R\) to \(C^*\) just in case \(C^*\) can be computed from C (together with the other constituents of \(C^*\)).\(^{40}\)

\(^{39}\) This complicates the empirical test of the theory mentioned above, based on the idea that processing time should scale with depth of embedding in a tree structure. But of course, more sophisticated experiments could be devised to test for whether this relation holds in overlearned versus novel situations, and potentially provide even more rich evidence for (or against) the model.

\(^{40}\) One may object that this definition is too permissive since it seems to cover inference from \(P\) to \(P^*\). I mean to assume that \(C\) and \(C^*\) are representations of the right semantic types to stand in constituency
This satisfies Fodor’s principle (1), about the possession conditions of concepts (see §2.3 above), provided that “requires” involves a modality defined in terms of the idealized operations of the cognitive system in question. That is: except by freak accident, $C^*$ is a representation that will only ever arise in the system as a causal consequence of having represented $C$ (along with the other constituents of $C^*$). This claim is underwritten by the functional architecture that I am stipulating the system to have: $C^*$ is constructed as needed by processes of natural language perception, inference, action planning, and so on, that at least initially and perhaps typically depend on the prior representation of constituents $C_1$, $C_2$, etc.

The point about direct, overlearned associations does not affect this conclusion, since the associations must have been learned by dropping out the intermediate steps that were originally computed explicitly. Moreover, the recoverability of constituents from a given $C^*$ is underwritten by an analogous principle of computational derivability. Thanks to the unbinding operations together with compositionality in the forward direction, any system that tokens $C^*$ can also token $C_1$, $C_2$, and so on.

To be clear: the computational derivability relations in question support systematicity because the same generic operation can be used to combine any two atomic representations that are available in the system. If I have a vector for dogs, one for cats, one for conjunction, one for disjunction, and an appropriate assortment of role vectors, I can equally well compute a representation for ‘Dogs or cats’, ‘Cats or dogs’, ‘Dogs and cats’, and ‘Cats and dogs’. Moreover, productivity (insofar as it exists) is also thereby explained: up to the saturation point fixed by the capacity of the network, at least, I can continue to recursively embed clauses as much as I like.
It is worth pointing out that VSAs do exhibit compositionality as it is described in at least some passages in (Fodor and Pylyshyn 1988), though not according to the later, more strict definition in terms of constituency (Fodor and McLaughlin 1995). Smolensky’s system exploited a loophole of sorts in the original definition of compositionality, and in effect forced Fodor and others to become more conservative in their definition of Classical systems.

Hardline Classicists take the relation between brain states that corresponds to the “part of” relation between simple and complex symbols in an external, public language also to be the “part of” relation. This is just the idea that complex symbols contain their semantic constituents as syntactic parts, applied to mental representations. See (Fodor and Pylyshyn 1988, p.13):

...the symbol structures in a Classical model are assumed to correspond to real physical structures in the brain and the combinatorial structure of a representation is supposed to have a counterpart in structural relations among physical properties of the brain. For example the relation ‘part of’, which holds between a relatively simple symbol and a more complex one, is assumed to correspond to some physical relation among brain states.

But, as the ensuing footnote (9) makes clear, the move to literal constituency isn’t necessitated by the assumption in the quote. There must simply be a dependence of the “causal relations among brain states” on “the combinatorial structure of the encoded expressions”, presumably just in the sense that the former track the latter reliably. What’s required is that $F(P\&Q) = B(F(P), F(Q))$, where $F$ is a function from symbols to brain states, and $B$ specifies some physical/causal relation between brain states (Fodor and Pylyshyn 1988, fn.9). So, the principle states that whatever brain state serves as the vehicle for $Q$ must be related to whatever brain state serves as the vehicle for $P$ via the relation $B$ to yield a representation whose content is $P\&Q$, and the same must hold for all representations whose contents are related in this way.

I imagine that Fodor and Pylyshyn intended hereby to enforce the syntactic constituency relation explicitly enforced in (Fodor and McLaughlin 1995). But it’s not necessary to posit
constituency in that sense in order to satisfy the requirements on \( B \) just mentioned. The tensor product representation in effect takes relation \( B \) to be the outer product plus element-wise addition. There is a subtlety here that perhaps Smolensky is too glib about: Fodor and Pylyshyn require \( B \) to be a physical relation between brain states, while Smolensky concedes that his tensor product mapping is a mathematical relation, not a physical one. I am tempted not to make much of this difference: if we assume that the mathematical relations themselves have a precise physical meaning once an implementation of the tensor product system has been settled on, there is no harm in seeing Smolensky’s \( B \) as an abstract placeholder for whatever actual \( B \) obtains in the brain.

That said, it may be instructive to examine just how this loophole works given a particular implementation assumption. Assume that activities of nodes correspond to population rate codes, one for each node.\(^{41}\) Then the outer product \( u = v \otimes w \) between vectors \( v \) and \( w \) corresponding to (meta-)populations \( V \) and \( W \) maps onto to the following physical relation: the average firing rates of the groups in meta-population \( U \) are given by the outer product of those in \( V \) and those in \( W \). Provided again that there is an independently fixed way of measuring population rates, i.e. mapping them onto scalar values (see §2.4 above), this is a well-defined physical relation among brain states. Similar remarks apply to vector superposition (‘+’).

One way of seeing how the loophole works is to note that both ‘\( \otimes \)’ and ‘+’ map onto diachronic relations between neural states. Note, however, that the diachronic aspect is not essential. Even if we imagine that the states of \( U \) depended instantaneously on those of \( V \) and \( W \), or that things were arranged so that all three populations were always active (with their relevant values) at once, we still would not get the result that the semantically atomic representations were physical parts of the semantically complex representation. We may get

\(^{41}\) This is in fact a common assumption, motivated among other things by the need for robustness against noise. See e.g. (Eliasmith 2005).
co-tokening, but not necessary co-tokening. It is worth noting that if distinct vector spaces are used to represent the filler, role and binding vectors, one will predictably get co-tokening, at least of a total tree representation and the constituent currently being processed. Such co-tokening would not in this context do relevant explanatory work, however, because relatively rapid sequential tokening would accomplish the same thing.

Note that what’s really needed for the relevant mental processes, strictly, is not that the system have access to the constituents of a semantically complex representation in virtue of having access to the complex representation itself, however natural this may seem when we take, say, reading natural language sentences as our model. The system needs only to be able to access a representation with the content of the Classical constituents whenever it can access the content of the corresponding complex (Classical) structure, in some way that keeps track of the semantic role that the contributing representations play in the relevant whole. And this condition is met in VSAs, thanks to the unbinding operation.

The supposition that a new token is computed whenever a constituent is accessed may seem extravagant, and to incur unnecessary processing costs. However, even in a purely Classical system, mental processes that are “sensitive to” the presence of the constituent JOHN in JOHN LOVES MARY must somehow be directed at that constituent, as it were—one needs an addressing mechanism. The addressing mechanism can serve, in the Smolensky architecture, at the same time as an unbinding process (in fact, connectionist systems naturally model this sort of content-based addressing process, as I’ve noted already). So, computationally speaking, you’re not losing anything, not even efficiency (which it’s not like a Classicist to worry about, but still), by switching to a Smolensky architecture.

The account just sketched, however, goes only so far. Why, in a VSA, can’t constituents be bound to vectors willy-nilly, encoding trees like [[Displayed_{\text{NP}}[\text{NP}_{\text{det}}[\text{NP}_{\text{prep}}[\text{NP}_{\text{PP}}]])]_S ?
Why can’t I superpose two left-constituent role vectors? The machinery, purely in the abstract, allows these operations. The questions of how it is that role vectors are constrained to be bound only to appropriate filler vectors, and of how variable bindings are combined with one another only in the right ways, have not so far been raised. This is, in a way, an important challenge for VSAs. It shows that they might provide the semblance of an answer to the systematicity challenge for connectionism only by overgenerating representations. It’s not typically taken to be part of the systematicity thesis that a system should be able to represent \(aRb\) just in case it can thereby represent \(bRa\), but not \(RabR\) or \(abb\)—but maybe it should be.

This shows that the explanation of systematicity afforded by the VSA machinery on its own is in an important sense incomplete. Ultimately, I think a different style of connectionist system is to be preferred, whose processing would be automatically constrained by training data to yield only legitimate representations and inferences—see the material below on the RAAM system (Pollack 1990) and its descendants. But in terms of the present dialectic, it might be pointed out that even if Smolensky were to stipulate that a VSA, by definition, only allows those vector operations that correspond to well-formed parse trees, it would still provide a non-Classical route to accounting for systematicity, given these further constraints. It could also be argued that the fact that VSAs can in principle compute a wider range of representations than those licensed by, say, a generative grammar for English, is a virtue: there are many potential applications for a system that can encode arbitrary structures, and each domain to which the system is applied may contribute its own constraints to the set of representations, though this of course would necessitate further machinery.

---

42 It is important to note here, however, that very same applies to Classical systems insofar as they appeal simply to the idea of serial symbol processing as it might be instantiated in, for example, a Turing machine. The notion of a physical symbol system (Newell 1980) on its own doesn’t get you language-like syntactic constraints. Pollack (1990) exploits this point in reply to Fodor and Pylyshyn—see §4.1 below.
2.8 Inferential systematicity

The discussion thus far suggests that there are at least two ways in which systematicity among a cognitive system’s representations can be explained: (1) in virtue of its representations possessing Classical constituent structure, or (2) in virtue of its representations being derivable from one another in ways that conform to the structural relations among Classical constituents, without the computed results necessarily containing the input expressions as parts (Fodor’s relation $R$ is taken to be computational derivability).

In this section, I argue that the second sort of proposal can in principle accommodate what Fodor and Pylyshyn call “the systematicity of inference” (Fodor and Pylyshyn 1988, §3.4) as well as the systematicity of representation already discussed. According to Fodor and Pylyshyn, “Classical theories are committed to the following striking prediction: inferences that are of similar logical type ought, pretty generally, to elicit correspondingly similar cognitive capacities” (Fodor and Pylyshyn 1988, p.47). In other words, “You shouldn’t, for example, find a kind of mental life in which you get inferences from $P\&Q\&R$ to $P$ but not from $P\&Q$ to $P$” (p.47).

Continuing the quote, which runs with the example of simplification of a conjunction:

This is because, according to the Classical account, this logically homogeneous class of inferences is carried out by a correspondingly homogenous class of psychological mechanisms: The premises of both inferences are expressed by mental representations that satisfy the same syntactic analysis (viz. $S_1\&S_2\&S_3\&\ldots\&S_n$); and the process of drawing the inference corresponds, in both cases, to the same formal operation of detaching the constituent that expresses the conclusion.

There are really two interrelated ideas here. This class of inferences depends on (1) a uniform syntactic type in the premises, and (2) a uniform “formal operation” applied to representations of that syntactic type.
Fodor and Pylyshyn suppose the following: “it’s a psychological law that thoughts that \(P \& Q\) tend to cause thoughts that \(P\) and thoughts that \(Q\), all else being equal” (p.46). This generalization is obviously at some distance from observable behavior: the *ceteris paribus* clause is doing a ton of heavy lifting. To see this, consider that it’s presumably, on this view, also a psychological law that thoughts that \(P\), when accompanied by thoughts that \(Q\), tend to cause thoughts that \(P \& Q\), all else being equal. So, all else being equal, I will be stuck in a loop, constantly thinking \(P \& Q\), then \(P\) and \(Q\), then \(P \& Q\) again, etc. I take it that, cases of rumination aside, this kind of tight inferential loop simply doesn’t occur. Consider also addition of a disjunct: am I *ceteris paribus* prone to think \(P \lor Q\), \(P \lor R\), \(P \lor Q \lor R\), etc, whenever I think \(P\)?

Thus, it isn’t obvious that inference should be conceived of, at least across the board, as an automatic process that’s simply triggered when tokens of the right syntactic type show up in a cognitive system. Inferences are, I would argue in any case, more likely to occur under some circumstances than under others. Certain among the possible inferences one might draw from a currently entertained thought might be favored by contextual factors (see §6.4 below and Johnson-Laird 2010a), including one’s interests (e.g. “motivated reasoning”, which, however biased, is still reasoning). If I’m told by a reliable source that Sherlock Holmes is a liar and a thief, and I’ve just placed a bet on Holmes’s advice, I may immediately start reasoning on the basis of the first conjunct without much worrying about the second.

I’ll flag one conclusion that might be drawn from this, to set it aside for later: inferential processes can be biased (perhaps in subtle ways) by doxastic, as well as desiderative and volitional, context. For present purposes, what this kind of thing suggests is that, in addition to (1) and (2) above, we need some mechanism in virtue of which, when tokens of the right syntactic type occur, then (if additional conditions \(C\) hold) the formal operation responsible for

\[\text{43 Some of these worries at least could be mitigated by adopting (Quilty-Dunn and Mandelbaum 2017)’s agnosticism about the precise logic of thought.}\]
the inference is applied to those tokens. The underlying point, put abstractly: formal operations
defined over syntactic types are well and good, but they are not identical to procedures, and the
latter are needed for a full computational/representational explanation of psychological
processes like inference.

That’s the Classical account of inference, suitably qualified. How might a connectionist
account go? We’ve seen, above, that vector representations corresponding to syntactically
complex representations can be derived systematically from those corresponding to their
syntactic parts by application of the binding and superposition operations within VSAs. Perhaps
a similar approach to inference can be expected to work, since at a certain level of abstraction,
the syntactic constituency relation is formally the same as the premise-conclusion relation in at
least an important subset of valid arguments: in almost every inference licensed by
propositional logic, for example, the conclusion either contains the premises as parts or
vice-versa. And more generally, chains of deductive reasoning, like parse trees, can be
understood in terms of the recursive application of formally specified rules. Thus, the fact,
lamented at the end of the previous section, that VSAs put too few constraints on the vector
operations they allow, may indeed be put to good use.

Consider first the inferences licensed by propositional logic, which depend only on
logical connectives such as ‘and’, ‘or’ and ‘not’, where these connectives are construed within
the logic as truth-functions. One idea that is broadly in the spirit of VSAs would be to define

---

44 There are exceptions of the kind that “relevance logics” are meant to rule out: for example, arguments
with arbitrary premises and a tautologous conclusion, and the principle of Explosion. But perhaps the
“logic of thought” is a relevance logic.

45 Notoriously, it seems doubtful that the connectives employed in standard propositional logic really
capture their alleged natural-language counterparts in many cases. The worst offender is probably ‘⊃’,
which treats sentences like “If Bill Clinton was not the 42nd President of the United States, then Venus
exploded yesterday” as true, since their antecedents are false. Other difficulties arise in part because
such a limited range of connectives is used to represent a wide variety of natural language sentences in a
way that sometimes abstracts from nuances of the latter. For example, the conjunction operator is used
to capture sentences containing ‘but’ and ‘then’ as well as ‘and’. Some of this can be explained away by
relying on a semantics/pragmatics distinction. I think that the right approach in the case of the conditional
vector operations corresponding to each connective: $\text{CONJUNCTION}(p, q)$, for example, would be some function that returns a vector $r$ representing `$p \& q$`. This will work straightforwardly for inferences that map from atomic propositional representations to a conclusion, for example the inference from `$p$' and `$q$' to `$p \& q$`, or that from `$p$' to `$p \lor q$`. But what about inferences that depend on which connectives appear in input sentences, such as that from `$p \& q$' to `$p$' (or to `$q$')? How can we ensure that the representations of sentences derived via the $\text{CONJUNCTION}$ operator from input sentences $p$ and $q$ are such that, when the $\text{SIMPLIFICATION}$ operator is applied to them, they yield representations for $p$ (or $q$) as output, unless the computation taking the representation for `$p \& q$' to that for `$p$' (for example) is sensitive to constituent structure?

One natural way to approach this, within the specific class of architectures I've been discussing (i.e. VSAs), is to implement an inference rule like Simplification as an unbinding operation that retrieves `$p$' or `$q$' from `$p \& q$'. Of course, for this to work, `$p \& q$' must have been derived in the first place by somehow binding `$p$' and `$q$' vectors into a single resultant vector. A proposal of this kind that could be implemented using existing VSAs would be to define each connective implicitly by binding "filler" vectors to connective-specific role vectors. For instance, using the tensor product notation, `$p \& q$' could be represented by $r = [p \otimes \text{conj1}] + [q \otimes \text{conj2}]$, `$p \supset q$' by $[p \otimes \text{ant}] + [q \otimes \text{cons}]$, and so on. Negation could be represented by binding the $p$ vector with a $\text{not}$ role vector. Since any formula in propositional logic can be constructed by sequential application of unary and binary sentential connectives to propositional variables,

---

is to take it to be a modal operator outside the scope of ordinary propositional logic. In any case, I will skirt these controversies here as follows: take the account I'm offering in this section to apply to whatever portion of English can be uncontroversially captured just using the standard resources of propositional logic. Surely there are at least a few inferences in this set, such as that from "Sam is a dog and Melanie is a cat" to "Sam is a dog".

46 This is $\text{modulo}$ the points discussed above concerning the psychological reality of the derivation relations in Smolensky's system. It must at least be the case that the representation I'm calling `$p \& q$' would be the result if `$p$' and `$q$' were bound in the relevant way, even if this binding step is (often) skipped during processing.
more complex sentences present no special problems. For example, \( \text{NOT}(p \text{ AND } q) \) could be represented as: \( \text{bind}(\text{NOT},(\text{bind}(\text{CONJ1},p) + \text{bind}(\text{CONJ2},q)) \). Recursive application of the unbinding procedure in an HRR system can recover \( p \) and \( q \) from this example.\(^{47}\) And the representation of premises in an argument presents no problem as such, since any set of premises is equivalent to their conjunction.\(^{48}\)

Of course, as in the case of the construction of syntax trees, the system as just described would allow for the construction of ill-formed expressions. Moreover, it would be ideal if a simple system could be set up such that only representations corresponding to propositions that follow from a premise set could be unbound from the representation of that set, but this requires appeal to machinery beyond the basic VSA operations, since we don’t in general want the sentences used to construct a complex sentence to be inferable from that sentence. In particular, probing a disjunction or conditional with the \text{disj1} \text{ or cons} role vectors should not in general yield the relevant disjuncts or consequents as inferences. However, it should be possible to recover the vector for ‘\( q \)’ if ‘\( p \text{ v } q \)’ and ‘\( \neg p \)’ both occur in the premise set—similarly for other rules that take more than one premise as input, like Modus Ponens.

Leaving that issue aside for the moment, and moving on to first-order logic: it does not take much imagination to see how this kind of approach can be extended to the representation of predicate structure. First-order inference does introduce new challenges because it involves \text{substitution} rather than simply derivation of new propositions from the assumed ones. This can introduce items of knowledge extraneous to what is explicitly represented in the premises. For

\(^{47}\) As a sanity check, I have confirmed that this encoding and decoding operation works: for 100-dimensional vectors drawn from a normal distribution, the cosine similarity (see below) between the \( p \) vector originally stored and its reconstruction \( \hat{p} \) recovered from the \( \text{NOT}(p \text{ AND } q) \) vector is greater than that between \( p \) and any other vector in the set, by an order of magnitude on average.

\(^{48}\) It would be nice if lists of premises could simply be represented by superposition, since their order doesn’t matter, but in order for multiple instances of the same role not to be confused, some more complex structure is necessary.
example, inferring from ‘(∀x)(Fx)’ to ‘Fa’ depends on the fact that ‘a’ is a singular term in the system of representations. Relatedly, ‘a’ cannot be recovered by any series of unbinding operations from ‘(∀x)(Fx)’, assuming that the latter was encoded from its constituents as described. The inference from “Sam swims” to “Someone swims” (i.e. from ‘Ss’ to ‘(∃x)(Sx)’) likewise requires introduction of a quantifier not present in the premise. In order to implement first-order inference rules, then, it is necessary to appeal to processes that can retrieve and manipulate representations that don’t occur in the premises.

Thus, a functioning system would require extra machinery, though nothing beyond the capability of a relatively simple connectionist network. Plate (1995), for example, shows how simple “machines” can be built that use HRRs as modules. The system might be set up, for example, in such a way that the ‘q’ vector can be unbound from the ‘(p ⊃ q)’ vector only if the latter has previously been unbound from ‘(p ⊃ q) & p’. As in the case of syntactic constituency, VSAs have the requisite representational power, but must be supplemented by additional procedural constraints. However, as noted at the start of this section, even in a Classical system procedures are needed to recognize the syntactic form of an input representation such that the proper formal rules are applied to it, and arguably to determine in which cases a potential inference is actually carried out based on contextual factors.

2.9 Some problems for VSAs

I have spent the bulk of this chapter considering the merits of VSAs as answers to the challenge of systematicity for connectionist systems. Although I have found them to be more promising as genuine alternatives to Classicism than general consensus in the philosophy of
mind might suggest, I do not think they are actually the most promising route to a defense of connectionist models of sophisticated cognition.

It is telling that many of the most high-profile connectionist researchers apparently tend not to think so, either: as suggested by the selective survey of relevant work in machine learning below, VSAs are nowhere employed in large-scale practical applications of neural network models to tasks like natural language processing. As Gayler (2003) suggests, the lack of interest in VSAs on the part of researchers in this area might be in part due to one key respect in which these architectures differ from typical connectionist approaches: the latter tend to focus on statistical modeling and the learning of associations from data, while the former in effect employ fixed “logical” rules for the derivation of vectors from other vectors.49

In this section, I’ll explain why I don’t find VSAs as promising as the preceding discussion may have suggested, then pivot (in the next chapter) to a discussion of how the more basic, fundamental principles of connectionist modeling mentioned in §1.3 may, when combined with additional assumptions, have the right features to tackle the systematicity challenge. I think the detour through VSAs is nonetheless useful, even crucial, because apart from bringing the casual reader up to speed on the dialectic, it serves to abstract a key principle that may be of use in any response to this challenge: that systematic computational derivability can do (some of) the work of Classical constituency. As we’ll see, connectionist networks that have been deployed at scale on natural language processing (and, to a lesser extent, reasoning) tasks instantiate this principle. However, unlike VSAs, they rely on empirically learned transformations rather than a small set of fixed ones.50

49 The fact that such “rule-based” behavior could be implemented in connectionist nets without the use of actual constituents may be part of what Fodor found provocative about Smolensky’s work. The latter really was a departure from the typical “associationist” character of connectionist systems (without yet being Classical).
50 They also in some cases appeal to the embodiment of Classical syntactic structures in neural networks, as briefly discussed in §1.2.
Aside from their departure from associationism (which is a boon or a liability, depending on one’s theoretical perspective), what exactly is wrong with VSAs? One problem is that the tensor-product representation, HRRs, and other VSAs are primarily systems for encoding arbitrary structures. They are designed to store a range of distinct items in a memory such that they can be recovered later, not to enable the manipulation of specifically symbolic structures, i.e. structures with content, in such a way that the representational properties of the stored items are tightly linked to the properties of the representation. Plate (1995) for example introduces the HRR system as a version of a “convolutional memory”, contrasting it to other proposals for vector-based memory systems.

Because a vector in a VSA like the HRR system can be used to store multiple other vectors of the same type, these systems naturally lend themselves to applications involving hierarchical data structures. Thus, they obviously enable the kind of recursive computational derivation that a non-Classical combinatorial system requires. And because cues can be used to recover stored vectors from the memory, the computational derivations are reversible, as required, and may be useful in processing, as discussed in the previous three sections. But all of these properties are motivated simply by the requirements of a memory storage system for structured entities, and there are at least two ways in which such cue-addressable memories are not well suited to serve as the basis for a compositional system of mental representations.\footnote{One could argue that a parallel point applies to systems of symbolic logic, which are also framed purely formally or syntactically (though see the remarks in §6.2). However, logical systems are designed to capture truth functions. Even if only syntactically specified truth-values ‘T’, ‘F’ and the like are part of the system, the whole thing is designed to capture relations of consequence which are fundamentally semantic relations.}

First, as discussed in the previous two sections, these systems do not build in constraints on which sets of vectors can be combined into more complex structures sufficient to provide a notion of well-formed structure, syntactic, semantic or otherwise.\footnote{On this point, see (Fodor and Pylyshyn 1988, p.23).} Of course, as I’ve
suggested, some of these problems may be solved in practice by careful choices of vectors, independently principled constraints on allowable bindings, and so on (see Hinton 1981, p.208). But it seems as though the apparatus left unspecified here is (a) very important, and (b) perhaps importantly parasitic on Classical accounts of processing.

Second, and somewhat more technically: in order to keep the items stored in such systems distinct (thus recoverable), the vectors for any two items (including, as an important special case, two copies of the same entity bound to different roles in a larger structure) must be dissimilar, where vector similarity is generally measured in terms of the angle between two vectors in their common vector space (for example, via the cosine similarity metric, or the dot or “inner” product). As Gayler (2003, p.4) notes, “superposition is a similarity preserving operation, whereas binding is a similarity destroying operation”. Provided the binding process results in dissimilar vectors, the superposition process will create minimal noise when layering these vectors on one another.

This need for vector dissimilarity is serviced differently in various systems: in the HRR system, in which items can directly be bound to one another by the circular convolution operation, the atomic vectors themselves (those used for roles, variables, and atomic representations) are to be high-dimensional with each element drawn from a distribution (as discussed in §2.5), such that they will (strongly) tend not to be very similar to one another. In the tensor product system, the role vectors are (as discussed at length above) chosen so as to be at least linearly independent and in the limit orthogonal to one another, and in fact, in the

---

53 I will refer to this measure throughout, so I may as well define it: the cosine similarity between vectors $a$ and $b$ is the dot product of the vectors divided by the product of their Euclidean norms or magnitudes. More intuitively, it measures the difference in orientation between the two vectors, on a scale from -1 (opposite orientation) to 1 (same orientation).

54 Orthogonality is the best guarantee of dissimilarity: if two vectors are orthogonal, their dot product is 0. Provided all vectors in a set are nonzero, orthogonal vectors must be linearly independent.
system considered in Smolensky (1990), the filler vectors must have these properties as well to allow exact recovery.

The metaphor used earlier in §2.2, of storing long sentences in tiny boxes, is apt here. VSAs provide a way to throw many vectors at once into a (vector-shaped) box such that they can be retrieved later. They are ideal for the storage and recovery of discrete items. This may seem like an advantage in the present context, because linguistic expressions are such discrete structures. But it leads in some cases to technical problems (see Smolensky 1990, p.172), and it is generally very inefficient, as discussed below. Moreover it’s important to stress that we are not, at present, in search of a method via which linguistic structure can be represented in neurons, but rather, of a neurally-based explanation of the systematicity of thought. It may be that a fundamentally language-like system is the way to go (cf. Fodor’s famous “only game in town” argument), but we’ve already seen some ways in which VSAs can differ from Classical systems and retain at least many of their capacities, and there are more natural and much more powerful ways of encoding information in vector spaces.

As was briefly discussed in §1.3 and will be discussed further shortly, vector spaces potentially provide a very rich representational medium in which variations along any dimension of the space correspond to variations in what is represented. This cannot be the case in memories like those involved in the tensor product representation, for example. As Smolensky (1990, p.172) notes, if one wants to store \( n \) linearly independent patterns in his system, one needs \( n \) dimensions in the vector space (corresponding to \( n \) nodes in the relevant network). \[^{55}\]

\[^{55}\] This amounts to using vectors as analog representations in the sense defined in (Maley 2011), or iconic representations as defined in (Quilty-Dunn 2017)—itself a special case of representation by structural similarity (O’Brien and Opie 2004, Gladziejewski and Miłkowski 2017, Kiefer and Hohwy 2017).

\[^{56}\] Intuitively, this can be motivated as follows: a vector space, such as an infinite 2D plane, can be “spanned” by almost any two 2-dimensional vectors chosen at random, for example \([0 2]\) and \([-1 4]\). To say that these vectors span the space is just to say that any vector in the space can be computed as a linear combination of these vectors (that is, by scaling each vector by a constant and adding them). Such vectors provide a “basis” for the space, defining a coordinate system (the most common choice of such vectors, of course, being the “orthonormal” unit vectors \([1, 0]\) and \([0, 1]\)). The chosen vectors will only fail
By contrast, consider a simple representation in which the magnitude of each vector component corresponds to a represented magnitude—for example, a color solid in 3 dimensions used to represent a space of colors in terms of hue, saturation and luminosity. Here, every vector in the 3D space represents something different. Its capacity is limited only by the range of discriminable values each vector component can take. A tensor product representation operating on 3D vectors, by contrast, could store only 3 distinct “memories” without degradation. This is obviously an extremely inefficient use of the vector space (cf. Plate 1995, p.5), and in that respect not very biologically plausible.

A related problem that I view as a major one is that collections of vectors with the properties required by VSAs are, as Plate (1995, §VI.B) notes with respect to the HRR system, unlikely to be output by the processes that supply the inputs to cognitive systems, such as perception. One way of seeing this point is to suppose that perceptual representations, at least early ones, are iconic. In any case, sensory input vectors are certainly not random vectors, nor will the higher-level representations derived from them be, insofar as they systematically carry information about sensory stimuli (note for example that topographic mappings like retinotopy extend fairly deeply into cortical processing hierarchies).

I should admit that I have pointed up the contrast between these two approaches to representation somewhat artificially. In Plate’s system (Plate 1995), sequences constructed by binding atomic representations will tend to have vector representations similar to those of their constituents, and also similar to other sequences with constituents in common. This is to span the space in degenerate cases in which one is actually just a rescaling of the other (as in the pair [1 2], [2 4] that McLaughlin (2014) uses as an example, or, for that matter, in the pair [0 0], [0 0]). In this case, rescaling and adding the vectors yields only vectors lying on the same line as the chosen vectors.

Similar remarks apply to 3D spaces and beyond. In general, choosing \( n \) \( n \)-dimensional vectors at random, you have an extraordinarily good chance of spanning the \( n \)-D space. However, accordingly, choose one more \( n \)-dimensional vector and it is almost certain to be linearly dependent on the others already chosen. Moreover, the only way in which it would be independent would be if two of the others failed to be independent. Thus, if one wants \( n \) linearly independent vectors, as Smolensky says, one needs \( n \)-dimensional vectors.
important, because part of the motivation behind the HRR system was to provide an implementation for Hinton’s idea of “reduced descriptions” (Hinton 1990)—representations that, like thumbnail images, are both descriptive (at a “low resolution”) of a certain entity and serve as pointers to a fuller description. Content-similarity among similar HRRs could facilitate the first function (see Plate, p.3). However, the items being stored, the “atomic” representations, are themselves arbitrarily related to one another (apart from the requirement that they be, simply, distinct), hence the problem of interface with perceptual systems remains.

As also noted in (Plate 1995, §VI.B), this problem can be overcome via a module that maps vectors derived from sensory processing to the randomly generated vectors suitable for use in an HRR memory system. This technique is used in recent work by Chris Eliasmith and colleagues (see e.g. Blouw and Eliasmith 2015 and Eliasmith et al 2012) that successfully applies the HRR machinery (i.e. circular convolution) to a very wide range of tasks in cognitive modeling. The concept of a “semantic pointer” developed in this work is very similar to that of the compressed vector representations posited in contemporary connectionist systems.

It may be that such a hybrid architecture, with empirically learned compression functions employed in early sensory processing (as in standard connectionist approaches to computer vision and generative modeling), followed by a mapping of compressed perceptual representations to a distinct “conceptual” representation employing a fixed composition function on random vectors, is the way to go. But any such solution will still involve mapping to a set of

---

57 Dretske’s (1981) view that cognition involves extracting, from the “analog”, information-rich sensory signal, representations that carry specific information only about discrete categories (objects, relations, etc), may seem to support this type of model. But this issue of extracting “digital” (in Dretske’s somewhat unusual sense) information from analog information is orthogonal to the one under consideration. It could be, for example, (a) that high-dimensional vectors low in a sensory processing hierarchy represent by means of analogy or iconicity (for example, an array of photoreceptors might represent an array of light intensities in a visual scene in a way that relies on isomorphism), (b) that higher-level representations in the system, derived from these (perhaps via intervening representational layers) are selectively responsive to the visual presence of dogs, and yet (c) that the vector space representations of the set of (visually discriminable) dogs themselves function iconically, with each dimension of variation corresponding to some feature of the dog being represented. This would still amount to “digital”
stored discrete items in memory, sacrificing the potential informational richness of a
representation in which each dimension of variation may be significant—a feature that seems
clearly useful for purely conceptual as well as perceptually based representations, as discussed
at length later (cf. for example §§3.3 and 5.6). \(^5\) Indeed, recent empirical work (Constantinescu
et al 2016; see also Kriegeskorte and Storrs 2016) suggests a common representational format
(namely, grid cells) for both physical space and abstract conceptual “spaces”.

Over the course of the next three chapters, I attempt to explain why I am making a
different bet: that end-to-end empirical learning is capable of producing both an effective
composition function and good representations. For what it’s worth, this is the approach taken
by many contemporary connectionists, including those behind Google’s machine translation
system, for example (see §5.4). It could be argued that such connectionist systems, though
cutting-edge, are far more task-specific than the general-purpose brain model pursued in the
work of Eliasmith and colleagues, and that a VSA architecture shines most in the latter context,
where the sequential recovery and interpretation of stored symbolic representations is
paramount. This may be so, but it is one of the burdens of the next several chapters to argue
that the combinatorial representational structure of VSAs can be combined with dense vector
space representations to yield a simpler, more fluid, and more thoroughly connectionist type of
cognitive architecture. Dropping restrictions on the input vectors requires giving up the
theoretical guarantee of reversibility of the composition function, but as we’ll see this is not a
problem in practice, so long as a system is able to learn to invert its composition function (§4.1).

---

representation in Dretske’s terms just in case the higher-level representations covary only with
dog-features, not with lower-level features that may fluctuate while the former are held constant.
\(^5\) While vectors with real-valued components are useful for their informational richness and are often
used in computational modeling of psychological processes, this point does not reduce to anything about
real values or continuity. As Quilty-Dunn (2017) argues, what is essential to “iconic” representations is a
certain sort of covariation relation between dimensions of variation in the representational vehicle and
those in the represented medium. These variations may be either continuous or involve discrete steps
(see Maley 2011 for similar remarks about analog representation).
To summarize the results of this chapter and to anticipate a bit: a system that shares with VSAs the property that novel representations can be computed (potentially recursively) on the basis of existing ones, while also allowing that each vector may be part of a dense, semantically meaningful vector space, would be desirable. We’d also like the composition functions in question to ensure that appropriate semantic and syntactic constraints on the construction of representations, and the interactions between them (e.g. inferential transitions), are enforced. And let’s throw in that we’d like to preserve the feature of the HRR system that input and output vectors are in the same space (and thus have the same dimensionality), since, among other things, this seems essential for a reasonable vector-based approach to recursion.59

59 The “other things” include the fact that the resulting compression of information as elements are recursively combined is desirable on independent grounds (see §4.1 below).
Chapter 3: Neural Vector Space Models

3.1 Distributional semantics

Vector Symbolic Architectures, as we’ve seen, are concerned particularly with the derivation of semantically complex representations from primitives. In this chapter, I discuss how “atomic” representations corresponding in content to words or concepts\(^60\) can be derived from linguistic data in a way that results in dense vector space representations where similar vectors are similar in meaning. I return to the issue of deriving complex (phrase- and sentence-level) representations at the close of this chapter, and explore it further in the next.

The approach to representation in question begins with a research program in semantics, called “distributional semantics” (Harris 1954, Harris 1968, Miller and Charles 1991), that has received comparatively little attention in philosophy, but has long been a cornerstone of statistical approaches to computational linguistics, and in particular of quantitative theories in semantics. The central hypothesis of this research program is that “difference of meaning correlates with difference of distribution” (Harris 1954, p.156). There are, of course, many ways in which this basic idea can be elaborated. I will dive in with a quote to use as a reference point (Coecke et al 2010, p.3, where for continuity I have used single quotations for mentioned expressions in place of their italics):

The basic idea is that the meaning of a word can be determined by the words which appear in its contexts, where context can be a simple \(n\)-word window, or the argument slots of grammatical relations, such as the direct object of the verb ‘eat’. Intuitively, ‘cat’ and ‘dog’ have similar meanings (in some sense) because

\(^{60}\) Following David Rosenthal (in conversation), one may call mental representations “word-sized” that have whatever sorts of contents individual words in natural languages have, without intending to commit oneself to the existence of concepts, in Fodor’s sense of sub-sentential Mentalese syntactic constituents. These sorts of contents will turn out to be “conceptual” on some uses of that term, including the somewhat liberal usage introduced in §§5.1—5.2.
cats and dogs sleep, run, walk; cats and dogs can be bought, cleaned, stroked; cats and dogs can be small, big, furry. This intuition is reflected in text because ‘cat’ and ‘dog’ appear as the subject of ‘sleep’, ‘run’, ‘walk’; as the direct object of ‘bought’, ‘cleaned’, ‘stroked’; and as the modifiee of ‘small’, ‘big’, ‘furry’.

This quote is in fact about a more specific proposal derived from distributional semantics, “vector space models of meaning”, which I’ll come to in the next section, but it gets across the basic idea: that the meaning of a word is at least largely a function of its linguistic context.

Several writers on the subject (such as Harris 1954) stress that they conceive of facts about meaning as distinct in principle from facts about context in this narrow sense, and that a full specification of meaning involves taking a wider range of facts into consideration, including those about the extralinguistic contexts in which an expression occurs (see Miller and Charles 1991, p.4). Thus, “determine” in the above quote should probably be understood epistemically. But it is possible to imagine a distributional semantics that takes “determine” metaphysically: on that view, the meaning of an expression would be constituted by its contexts of use.

Sometimes, proponents of distributional approaches to semantics seem to have this kind of view in mind. Many researchers, for example, take the distributional hypothesis to be a form of the Wittgensteinean notion that meaning is use (Wittgenstein 1958, Horwich 1998). This kind of view only stands a chance of working if we consider the contextual facts in question to be the infinite range of possible linguistic contexts (Clarke 2012), however. Moreover there seems no reason to exclude extralinguistic contexts from consideration as determinants (in this metaphysical sense) of meaning.

---

61 Gary Ostertag has pointed out to me that grounds may be lacking for attribution to Wittgenstein of anything stronger than the epistemic view of the relation of meaning to use. Passages such as the following do, however, at least in translation, suggest (a qualified form of) the constitutive view: “For a large class of cases—though not for all—in which we employ the word ‘meaning’ it can be defined thus: the meaning of a word is its use in the language” (Wittgenstein 1958, §43).
At least as Coecke et al conceive of it, the similarity between the meanings of ‘cat’ and 'dog' seems to amount to a set of similarities between cats and dogs themselves (both are pets, furry, and so on). And of course, these similarities are reflected in linguistic corpora. This suggests that we might conceive of the overall way in which linguistic representation works, guided by the distributional hypothesis, as follows: the structure of the relations among objects of discourse (objects, properties, states of affairs, events, etc) in the world described is mirrored in the distributional facts about language. On this view, linguistic representation is a case of representation by second-order structural resemblance (Cummins 1991), (O’Brien and Opie 2004), (Gładziejewski and Miłkowski 2017), (Kiefer and Hohwy 2017). Indeed, (Harris 1968) argues that language as a representational system carries meaning only in virtue of the structural relations among its elements.62

Unless otherwise indicated, I will henceforth use the term “distributional semantics” to refer specifically to the idea that the meaning of a linguistic expression can be determined (i.e. discovered) by examining its linguistic context, leaving the metaphysics of representation to one side. But how is “linguistic context” to be understood? Coecke et al, in the passage just quoted, suggest several possibilities: we may appeal specifically to syntactically defined relations among words, to words that tend statistically to occur within a certain distance of the target word, etc. One may also appeal to the sorts of broad discourse contexts in which expressions occur: expressions that tend to occur in similar discourses are (perhaps) similar in meaning. Harris (1954, p.146) stipulates that “The distribution of an element will be understood as the sum of all its environments”, where the environment, for a word, morpheme or other

62 The elements themselves must be arbitrary and intrinsically meaningless, Harris argues, because they could not be discrete and repeatable as required if based on “natural” relations between sign and meaning that might differ across speakers. The following quote is indicative of Harris’s thinking on this topic: “A natural representational relation between word sounds and word meanings would be useful only if it could be varied, guessed at, and in general if it eliminated the need for the language users to have to learn a fixed and jointly accepted—i.e., preset—stock of elements. But the fixed stock of elements is precisely what is required for transmissibility without error compounding” (Harris 1968, p.8).
subsentential element, is the set of other expressions of its class (again, morpheme, word, etc) with which it co-occurs.

I propose to take “distribution” broadly: on my usage, it simply refers to the sum total of statistical properties of the language under consideration, within the available corpora: the marginal probabilities of characters, words, \( n \)-grams, sentences, and discourses, and their probabilities conditional on one another.\(^63\) These distributional facts are of course reflective of syntactic, as well as semantic, relations among expressions. Focusing on the distributional relations as sources of evidence for syntactic and semantic relations rather than on the syntactic and semantic relations themselves might seem perverse from the point of view of pure semantic theory, since the distributional facts are plausibly determined by syntax and semantics. But it makes perfect sense if the goal is to understand the learnability of languages from one’s ambient linguistic environment, or the representations derived from such learning.

Moreover, the distributional facts are not really on a par with other potential sources of evidence about meaning. Linguistic expressions are much more intimately related to their intralingual environments than they are to other environments. It is shown in (Harris 1968) that morpheme boundaries correspond to points in a phoneme sequence at which the number of possible subsequent phonemes increases, after a steady decrease since the last such inflection point (e.g. given \( ba- \), there are fewer possible completions at the next step than there would be given \( baseball- \)). This method for segmenting utterances into morphemes relies on only a tiny sliver of the available distributional information: it depends on how many phonemes have a nonzero probability of occurring after others, regardless of the precise probabilities. Given this

\(^{63}\) Obviously, some of these statistics will be more readily available, and useful for framing generalizations, than others. Statistical analysis is difficult to apply directly to sentences or even longer segments of discourse, simply because the requisite statistics for sentences will be degenerate (e.g. most grammatically possible sentences, even semantically acceptable ones, have never been uttered). In practice, conditional word and \( n \)-gram probabilities have been most exploited in computational models based on these ideas, for obvious reasons. But for a model that exploits sentence-level statistics, see (Kiros et al 2015), and for a powerful character-based model, see (Kalchbrenner et al unpublished).
segmentation, syntactic categories can be identified by appeal to the reductions in redundancy one obtains by representing sequences of the language (i.e. utterances, discourses) in terms of derived classes (words, word sequences, and sentences). In this way, one can define all meaningful elements of a language, down to the individually meaningless phonemes.

One important, though perhaps obvious, point, is that since linguistic contexts vary across many dimensions, distributional theories of meaning are best paired with a graded notion of similarity of meaning rather than an absolute one of synonymy or non-synonymy (Miller and Charles 1991, pp.2-3). In philosophical semantics, graded conceptions of meaning have been associated with holism (itself unfairly associated, perhaps, with relativism and imprecise thinking), and have taken a back seat to propositional models. Within the distributionally based computational models to be discussed below, it is possible to precisely quantify the number of dimensions along which meaning varies, and how close two expressions are in meaning, both in total and along specific dimensions, as captured by a given model.

As the variety of computational models based on this sort of idea illustrates, there are many specific approaches within the framework of distributional semantics. My focus in this section has been on understanding the conceptual foundations of the approach, rather than on such details. But keeping in mind that meaning varies across many dimensions, a surprising amount of the varied facts about linguistic meaning can be captured by this hypothesis.

For example, it may be argued that the meanings of “night” and “day” are in a sense opposite, but that distributional semantics would treat them as similar, since they are likely to occur in many of the same contexts. However, the meanings of “night” and “day” are similar in important respects (both refer to important phases of short time intervals on (typically) Earth), and the sense in which they are opposites would be precisely captured by the distributional
facts as well (“day” would tend to occur along with such words as “picnic”, “sunny”, “beach”, while “night” would occur along with “darkness”, “stars”, “moon”, etc).

3.2 Vector space models in semantics

Especially in recent years, a family of computational models based on the distributional hypothesis, “Vector Space Models” (VSMs), has gained traction within computational linguistics.\(^{64}\) VSMs, broadly construed, form the core of many cutting-edge computational approaches to natural language processing (henceforth, intermittently, “NLP”), in particular connectionist approaches, as I’ll discuss below. The basic idea behind a vector space semantic model is simple: it is just the idea that a high-dimensional vector space can be used to capture the meanings of linguistic expressions. In a vector space model, each expression is represented by a vector, and semantic relations between expressions are captured by relations of proximity in the vector space. Expressions similar in meaning will correspond to points nearby in the vector space (cf. §1.3). There are at least two basic questions one might ask about such models: (1) what are they good for? and (2) how might such a vector space be constructed? I’ll begin with (2) in this section. Applications will be discussed in the remainder of this chapter, and are a central topic of discussion in subsequent chapters as well.

Consider the case of word meaning for concreteness, and because VSMs have most often been applied to this case in practice. Perhaps the most direct way to construct a VSM for words is to set up an $N \times N$ table (or matrix), where $N$ is the size of the vocabulary under consideration, and in which each word occurs once along each axis. The entries in this matrix

---

\(^{64}\) For a useful introduction to this topic to which much of my discussion in this section is indebted, see (Jurafsky and Martin 2009), Chapter 19—pagination used here refers to a draft of the chapter freely available online. See also the survey in (Erk 2012).
are counts of word co-occurrence, relative to some specified context window (ranging from a small \( n \)-word context window surrounding occurrences of the word in question to entire documents—see (Jurafsky and Martin 2009, §19.1.2). In such a matrix, each word is represented by the list (vector) of co-occurrence counts with all other words in the vocabulary. For instance, we might take the rows of the matrix to represent words \( i \) (from 1 to \( N \)), where each entry in a row specifies how often word \( j \) occurs in the relevant context of \( i \).\(^{65}\)

This is clearly a distributional approach to semantics: the vector space is constructed directly on the basis of co-occurrence statistics (the raw counts could easily be normalized by the total number of occurrences of word \( i \) to yield probabilities \( p(j|i) \)). Note some interesting properties of this kind of model. First, it ensures straightforwardly that words that occur in similar contexts (i.e. near the same words, or in the same documents or discourses) have similar vector representations and thus lie near one another in the vector space. Assuming that the distributional hypothesis in semantics is correct, this means that representations for words with similar meanings will lie near one another in the vector space (I discuss the confirmation of this hypothesis, in the case of slightly more sophisticated distributional approaches, below).

Second, in this particular model each word is used, in effect, as a basis vector for the space (there are \( N \) dimensions, one for each word in the vocabulary). However, each word is also represented in the space. It is important to keep in view that the representation of a word’s meaning (or the representation that captures its meaning) is its co-occurrence vector, not the dimension in the vector space corresponding to that word. Indeed, depending on how context is defined, the component of a word’s vector that corresponds to that word itself may be arbitrarily small. For example, if only the words immediately to the left and right of a given word are used

\(^{65}\) The ordering of words in such a matrix is, in general, arbitrary with respect to the semantics.
as the context, then apart from exceptional cases like “very very”, words never appear in their own contexts. The probability that they will increases with the size of the context window.\footnote{It is possible to define context differently, so that each context window includes the word it is centered on. In this case, \( p(j|i) \), construed as the probability of \( j \) occurring somewhere in \( i \)'s context, would be 1 when \( i = j \). But it’s not clear that this is actually a desirable property of a semantic vector space. The fact that the meaning of ‘cat’ is maximally similar to the meaning of ‘cat’ (e.g. that its meaning is self-identical) is already captured in the fact that the cosine similarity between the ‘cat’ vector and itself is 1. There is no need to try to force the representation for ‘cat’ to have a large component along the dimension of the space corresponding to ‘cat’. Moreover, for wider contexts, words may well appear more than once, and we’d want those co-occurrence statistics to be treated on a par with the statistics for distinct words. It is best to think of the dimension in the vector space that corresponds to each word \( w \) (as opposed to the vector representation for \( w \)) as simply providing representational space for keeping track of how often \( w \) occurs in the context of other words (including tokens of \( w \) itself).}

This method provides a compellingly picturesque example of the structural representation approach to natural language, as is suggested by the name of one of the earlier proposals in this vein, the "Hyperspace Analogue to Language (HAL)". There are, however, several downsides to such a representation (see Pennington et al 2014, as well as Jurafsky and Martin 2009). First, raw word counts skew any measure of vector similarity in a way that places undue emphasis on words that commonly occur with very many others and so are not very informative about specific meanings, such as ‘the’ and ‘of’.\footnote{This might not always be an undesirable property. Sometimes, coarse-grained categorizations that count many words as relevantly similar might be desired (for example, syntactic categories are arguably abstractions over finer-grained semantic categories). But for the purposes of deriving a vector space in which fine-grained differences of meaning are easily accessible, this is not ideal.} Second, for large context windows (such as an entire corpus), co-occurrence counts might span “8 or 9 orders of magnitude” (Pennington et al 2014, p.1533), so that the representation of many differences between words will be dwarfed by the overall magnitudes represented in the vector space. In addition, again depending on the context window, the vector representations for words thus obtained may be very sparse (0 for most entries). While sparse representations are useful in some ways (see Olshausen and Field 1997), they are difficult to work with in others. Finally, even if various tricks are used to reduce vocabulary size (such as excluding very rare words), many natural

\footnote{See (Lund and Burgess 1996).}
languages (such as English and French) have on the order of hundreds of thousands of words. Even the tens of thousands of words needed to represent the vocabulary of individual competent speakers yields very large vectors that are difficult to work with in practice.  

Some of these problems can be overcome by using more sophisticated statistical properties of words than raw co-occurrence counts to define a vector space. For example, the (positive) pointwise mutual information (ppmi) provides a good measure of the degree of statistical dependence between two words. Because the ppmi is defined in terms of the contrast between the actual joint probability of a pair of words and the probability that they would co-occur by chance (i.e. their joint probability were their probabilities to be independent), the ppmi is a measure of “how much more often than chance the two words co-occur” (Jurafsky and Martin 2009, p.6). This scheme introduces its own problems, however, since measures of information are (of necessity, given Shannon’s definitions—see Shannon 1948) biased toward favoring low-probability events (see Jurafsky and Martin 2009, p.7).

Starting either with raw word counts or with some principled transformation of the base statistics, the sparsity and high-dimensionality problems can be solved by using some method for dimensionality reduction, such as Singular Value Decomposition (SVD)—see (Jurafsky and Martin 2009, §19.5). This involves mapping the original vector space to a different space of lower dimensionality while preserving as much information about the original space as possible. There are many techniques for dimensionality reduction that are employed widely in statistical analysis, but their common essence is to attempt to capture as much variability in a data distribution as possible using few dimensions. Thus, the final vector space derived via such

---

69 This problem is potentially less severe for languages like Japanese, in which a few thousand characters are combined to form most “words”. But even in this case, there is a phonetic alphabet that may bring arbitrary novel words into the vocabulary.

70 In fact, the well-known technique of Latent Semantic Analysis (LSA) is a paradigm example of this kind of VSM, in which SVD is applied to term-document counts. See (Deerwester et al 1990).
methods will be a smaller, compressed representation of the original vector space. Because of the compression, the sparsity of the original space will be reduced.

Despite the fact that individual axes in the resultant model will no longer correspond to individual words in the vocabulary, and that the dimensions of the vector representation of a word no longer correspond to individual co-occurrence statistics, the distributional hypothesis will still apply to the resulting vector space models if it applied to the input models, provided the compression is not too “lossy”. Reasonable compression schemes will not arbitrarily distort or scramble the relationships in the original vector space, but will rather (at worst) sample them at relatively low resolution. If word-word matrices can be seen as iconic representations of word meanings (via their distributional traces), compressed vector spaces derived by such techniques as SVD may be seen as “thumbnail images”.

3.3 Neural language models and vector space embeddings

Dimensionality reduction (as just discussed) involves a systematic mapping from vectors in one vector space to those in a different, lower-dimensional space. As mentioned in §2.6, mapping from one vector space to another is also of the essence of computation in connectionist models, whose representations are often described in terms of vector spaces (cf. §1.3): typically, one vector, representing the activities of neuron-like units in a given layer of a network, is mapped to a vector representing activities in a different layer, via a matrix that represents a collection of “synaptic” weights between interconnected units in the layers.

Moreover, dimensionality reduction has been proposed (Hinton and Salakhutdinov 2006) as a generally useful representational principle in connectionist models. So VSMs and artificial neural network models are a natural fit for one another.
Further, because the data used to construct VSMs is just a distribution over linguistic expressions, in principle a neural network that attempts to model this distribution, i.e. a generative model (Hinton et al 1995, Hinton 2005, Goodfellow et al 2014), should, via a training procedure sensitive just to the statistics of the input corpus, be able to capture the statistical correlations that follow from the distribution in question. Learning the meaning of words can then be cast as simply a special case of inducing a generative model from data (Hinton 2005, Hohwy 2013, Clark 2013). This has obvious implications for debates over special-purpose "language modules" and related innateness hypotheses, some of which I'll discuss later on.

There are many different neural network models of this type. What they have in common is that, rather than directly representing word co-occurrence probabilities, they construct a statistical model (the neural network) that captures these statistics via its learned parameters (and latent variables)—a "neural language model" (Bengio et al 2003). An important class of such models, which I'll discuss first, achieves this by learning to predict some of the words in a given context ("target" words) given the others. I begin with the rather transparent example of (Mikolov et al 2013a), which uses a recurrent neural network (RNN) to predict the next word in a sentence, given the previous words.

Mikolov et al’s model (see Fig. 4 below) begins with the kind of word representations that VSMs are designed to improve upon: “one-hot” encodings in which each word is represented by an N-dimensional vector in which the n entry is a ‘1’ and the rest of the entries are ‘0’s, where n is the index of the word in the vocabulary and N is, again, the vocabulary size. These

---

71 Jakob Hohwy and I have discussed generative models and their relation to unsupervised learning at length elsewhere (see Kiefer and Hohwy 2017, in print). Here, I largely presuppose these concepts without explanation, but see fn.13 above.

72 This is an interesting case of generative modeling, since the process being modeled is in effect the language-production faculty of other speakers. One might imagine on this basis how simple sign-based communication systems (akin to the bee dances and tail slaps that have intrigued many a philosopher—see Millikan 1984) might have evolved by bootstrapping into more sophisticated systems like human languages.
vectors are used as the inputs to the model. As in standard neural networks, the input vectors are transformed by a weight matrix, here \( U \), yielding an output “hidden layer” representation, \( s \). The recurrent connections are given in the recurrent weight matrix \( W \).

Since this is a recurrent network, the state of the hidden layer at any moment is a function of both its state at the previous moment and its current input. Here, the particular function employed is the logistic function \( f(x) = 1/(1+e^{-x}) \) applied to the sum of the weighted output from the previous hidden state and the current input. So, for input vector \( v \) and time index \( t \), \( s_t = f(Uv + Ws_{t-1}) \). From the hidden state \( s_t \), the predicted next word in the sequence \( y_t \) is computed, similarly, by a nonlinear function \( g \) applied to a transformation \( V \) of \( s_t \). Here, \( g \) is the “softmax” function, which is similar to the logistic in that it “squashes” its input to output values between 0 and 1, but in this case the output is vector-valued, and represents an entire distribution over possible vocabulary items.

---

73 For clarity, I generally defer to authors’ notations when discussing specific model parameters.  
74 The logistic function (sometimes called the “sigmoid” function for its S-shape) is a bread-and-butter nonlinear “neural” activation function, often used to represent probabilities. It takes arbitrarily large inputs and “squashes” them into the interval between 0 and 1 (with input 0 mapped to 0.5).
at time $t$, $w(t)$, is used to select a row of weight matrix $U$, which is combined with information from the previous time-step to predict the next word, $y(t)$. Once trained with backpropagation, the rows of $U$ act as word-level representations whose relations in the vector space capture semantic and syntactic regularities in the corpus. See text for further details.

The model is trained via backpropagation of error derivatives (Rumelhart et al 1985) to maximize the log probability of generating the correct next word, given the previous ones. Note that although the backpropagation algorithm is typically associated with supervised learning schemes, here the learning is strictly unsupervised, in the sense that it depends only on the data being modeled (the linguistic corpus), not on an independent “teacher” signal. The spirit of this kind of proposal, and indeed of the perspective I push in this chapter overall, is well captured by this quote (Mikolov et al 2013a, p.747): “The model itself has no knowledge of syntax or morphology or semantics. Remarkably, training such a purely lexical model to maximize likelihood will induce word representations with striking syntactic and semantic properties”.

The word representations in question are the rows of the weight matrix $U$. In fact, for the purposes of VSMs, the rest of the language model is irrelevant. The vectors in $U$ are in effect versions of the compressed word co-occurrence vectors discussed in the previous section: they are shaped by training on the word prediction task to capture co-occurrence statistics, and their dimensionality is determined by the dimensionality of the hidden layer $s$, potentially much smaller than the vocabulary size (the dimensions of $s$ can be as chosen, though of course a larger hidden layer corresponds to a higher-capacity model).

In (Mikolov et al 2013a) and elsewhere, it was demonstrated that such learned vector space “embeddings” for words encode syntactic and semantic information in a surprisingly transparent way that vindicates the distributional hypothesis in semantics. It is worth spending a bit of time on these rich results. First, Mikolov et al developed a series of tests for the encoding
of syntactic and semantic information based on analogies: they used their model (i.e. the word representations in $U$) to complete statements of the form “$a$ is to $b$ as $c$ is to ____”.

The syntactic analogies tested for relationships such as “base/comparative/superlative forms of adjectives; singular/plural forms of common nouns; possessive/non-possessive forms of common nouns; and base, past and 3rd person present tense forms of verbs” (Milokov et al 2013a, p.747). Semantic analogies tested for semantic relationships such as the one of class membership that obtains between “shirt” and “clothing”, by way of analogies that depend on such relationships, for example “‘Shirt’ is to ‘clothing’ as ‘table’ is to ‘furniture’”. Here, the task was to rank word pairs according to the degree to which they exhibit the same relationship as “gold standard” pairs, by way of considering the validity of the relevant analogy.

Results on these tests were very good compared to chance and compared to alternative statistical methods in the same ballpark, such as LSA. But the method used to achieve the results is of most direct interest: Milokov et al showed that simple arithmetic operations on relevant vectors yielded excellent results, proving that regularities across individual lexical items could be captured in terms of a constant offset that relates the vector representation of one of the words in a pair to the other. Similar remarks hold true of semantic relations. Different offsets correspond to different relationships, and in a high-dimensional vector space, all of these relations can be captured at once in the same representation (Milokov et al 2013a, p.749).

For example, to solve a syntactic analogy such as $small : smaller :: big : ____$, Milokov et al employed the following simple equation: $y = smaller - small + big$, where $y$ is the vector corresponding to the word predicted for the completion. They found that to a surprising degree given the syntax-agnostic training of the model, the following result held: the vector computed in this way was closest, in terms of cosine similarity, to the vector for the correct

---

75 Here, as elsewhere, I use bold font to indicate the vector space embedding of the corresponding word.
answer (e.g. in this example, \( y \approx \text{bigger} \)), compared to other items in the vocabulary. Similar relationships held for other syntactic relations, such as the singular \( \rightarrow \) plural transformation. As was subsequently widely discussed, this type of simple “offset” equation also worked well for semantic relationships. Somewhat famously by now, \( \text{king} - \text{man} + \text{woman} \approx \text{queen} \).

Similar equations have in fact been shown to work in the case of vectors encoding \textit{images} in generative models. For example, in (Radford et al unpublished), averaging the hidden-layer vector codes for three distinct images of men wearing sunglasses, subtracting the average for three men without sunglasses, and adding the average for three women produced an image of a woman wearing sunglasses (e.g. \( E(\text{image of man wearing sunglasses}) - E(\text{image of man}) + E(\text{image of woman}) = \text{image of woman wearing sunglasses} \), where \( E \) denotes expected value). This result is replicated in slightly higher quality and using a different unsupervised learning method in (Bojanowski et al 2018). I mention this here because it is clearly important within the overall dialectic around connectionism and compositionality, and interesting in its own right, but it is tangential at present—see §5.6 for further discussion.

In subsequent work, Milokov and collaborators developed more refined and streamlined approaches to VSMs along the lines discussed in (Milokov et al 2013a). In (Milokov et al 2013b), two now-famous window-based methods for learning vector space embeddings for words were proposed: the “skip-gram” and Contextual Bag of Words (CBOW) models (jointly known as \texttt{word2vec}, the name of the open-source implementation released by the authors). The first model takes as its training objective the prediction of the center word in a window based on the \( n \) words immediately surrounding it, and the second model proposes the inverse: predicting the \( n \) surrounding words given the center word. Both models can be implemented as minor variations of one another.
The virtue of this approach, which has proven extremely popular, is that it obtains embeddings similar to those discussed in (Milokov 2013a) without requiring the training of a full language model, most of which is irrelevant to the embeddings. A drawback of these models is that, like the one in (Milokov et al 2013a), they require the computation of the softmax function, which, due to the need to calculate a normalization term that depends on all the words in the vocabulary, is prohibitively computationally expensive for large vocabularies.

In (Milokov et al 2013c), principled workarounds are proposed that achieve similar results at much lower computational cost. For example, the cross-entropy cost function that depends on the entire output distribution can be replaced by “negative sampling”, which adjusts the word vectors so that they are better at predicting target words and less likely to predict a small range of incorrect targets chosen stochastically for each training example. They also achieve better results by learning new vectors for idiomatic phrases.

Before moving on, it is instructive to focus for a moment on exactly how the word vectors are learned in this model. Consider first that the inner or “dot” product of two vectors (the sum of the products of each of their components) is a somewhat noisy measure of their similarity: each term-wise product is positive if both its inputs are positive or if both are negative, and negative if they are of different signs (i.e. oriented differently along that dimension), and the inner product is an un-normalized average of these terms.

The softmax function converts the inner product \( \mathbf{w} \cdot \mathbf{c} \) of the vectors \( \mathbf{w} \) and \( \mathbf{c} \) (representing the word of interest and the target context word, respectively) into a probability \( p(\mathbf{c}|\mathbf{w}) \), applying the exponential function (e.g. \( e^{\mathbf{w} \cdot \mathbf{c}} \)) and dividing by a normalization factor.

---

76 I want to take care not to gloss over this somewhat idiosyncratic usage of “learned”. In the context of psychology, it would be odd to say that a thought, belief or other propositional attitude has been “learned”, as opposed to a fact, or a skill. Perhaps one way of making sense of the extended usage in machine learning is to think of learning a representation as learning how to represent some aspect of the world.

77 Hence noisy, since it combines information about vector angle with information about magnitude. The cosine similarity metric discussed above fixes this by dividing the dot product by the vector norms.
(which is just the sum over similar terms \( e^w \cdot w' \) for all other word vectors \( w' \)). As the context window slides across the corpus, the probability output by the network is compared with the “ground truth” of the actual context word to derive gradients for changing the word vectors, such that at each iteration \( p(c|w) \) comes to more closely reflect the corpus statistics. The relation between the probability \( \text{softmax}(c \cdot w) \) and \( c \cdot w \) itself is monotonic (in fact, strictly increasing). Thus, training forces the vectors \( c \) and \( w \) to be similar in proportion to the number of times the corresponding word \( c \) occurs in the context of word \( w \) in the corpus.

### 3.4 GloVe vectors

I examine two more VSMs in this chapter, one in this subsection and one in the next. First, consider the “GloVe” (Global Vectors) model proposed in (Pennington et al 2014). This model was initially motivated by the observation that ratios of the conditional probability relations between words are much more likely to make interesting semantic relations explicit than are the probabilities themselves. For example, as shown in Table 1 of the paper under discussion, \( p(\text{solid}|\text{ice}) \) and \( p(\text{solid}|\text{steam}) \) in a typical corpus are both very small, though they differ by an order of magnitude (a factor of \( 10^{-4} \) vs \( 10^{-5} \)), while the ratio \( p(\text{solid}|\text{ice})/p(\text{solid}|\text{steam}) \) is 8.9. The corresponding probabilities for ‘gas’ given ‘ice’ and ‘steam’ are similarly small (differing by an order of magnitude in the opposite direction), while the ratio analogous to that just mentioned is also small (on the order of \( 10^{-2} \)). By contrast, the corresponding probabilities involving words irrelevant to the difference between ‘ice’ and ‘steam’, such as ‘water’ (which is strongly but equally relevant to both) and ‘fashion’ (relevant to neither) are likewise tiny, on the order of \( 10^{-3} \) for the relevant words and \( 10^{-5} \) for the irrelevant ones, while the ratios are both close to 1.
Thus, the ratio captures the semantic relevance of words to other words nicely: words that are strongly associated with the top word in the ratio but not the bottom one will have ratio values well above 1, words strongly associated only with the bottom word will have ratio values much less than 1, and words that fail to discriminate between the two words in the ratio will have ratio values close to 1. From this, the authors conclude that in general, an appropriate model based on co-occurrence statistics should be set up so that some function $F$ of the word vectors learned is equal to the relevant ratio: $F(w_i, w_j, w_k) = p(k|i)/p(k|j)$. $F$ will be a model containing parameters learned so as to match the observed probability ratios in the corpus.

From here, the authors derive a regression model similar to the Milokov et al models in that it relies on the dot product between two word vectors, but different in that it also includes a term that keeps track of the number of times in the entire training corpus that word $j$ occurs in the context of word $i$ (hence, the “Global” in “GloVe”). They also compare their model to proposals, such as that of (Milokov et al 2013b) just discussed, that ignore global co-occurrence counts and instead rely for optimization only on a sliding context window. They show analytically that a series of independently motivated modifications to the latter models (based mostly on considerations about optimization) yields their own model (with a weighting function that is a free parameter, versus the implicit weighting function in Milokov’s models).

There are a priori arguments in favor of both “moving context window” approaches like skip-gram and the GloVe approach. The former is more realistic as (part of) a model of human learning, since it involves keeping track of local co-occurrence statistics as a text is scanned (it

---

78 Here, the vector for word $k$, $w_k$, receives a different notation because a separate set of “context” vectors is learned for words playing this role as opposed to the words that occur in the ratios. A similar splitting of the vector representations into “input” and “output” vectors figures in the skip-gram and CBOW models discussed above. I do not dwell on this feature of these models here because it is included so as to ease optimization, without affecting the conceptual issues involved. Typically, when the vectors are used for semantic tasks or tested for similarity and the like, the vectors for the same word in various roles are averaged, or summed (see e.g. Pennington et al, p.1539). See (Levy et al 2015) for a more sophisticated interpretation of what such summation amounts to, that invokes the distributional semantics hypothesis in the form advocated by Zellig Harris (1954), discussed above.
employs a version of online stochastic gradient descent). However, the latter model exploits information about how many times, within a broader context, a word appears in the local context of another word, information that is not explicitly modeled by the former approach. This information is available to the skip-gram model if trained on the full corpus, but only implicitly over an entire epoch of training, so it is possible that GloVe finds a better solution.

A casual examination of the cosine similarity of various pre-trained GloVe vectors (made available freely online by Pennington et al and the Stanford Natural Language Processing group) verifies that the vector space embeddings learned by the GloVe model indeed capture important syntactic and semantic relations. With respect to syntax, testing on the equation $y = \text{smaller} - \text{small} + \text{big}$ from (Milokov et al 2013a) produces the expected result: the vector $y$ is more similar (by the cosine similarity metric) to $\text{bigger}$ than to any other vector I tested by a significant margin, stands in relations to other vectors that are similar to those in which $\text{bigger}$ stands, and is far more similar (by an order of magnitude) to $\text{bigger}$ than to all the other vectors in the vocabulary on average. Similar results obtain for other sets of syntactically related terms.

With respect to semantics, nearly every word pair that I tested for cosine similarity yielded the expected results: $\text{car}$ and $\text{truck}$ are more similar to $\text{vehicle}$ than are $\text{dog}$ or $\text{cat}$, while the latter two are more similar to $\text{pet}$ than are the first two, and the cosine similarity ratings between all semantically related terms are also high relative to arbitrary pairs of vectors (such as $\text{truck}$ and $\text{dog}$), usually by a factor of 4 at least. Similarly for analogy-based equations like those discussed in (Milokov et al 2013a): the vector that results from $\text{king} - \text{man} + \text{woman}$ has

---

79 I.e. a cycle through the training data.
80 The approach to learning word embeddings in (Huang et al 2012) also appeals to both local and global context, and addresses ambiguity by learning multiple embeddings per word (methods without this feature will attempt to find a common embedding for both senses of “bank”, for example, which is probably a poor modeling choice as it averages two very different sets of contexts). Huang et al’s approach has received wide uptake in further research and is often used as a benchmark to test new methods.
81 They can be downloaded, as of 8.9.2018, from https://nlp.stanford.edu/projects/glove/. For my casual experiments, I used the 300-dimensional vectors.
a higher cosine similarity with **queen** (0.689) than with **dog** (0.148), **truck** (-0.001), **man** (0.105), or **woman** (0.467).

The latter, rather high, score can be explained by the fact that ‘woman’ and ‘queen’ should, of course, be somewhat closely related. And note that the resultant **queen** vector is least similar to the vector for the inanimate ‘truck’ (negative values of the cosine similarity metric indicate vectors pointing in opposite directions), versus all the animate nouns tested. The similarity of **queen** to words of different syntactic types also tends to be low: for example, **swimming** (0.063) and **improved** (0.022). Comparing **queen** to **of**, somewhat surprisingly, yielded a similarity value of 0.269 (higher than that for **man** or **dog**). But this could be chalked up to the fact that the GloVe vectors are, in the end, dependent on co-occurrence counts, and as mentioned earlier, these will be high for all terms with respect to words like ‘and’ and ‘of’. And indeed, similar values (~0.1 - 0.2) can be observed when most of the above words are compared with **of**. On the other hand, cosine_sim(**of**, **at**) is slightly higher, at 0.392.

These informal experiments are little better than anecdotal evidence, but they suggest the power of such vector space models of meaning, at least at the word level. Not to cherry-pick, I should admit that these experiments yielded counterintuitive results as well. For example, **big** and **small** are more similar to one another by a significant margin (about 0.2) than are **big** and **tall** (similarly for **large/small** VS **large/big**), whereas intuitively, the latter pair are more similar in meaning than the former pair. However, this may be due to the fact that pairs like ‘large’ and ‘small’ simply occur together often, despite their being, in one sense, opposite in meaning (for example, in phrases like ‘large or small’, ‘great and small’, etc).

This may suggest a limitation in VSMs or at least in extant methods for constructing them from data, but it is not clear that such results are necessarily damning: ‘great’ and ‘small’ are similar in meaning along one dimension (they are both basic terms for magnitudes), though
they clearly differ along others that, to the average human at least, are perhaps more salient. The cosine similarity metric is sensitive only to the difference between the orientation of two vectors in $N$-dimensional space. Differences along any dimension contribute equally. We may care more about certain dimensions than others, but it's not obvious why the similarity metric should. It is thus important to supplement direct similarity measures of this sort with performance measures on relevant tasks.

The jury seems to be out on which of the approaches considered so far is superior (see [Pennington et al 2014](#) for strong *a priori* arguments and empirical results that favor GloVe vectors, but [Levy et al 2015](#) for some principled doubts), and of course there are many other similar approaches that I have not mentioned here. [Baroni et al 2014](#) presented *prima facie* strong evidence that using vectors derived from “predictive” methods like skip-gram and GloVe improved performance on natural language processing tasks relative to using more traditional count-based approaches (including those that transform raw count vectors in sophisticated ways, as mentioned above). However, [Levy et al 2015](#) argue that the apparent advantage of prediction-based models over count models claimed by Baroni et al is really attributable to unexamined hyperparameter choices.

For my purposes, it does not matter much whether prediction-based models (which are closely associated with connectionist models, for reasons just discussed) outperform count-based models, so long as they perform competitively. I am interested in prediction models because, given their focus on local context windows and online learning methods, rather than explicit use of global co-occurrence counts, they provide more psychologically plausible models of language learning. In the next section, I consider a VSM model that is inferior to those just discussed in terms of performance, at least as tested in [Baroni et al 2014](#), but
interesting from a philosophical point of view for its rationale, and the light it may shed on the issues about compositionality that are the main theme of this dissertation.

3.5 Collobert et al’s model: syntax from distribution

In a relatively early proposal in this particular area (e.g. VSMs learned in the context of artificial neural network models), (Collobert et al 2011) propose a different way of constructing an adequate vector space model that shares some of the characteristics of Milokov et al’s approach (in particular, the “skip-gram” model). First, I’ll describe the role that VSMs play in Collobert et al’s model, since this helps bolster the empirical case for the utility of VSMs in natural language processing. Then, I’ll discuss the model itself and its rationale.

Collobert et al were interested in building a *general-purpose* model for use in natural language processing (NLP) tasks: a single model that would obtain good results on disparate tasks (for example: part-of-speech tagging, chunking or “shallow parsing” and semantic role labeling). By contrast, state-of-the-art approaches at the time typically relied on features (i.e. inputs) crafted specifically for the task at hand (as many cutting-edge systems today still do). The goal of (Collobert et al 2011) was to avoid the use of such hand-tailored features and come up with a single way of representing linguistic expressions that lent itself well to many different downstream tasks. Part of their explicit motivation was to provide an NLP model that constituted a plausible step toward genuine artificial intelligence (Collobert et al 2011, p.2494).

Before the present era in which machine learning and in particular deep learning approaches tend to rule in many fields (such as, perhaps most overwhelmingly, computer vision), multilayer neural networks were not in wide use in such applications. But the ability of such nets to learn appropriate representations from data attracted Collobert et al to the use of
neural language models. As a result, they constructed an early example of the kind of large neural network model, trained end-to-end on data with learned representations in many hidden layers, that has become a staple of the discipline of machine learning, employing many of the techniques (such as convolution and max pooling) that have featured mostly in contemporary computer vision models (cf. §§4.4—4.6 below).

Collobert et al proceeded as follows: first, they trained their models in a supervised way on data, using backpropagation through all layers. It is not to my purpose to describe their models or optimization procedure in detail, except to note that the first layer works just as the first layer in (Milokov et al 2013a) did, minus the recurrence: a “one-hot” word encoding is used as a lookup table to select a feature vector that corresponds to each input word. These vectors are then fed to subsequent layers for further processing, ultimately outputting tags (semantic role labels, part-of-speech tags, etc) that were matched against the “ground truth” data to determine error gradients for the weights.

The same architectures were trained on each successive task (in fact, despite the goal of finding a single general-purpose architecture, they employ a window approach for most tasks, and a network that looks at entire sentences for the semantic role labeling task, which unlike the other tasks depends on information about long-term dependencies within sentences). The results obtained using this method were below benchmark systems (e.g. solid then-state-of-the-art approaches that employed hand-engineered features).

To improve performance on the tasks, Collobert et al proposed learning better word representations for use in the first, lookup-table, step in processing, using “Lots of Unlabeled Data” (Collobert et al 2011, p.2510). That is: unsupervised learning from a large corpus would be used to improve the word representations fed to the rest of the network and used in the NLP tasks of interest. Of course, representations $W$ learned in this way could not be expected to
yield correct predictions for tags without some principled function from $W$ to the output tags. Collobert et al first learned the word representations in the unsupervised way just described, then simply repeated the original supervised training procedure, with the initial states of the word vectors initialized to the learned representations $W$, rather than beginning from a random initialization as before.

The result, as desired, was performance competitive with benchmark systems, but without much of the computational complexity, and human labor, required by the latter. (Collobert et al 2011) was then an early proof-of-concept of what was later demonstrated beyond doubt by Milokov and his collaborators: distributionally derived vector space representations of word meaning are powerful general-purpose representations capable of capturing syntactic and semantic information. These VSM representations can act as modules that can be combined with other (supervised or unsupervised) learning goals. While the newer models discussed earlier (e.g. skip-gram and GloVe) generally outperform the Collobert vectors (see Baroni et al 2014), the latter nonetheless perform very well.

How did Collobert et al achieve this? First, they trained their word vectors on a very large corpus (by the standards of 2011, at least): “the entire English Wikipedia” (p.2511). They use a training procedure that is in a way very similar to the skip-gram method: a window (whose size is a free hyperparameter of the model) “slides” over all the text in the corpus during training, and for each window, the network is trained to maximize the likelihood of the observed text windows with respect to parameters $\theta$ (in this case, the word vectors).

The real interest of the Collobert et al approach to VSMs, for my purposes, lies in the training objective and cost function used for training, and its rationale. The skip-gram model discussed in (Milokov 2013b) and (Milokov 2013c), again, attempts to maximize the (log) probability of the center word, given the surround. It uses the cross-entropy cost function to
reduce the discrepancy between the probabilities observed in the data and those predicted by the model, or a more efficient approximation to this cost function. Collobert et al use a similar but crucially different approach: they attempt to maximize the ranking (or “score”) assigned to observed windows, and to simultaneously minimize the score assigned to “dummy” windows generated by replacing the center word in observed windows with every other word in the corpus. The cost function is at 0 when the real window is assigned a score of 1 or more and the fake windows are assigned 0 (see the cost function in Eq.17 of Collobert et al 2011).

Collobert et al (2011, §4.6), in a somewhat unusually philosophical passage for a technical paper, discuss why such a ranking criterion should be, on conceptual grounds, supposed capable even in principle of learning representations that would help the network perform well on tasks that depend on syntax and semantics. They appeal (perhaps not surprisingly, given the tight connection between distributional semantics and VSMs) to the work of Zellig Harris. They point out that Harris’s approach to language, which they call an “operator grammar” (Collobert et al 2011, p.2515), shows that in principle, the structure of language can be described in terms of relations of derivability that obtain among sentences, rather than in terms of hierarchically organized syntax trees.82

As described in (Harris 1968), sentences can be grouped into equivalence classes by noting how the replacement of one of their words with arbitrary other words affects graded acceptability judgments. If the ranking among candidate replacement words is preserved across two sentences, they are in such an equivalence class, and correspondingly one can be derived from the other by application of an operator that performs a specific transformation. A virtue of this approach is that such acceptability judgments encompass both strict grammaticality judgments and semantic oddness (“green ideas sleep” would be assigned a low ranking, but

82 Of course, the former representation should capture the syntactic facts explicitly marked in the latter (though the former is richer in that it also captures semantic relations).
perhaps not as low a ranking as “green sleeping therefore”). Given that the words used as candidate replacements can themselves be defined in terms of distributional properties, e.g. of phoneme sequences (see Harris (1968) and §3.1 above), this “algebraic structure” defined over sentences should in principle be learnable from the corpus.

Despite this, one might wonder whether the sort of learning involved here is truly psychologically realistic. Language learners may be expected to assign high probability to phrases they hear, as opposed to those they don’t. But they aren’t generally exposed to negative examples in nearly the quantity that this training regime suggests, if at all. A brief digression on principles of unsupervised learning explains why this is not a fatal objection.

In general, unsupervised learning involves the incremental improvement of a generative model, implicit or explicit (Kiefer and Hohwy 2017, §2.3), in which the goal is to narrow the gap between the model’s distribution over inputs and the distribution of the data. This requires both assigning high probability to the data points and assigning low probability to other points in the same space. This can be accomplished in bidirectional models (e.g. ones with both top-down generative connections and bottom-up connections for inference from data—see e.g. Hinton 2007) by adjusting the model parameters so as to maximize the probability that the latent variables (i.e. hidden-layer states) inferred from the data via bottom-up connections will generate vectors in or near the data set via the top-down connections. Increasing the information carried by hidden-layer states about the input, while at the same time penalizing arbitrary “fantasies” produced by the generative model when it is running free of input, accomplishes this objective.

83 Here, the difference between a purely statistical account of language processing and a VSM is important: even though the probability of either of these sequences in the corpus may be 0, the relatively greater similarity of the first to observed sequences may yield a higher ranking for it.

84 I am, of course, assuming vector-valued input data.
The Contrastive Divergence algorithm (Carreira-Perpiñán and Hinton 2006), for example, works by adjusting the (symmetric) weights of a Restricted Boltzmann Machine (RBM) (Hinton 2010) so that the generative model embodied in the network’s top-down connections is more likely to generate vectors similar to data points, and less likely to generate merely “fantasized” data. Positive samples are supplied by feeding the network external input and computing states of the hidden layer. Negative samples are supplied by letting the network generate states of the input layer top-down. Weights are then adjusted in a way that combines Hebbian and anti-Hebbian learning rules (see, e.g., Hinton and Sejnowski 1999) to maximize the probability of the real data while minimizing that of the “fake data” under the generative model. When the generative model is perfect, the updates cancel out and learning stops.

We may suppose that negative examples are supplied in the same way for natural language learning: “fantasized” sentence representations can be produced, and a learning algorithm can use them to decrease the “score” of non-observed sentences. This learning procedure of course depends on the bootstrapping of decent generative and recognition models from one another, as discussed in (Frey et al 1997)\textsuperscript{85} and elsewhere. Learning based on negative examples will perhaps be more effective when the generated examples are somewhat similar to the input data, i.e. after some learning has already taken place. Collobert et al’s procedure may then be seen as a more efficient version of this procedure in which the “generative model” is perfect from the start, except for the center word.\textsuperscript{86}

I’ll conclude this section with a brief discussion of Collobert et al’s empirical results. In addition to (admittedly somewhat modest) performance increases, the vector embeddings

\textsuperscript{85} See (Kiefer and Hohwy in print, §1.3) for discussion.
\textsuperscript{86} It is also unlikely, of course, that an organically grown generative model would regularly swap out just the center word in a phrase for a random replacement. I assume (I hope safely) that this does not affect the argument in its essentials, and that generically poor examples of phrases generated by the model would enable the same sort of learning, perhaps less efficiently.
learned via Collobert et al’s unsupervised approach were, in terms of semantic and syntactic relatedness properties, far superior to the vectors learned from the supervised tasks alone, and fairly decent in absolute terms. I reproduce a portion of the sample of embedding vectors exhibited in Table 7 of (Collobert et al 2011) as Figure 5 below.

As can be seen from the table, the word vectors are of good quality in terms of semantic and syntactic relatedness. They are not perfect, however—at least, debatably. Intuitively, “scrapped” is perhaps closest to “scratched” of the 10 nearest neighbors listed—certainly closer in meaning than “smashed” or “punched”, though keeping in mind how distributional semantics works, it’s worth bearing in mind that “scratched”, like “smashed”, “nailed”, and “punched” but unlike “scrapped”, probably often occurs as a transitive verb in contexts involving aggression. The ranking of “God” above “Christ” for “Jesus” is, while perhaps theologically correct, surprising, as is the inclusion of “Satan” on the list just below “Christ”. Vector space models in semantics perhaps have a tendency to build guilt by association into the meaning of a word.

<table>
<thead>
<tr>
<th>FRANCE</th>
<th>JESUS</th>
<th>XBOX</th>
<th>SCRATCHED</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUSTRIA</td>
<td>GOD</td>
<td>AMIGA</td>
<td>NAILED</td>
</tr>
<tr>
<td>BELGIUM</td>
<td>SATI</td>
<td>PLAYSTATION</td>
<td>SMASHED</td>
</tr>
<tr>
<td>GERMANY</td>
<td>CHRIST</td>
<td>MSX</td>
<td>PUNCHED</td>
</tr>
<tr>
<td>ITALY</td>
<td>SATAN</td>
<td>IPOD</td>
<td>POPPED</td>
</tr>
<tr>
<td>GREECE</td>
<td>KALI</td>
<td>SEGA</td>
<td>CRIMPED</td>
</tr>
<tr>
<td>SWEDEN</td>
<td>INDRA</td>
<td>PSNUMBER</td>
<td>SCRAPED</td>
</tr>
</tbody>
</table>

**Figure 5** (Adapted from Table 7 of Collobert et al 2011). Word vectors learned from the Wikipedia corpus, for the 100,000 most common words in the Wall Street Journal. Top row: word corresponding to a vector; below: words corresponding to 6 nearest neighbors of each vector in order from top to bottom, using Euclidean distance as the metric. “NUMBER” in “psNUMBER” is a generic token substituted for numbers in words, used to reduce vocabulary size.
3.6 Compositionality in vector space semantic models

Over the past several sections, I’ve been describing empirical results in computational linguistics and machine learning suggesting that in distributionally based vector space representations, words can be represented by vectors whose locations in the vector space map onto their syntactic and semantic properties. A representation that contains this information can, in principle, be exploited so as to yield good performance on tasks that depend on it, and thus should be capable of driving sophisticated cognitive behavior of various sorts. It’s also important that such representations can in principle be learned in a realistic, online fashion, from data that language-learners could expect to be exposed to.

However, I have so far focused on relationships between words, and have not provided much of a hint as to how VSMs are meant to accomplish the main task that Smolensky’s tensor product system and other VSAs were designed for: namely, explicitly allowing for connectionist representations that exhibit systematicity (and perhaps at least a limited degree of productivity). This is a matter of the relations between complex expressions (phrases and sentences). In this section I will consider how VSMs might capture the meanings of phrases in such a way as to yield a type of compositional (or, strictly speaking, combinatorial) system.87 One particular approach to this problem is the subject matter of the following chapter.

There is in fact a sizeable literature on compositionality in vector space models. In this section, I will attempt only the briefest of summaries of the options, as outlined in the survey in (Baroni 2013), focusing on a few models of particular interest. First, I will consider one type of

---

87 In this section, I defer to the practice in the relevant literature of using ‘compositionality’ to cover what I have called a combinatorial system of representations. Computational linguistics does not, on the whole, seem concerned with the philosophical niceties around Classical constituency, as defined above.
strategy for deriving sentence representations from word representations that shares a feature of the HRR system (Plate 1995) discussed above in §2.5. The feature in question is that, in HRRs, vectors that correspond to the most syntactically complex objects are in the same space as vectors that correspond to atomic constituents, and everything in between. This has important philosophical consequences, which I’ll take up in Chapter 5.

One such approach to compositionality in VSMs is to use simple arithmetic operations, such as addition, on the word vectors to yield phrase and, potentially, even sentence vectors. That is: the vector for ‘yellow cup’ might be, simply, the vector for ‘yellow’ plus the vector for ‘cup’. Interestingly, on this proposal, composition of words into complex phrases is, in functional terms, little different from paradigmatically inferential tasks like analogy completion (see again Milokov et al 2013a), or from similar compositional principles that have proven effective for non-linguistic representations such as images (see again Bojanowski et al 2018 and Radford et al unpublished). Milokov et al (2013c) demonstrate the surprising efficacy of this arithmetic composition method at the level of semantics: in a vector space learned via their methods (i.e. word2vec), the closest word vector to $x = \text{Vietnam} + \text{capital}$ is Hanoi (see their Table 5 for additional results). The phrase ‘Vietnam capital’ is of course ill-formed, but since syntactic and semantic properties are intermingled in vector space models, the same approach should work well for constructing representations of genuine phrases like “capital of Vietnam”.

Moreover, by adding phrase vectors in this way, a sentence vector could be constructed. More sophisticated methods might use more complex operations, such as averages or weighted sums. As shown in §3 of (Baroni 2013), componentwise multiplication of vectors seems to have some desirable properties (see in particular Table 2, in which it is shown that for a toy example involving 5-dimensional vectors, this way of composing sentence representations (cf. Mitchell and Lapata 2008) avoids some pitfalls of the simple addition method).
There is no reason, however, that the composition function must be limited to simple algebraic operations that map input vectors to an output vector in the same space. Indeed, arbitrary functions could be employed, including functions that map to a new vector space. The tensor product is often cited, along with (Smolensky 1990), as an example of this kind of composition function (for example in Baroni 2013 and Clarke 2012). Other functions of this sort including convolutional approaches that do not involve compression, such as aperiodic convolution (see Plate 1995), have been considered. I have emphasized the differences between VSAs like Smolensky’s, on the one hand, and VSMs, with their “continuous semantic spaces”, on the other. But the tensor product as a mathematical operation can of course be employed to combine vectors without worrying about the constraints on vectors, and associated guarantees of recoverability, that Smolensky’s analysis suggests. Rather than being used specifically to bind constituents to roles, the tensor product could be used simply to compute sentence or phrase vectors on the basis of word vectors. Similarly for other VSAs.

There are two principled reasons to prefer function-based composition methods to simpler and in some ways more natural arithmetic methods, independently of the issue of mapping to a new vector space, which is a separate matter. One is that functional composition potentially remedies a very important limitation of the arithmetic methods: since addition, multiplication and other basic arithmetic operations are commutative, sentence- and phrase-representations computed from word-representations using these operations are insensitive to the ordering of words in the sentence/phrase.

This is, in my view, a fatal limitation in the present context, where the aim is not just to construct a language model that performs well enough on some specified task, but to model psychological reality. Though systems insensitive to ordering can often perform surprisingly well on NLP tasks (see for example Blacoe and Lapata 2012), such representations cannot
differentiate the meanings of even simple sentences that are related by permutation of words, such as ‘John loved Mary’ and ‘Mary loved John’. Function-based composition methods can of course overcome this limitation, since many functions are not commutative.

The second motivation for function-based compositional systems is that, as Baroni (2013, p.514) notes and as mentioned in § 3.4, VSMs have trouble dealing with words that are primarily grammatical in their function, such as prepositions and logical constants, as opposed to “content” words like nouns and verbs. The trouble, again, is that the former do not strongly select for particular contexts. Indeed, the semantics of the logical connectives is so different from that of nouns and verbs that “formal” systems are distinguished from others by the fact that they are agnostic about (non-logical) semantics (see Quilty-Dunn and Mandelbaum 2017, fn.8). Prepositions, determiners and other words in this category function similarly abstractly. Even state-of-the-art approaches to natural language processing and other sophisticated cognitive tasks often discard such function words as “stop words” (Baroni 2013, p.514).  

The functional approach to composition doesn’t on its own solve this problem. But it suggests the following strategy: use VSM-type vectors, distributionally derived, only for the content words, and have grammatical function words trigger functions that operate on the representations of the content-words (Baroni 2013, p.516). Relative to the proposals discussed earlier, this implies a more complex overall architecture in which VSMs play a limited, but essential, role, along with an approach to composition that has much more in common with traditional approaches in formal semantics. Relatedly, (Baroni and Zamparelli 2010) argue for a system in which adjectives are modeled as empirically derived matrices that allow noun vectors to be mapped to other noun vectors.

---

88 Relatedly, the system for aligning images with verbal descriptions in (Karpathy 2016), discussed below in §5.6, does so, on the grounds that such words do not correlate well with visual appearances.

89 Importantly, such approaches would still fall short of the definition of Classical architectures in (Fodor and McLaughlin 1995).
This suggests a whole space of vector models in semantics that employ linear algebraic operations in a richer way than those contemplated in simple arithmetic models. The best approach may well lie somewhere in this space. However, Clarke (2012) argues that many vector spaces have an implicit lattice structure, which can be used to define asymmetric relations between vectors that could serve to model entailment relations and other operations in formal semantics. This shows that perhaps, given an extensive enough corpus and a sensitive enough learning procedure, VSMs could be learned that handle logical as well as statistical relations among representations, without proposing radically different types of representation for different types of linguistic expression (cf. Widdows and Cohen (2015) and §6.4 below).

An additional benefit of thinking of composition in terms of arbitrary mathematical functions is that artificial neural networks can, of course, be conceived of as complex functions. Baroni (2013) cites, as one motivation for pursuing a combination of compositional and distributional methods in linguistics, the fact that “Distributional models might be better at handling commonsense aspects of entailment, such as how the richness and ambiguity of lexical representations affect inference” (p.514). For example, “ten dogs in the kitchen entails many dogs in the kitchen but ten ants in the kitchen doesn’t entail many ants in the kitchen.”

For composition functions to be thus sensitive to the nuanced, sometimes empirically derived properties of the meanings of particular expressions, it would be preferable to use a set of learned composition functions rather than the small set of fixed functions employed in the case of tensor products and other VSAs. And as discussed in connection with (Collobert et al 2011) above, it’s hard to beat multilayer neural networks as tools for the learning of such representations from linguistic corpora.

One downside to the switch from VSAs to compositional VSMs is that the latter do not always guarantee that word vectors, once composed into sentence vectors, can be recovered
from the sentence representation. That is, there is not always a well-defined “unbinding” or extraction procedure. For some purposes, this doesn’t matter, but again, for the purposes of psychological modeling, it should be possible for representations corresponding to semantic constituents to be derived from the representation of a complex state of affairs (even if we do not assume Classical syntax). There are, however, neural network models that learn to decompose vector representations as well as composing them (Pollack 1990, Socher et al 2011a, b, c), which I’ll discuss presently.

Baroni (2013, p.516) suggests that linear transformations are most appropriate for VSMs, and neural networks of any complexity typically use nonlinearities at each layer. But, while this may be overkill, it is not obviously a serious flaw in the approach. Note that neural networks also supply a very natural way of building up representations for more complex phrases (up to sentences and beyond) out of those for simpler ones: recurrent neural networks accumulate information from multiple time-steps in a single “hidden layer” representation, in a way that preserves information about temporal ordering, and are thus ideal for representing sequences (such as sentences). As I’ll discuss in the next chapter, variations on this approach have yielded some of the most powerful natural language processing systems to date.
Chapter 4: Parsing in Connectionist Networks

4.1 The RAAM system and autoencoders

In this chapter, I'll consider neural-network-based approaches to vector composition for the purposes of natural language processing. The simple arithmetic and function-based proposals for vector compositionality discussed in the previous section have something important in common with the VSAs discussed earlier: they make use of a small, preset stock of composition functions. Over the next few sections, I'll consider models that instead learn their composition functions as well as their vector representations from data, and construct sentence-level from word-level representations on the basis of them. The primary goal in many of these systems is to correctly parse sentences.

I'll focus mainly on a group of models proposed for natural language processing by Richard Socher and colleagues, whose common architectural commitment is to the Recursive Neural Network (henceforth, “RecNN”, following (Kalchbrenner et al 2014), to distinguish it from the Recurrent Neural Network or RNN). A RecNN applies the same set of weights to successive input representations, where the input to the next stage of processing may include the output of the previous pass. Recurrent neural nets also have this feature, so, by this definition, RNNs are RecNNs. The difference between these types of network, apart from the extrinsic fact that RNNs are typically deployed on time-sequences rather than on static structures, is nicely illustrated in an example of Richard Socher’s (see Fig. 6).
Figure 6 (Adapted from Richard Socher’s lecture notes for CS224d: Deep NLP at Stanford University). Top: A recursive neural network whose structure mirrors the syntactic structure of a represented phrase. Here, boxes indicate vectors representing constituents at each node (from lexical items to NPs, VPs, etc, up to full sentences). Bottom: the same phrase as it would be processed by a vanilla RNN, shown “unfolded” left to right along the temporal dimension. Here, each “hidden state” vector represents the portion of the string processed so far, and needn’t map to a syntactic constituent. In both cases, arrows indicate transitions mediated by synaptic weight matrices.

The RecRNN, structurally speaking, is just a generalization of the RNN in which the same layer may receive multiple inputs in the same processing step. When RecRNNs are used to process structures in the obvious way, the networks themselves take on the topology of the processed structure, e.g. syntax trees in the case of NLP (where the tree structure corresponds to a series of diachronic algebraic operations on vectors). Importantly, as in RNNs, the synaptic weights used to compute the influence of the input on the output representation are the same.

---

across all representations and all temporal steps (abstracting, of course, from weight changes
during online learning), as illustrated in Fig. 7, adapted from (Socher et al 2011c).

RecNNs have their roots in the sister model to (Smolensky 1990): the RAAM (Recursive
Auto-Associative Memory) system (Pollack 1988, 1990), an early connectionist response to
Fodor and Pylyshyn that in effect instantiates a compositional vector space semantic model of
just the kind I’ve been discussing. It is worth examining this model at some length because,
despite its age and attendant limitations (for example, this model did not learn from real natural
language corpus statistics), it has the essential features of the most promising connectionist
approaches to systematicity that I’ve considered thus far, and helps to cement some of the
positions I’ve been arguing for (such as the view that ordinary connectionist networks can in fact
do semantically and syntactically sensitive processing, and demonstrate systematicity).

Essentially, the RAAM system is an autoencoder (Hinton and Salakhutdinov, 2006). An
autoencoder, whose principles of operation are very closely related to the unsupervised learning
of generative models (cf. §3.5), is a neural network that learns by minimizing the difference
between its inputs and the facsimiles of those inputs that the network reconstructs from a neural
code in its hidden layer. The simplest autoencoder architecture consists just of an input layer, a
hidden layer (typically containing fewer units than the input layer), and an output layer that is of
the same dimensionality as the input layer. Error is backpropagated during learning by
comparing the output layer’s “reconstruction” to the input layer for each data point. The result is
an encoding-decoding scheme learned entirely from data: the weights from input to hidden
layer are the encoder, the hidden-layer representations are the code, and the weights to the
output units are the decoder (see Pollack 1990, p. 8).

---

91 Conceptually, the Helmholtz machine (Hinton et al 1995), "unfolded" so that the top-down path from the
highest-level neural code to the input layer is conceived of as a new feedforward network, is very similar
to an autoencoder (see Kiefer and Hohwy 2017), though the weights are optimized in a different way.
The RAAM model, like the model in (Socher et al 2011a, b, c) to be discussed later, is a special case of autoencoder in which the structure of the network is recursive. In the RAAM, binary trees are first encoded by feeding both leaves of a parent node, represented as concatenated $k$-dimensional vectors, to the $2k$-dimensional input layer, and using it to compute a hidden-layer code of $k$ dimensions. The code for this constituent can then be paired with the representation of its sibling node in a larger tree structure (if applicable), and the process is repeated until the entire tree is encoded in a $k$-dimensional representation of the root node.

Decoding works by recursive application of the decoding weights to the representations of parent nodes, starting with the root node, until the full tree is decoded. For a given training example, a tree is first encoded, then decoded, and the reconstruction error is computed from the leaf nodes of the encoded tree, and backpropagated through the successive weights and representations, so that the weights can be adjusted to minimize the average reconstruction error on future examples. The recursive autoencoder in (Socher et al 2011c) works slightly differently: there, reconstruction error is computed directly for each node-representation. This network involves recursive application of the autoencoder architecture, while the RAAM model is in effect a single larger (multi-layer) autoencoder (see Fig. 7 for an illustration).

---

92 I must stress, as I did in discussing Smolensky architectures (§2.1), that the conflation between *representing trees* and *representations that function as trees* is principled here. I speak in the way (some) connectionists tend to here, mostly for brevity: compare “representation of the root node” to the extensionally equivalent “representation whose content corresponds to that of the phrase or sentence that would be represented by the root node”.

93 The decoding and encoding matrices can be different. What matters conceptually is that the same matrix is used for each recursive encoding operation, and each recursive decoding one.

94 The details of the way in which reconstruction error is computed vary across models, but the basic idea is simple enough: compare the input vector and the network’s reconstruction of it using one of the well-known metrics of vector similarity (e.g. Euclidean distance, or the cosine similarity measure mentioned earlier). One perspicuous example is a stochastic autoencoder trained on binary-valued vectors. There, learning could proceed via the cross-entropy cost function.

95 The latter architecture is also used in (Socher et al 2011a), there called an “unfolding” recursive autoencoder.
Phrase-level vector representations $y$ are computed by applying composition weight matrix $W$ recursively to inputs $x$ and previous outputs. Error is backpropagated from reconstruction targets (darkened boxes) through the structure to be learned (unshaded boxes). Left: as in the RAAM system (Pollack 1990), a set of “decoding weights” is applied recursively to the root-node representation and then to the reconstructed phrase representations, and error is backpropagated from the leaf nodes. Right: as in (Socher et al 2011c), reconstructions are computed separately from each intermediate output.

In general, the motivation behind autoencoders and many other unsupervised learning systems is to learn compressed representations that nonetheless allow for adequate modeling of the data distribution (cf. the “Minimum Description Length” principle—Hinton et al 1994). Autoencoders in general conform to this principle, but in a recursive NN applied to natural language structures such as phrases and sentences, the network must learn both compressed representations for constituents of various sizes (neural “codes” for syntactic structures) and a set of weights that allows syntactic elements of arbitrary depth\(^{96}\) to be composed with one

---

\(^{96}\) Normally, I would include some such qualifier as “within reason, given processing constraints, etc”, but in fact Pollack argues that a neural network could subserve an \textit{infinitely} productive capacity provided that its hidden-layer vector space representation is fractal in nature (see Pollack 1988). This is an interesting proposal that I will not, here, touch with a 10-foot pole.
another.97 The result is a system that has exactly the properties I attempted to motivate \textit{a priori}
earlier: it provides for the encoding and accurate retrieval of syntactic trees of arbitrary depth, without the artificial limits on allowable vector representations imposed by VSA systems.

Indeed, Pollack (1990) shows that the neural “codes” learnt by the system have the key property of VSMs: semantically and syntactically similar structures are represented by similar vectors in the hidden layer’s vector space. In this case, this result obtains not because of regularities in corpus statistics \textit{per se} (Pollack’s models were very small and only trained, in fact, on simple, artificial toy examples) but because in order to get the transformations right and thus decode many encoded trees properly given the single set of weights, the network is forced to learn economical representations.

Here is one way of looking at the connection between the structural-representational idea that differences among representations should correspond to differences among semantic values (O’Brien and Opie 2004), on the one hand, and efficiency considerations, on the other: representing similar structures by dissimilar codes would mean representing the same thing (i.e. whatever is in common across the structures) via two different codes.98 This redundancy is less efficient (uses more degrees of freedom) than representing that thing in only one way, if the latter can also achieve good reconstruction accuracy.

If similar trees have similar codes, one would expect the network to embody a certain kind of generalization: novel trees with codes close to “observed” cases will be treated similarly.

\footnote{97 Pollack (1990) stressed the fact that the weights and representations in such a network \textit{co-evolve}: the entire encoding-decoding scheme is learned simultaneously, rather than learning transformations between fixed representations. A similar principle is at work in most contemporary connectionist models, and is referred to these days as “representation learning” (see Bengio et al 2012).}
\footnote{98 Actually, and very importantly, (Hinton et al 1995) show that in the context of a \textit{stochastic} model like the Helmholtz machine, using more than one code to represent the same thing can actually improve the efficiency of the code. This kind of probabilistic coding scheme is central to the motivation, in that paper at least, for the idea that improving the (log) probability of evidence under a model is equivalent to minimizing the (Helmholtz) free energy of the system. I will not worry about this here, however, since the models under consideration are deterministic models.}
by the network. Pollack (1990) shows, in §4.1 of his paper, that this theoretical prediction is borne out: the network in fact generalizes to unseen examples (being able to properly encode and decode X LOVES Y for (X,Y) pairs unseen in the training data), which he takes as evidence that this proposal meets Fodor and Pylyshyn’s systematicity (of representation) challenge.

Surprisingly, perhaps, this kind of code also allows for the learning of direct associations between inferentially related propositions, whose representations had previously been learned using the RAAM system just described (see Pollack 1990, §4.3.1 for discussion). Pollack shows that the network can learn the (dubious) inference rule “IF (X LOVES Y) → (Y LOVES X)” for all values of X and Y, even if only exposed to a subset of them. Thus, the network models at least a degree of inferential systematicity.99

Moreover, Pollack argues that the system works in a way that builds such systematicity more deeply into the architecture than is possible in Classical systems, since systematicity is a necessary consequence of the architecture (when appropriately trained). That is: Classical symbol processors like Turing machines can in principle work with arbitrary bit strings, and syntactic rules must be imposed by software for processing in particular domains. The syntactic rules are only “built in” at the level of virtual machines. By contrast, the RAAM system has its syntax built into the hardware (once learned). The fact that this systematicity arises only after training does not in the least weaken the argument: since representation and processing are so intimately linked in this system (and, arguably, in connectionist systems generally), any system that can represent the items in a given domain at all will be able to do so if and only if it can also represent them systematically, and any such system that can perform inferences over them at all will, similarly, exhibit inferential systematicity.100

99 It will, no doubt, be objected that associative transitions can’t be inferential. I devote two sections to this argument in Chapter 6.
100 The fact that the word vectors discussed earlier in connection with VSMs can be treated as detachable “modules” might seem to conflict with this claim. But these vectors do exhibit systematicity: embedding
Finally, it is worth noting that these inferences, like those in Smolensky’s system, don’t depend on first decoding a syntax tree into its constituents. Rather, the systematicity of inference derives from the structural-isomorphism principle that we have also seen to be at work in the semantics of VSMs more generally: the network can correctly perform the unseen inferences because the input and output representations share structure with the observed cases, without thereby being sensitive to Classical constituent structure (cf. §1.3). I argued earlier that the ability of VSAs to “leap” directly from input to output representations of a syntactic transformation in a single associative step, without computing the steps in between, is consistent with structure-sensitive processing because such rules presumably would correspond, in anyone’s actual psychology, to cases of the overlearning of transitions that were once accomplished stepwise. By contrast, Pollack’s system, like VSMs generally, points to a more radical departure from Classical processing, in which the principles of composition themselves are moved into the domain of empirical learning.

4.2 Discovering parse trees

Contemporary research in machine learning and computational linguistics has made use of variants on the recursive neural network, the conceptual descendant of the RAAM system, to achieve state-of-the-art results on real, large-scale problems in NLP. In this section and the next, I describe these more contemporary models. While these systems are important to consider because they demonstrate empirically that the machinery in question lives up to its theoretical promise, my conclusion will be that, like the RAAM model, they fail to provide vectors learned via the skip-gram technique (for example) in a wider system that licenses interpretation of any of them as representations will license interpretation of all of them as representations (similarly for inference). On the other hand, if they aren’t embedded in a wider system, it’s not clear that they function as representations at all.
psychologically realistic connectionist models of the systematicity of language comprehension, simply because they involve learning procedures that are either based on (explicit or implicit) a priori data about individual target parse trees, or invoke implausible parsing mechanisms.

I'll discuss six models in this section and the next: (1) the recursive autoencoder (Socher et al 2011c), (2) a vanilla RecNN (Socher et al 2010), (3) the tensor-based Recursive Neural Tensor Network (Socher et al 2013b), (4) tree-structured LSTMs (Tai et al 2015, Zhu et al unpublished), (5) a conventional syntactic parser supplemented by semantic phrase vectors (Socher et al 2013a), and (6) a single RecNN applied to both vision and language (Socher et al 2011b). My discussion of each will be brief, and focus only on conceptually important details.

The architecture of a basic recursive autoencoder was described in the previous section. Here, I'll just focus on the merits and shortcomings of the elaborations of this basic model described in (Socher et al 2011a). First, note that like the RAAM model, the basic recursive autoencoder in effect builds a priori knowledge about the encoded tree structures into the learning procedure. Autoencoders are usually associated with unsupervised forms of learning, since they learn to model the distribution of signals in a given dataset, rather than learning a mapping from one variable $x$ to a “label” variable $y$ based on information about the latter. However, for a given training example, the syntax is in effect supplied to the RAAM model by the experimenter: the network’s composition function is applied to two sibling nodes at once to produce a representation of a parent node, and this procedure is then applied recursively until the whole tree is encoded. In supplying the proper ordering of the encodings and the constituent pairings, the experimenter provides direct information about the tree structure to be

---

101 I have perhaps not been as explicit about this as I might have, but I take it that there is a short argument from the systematicity of language comprehension abilities to the systematicity of conceptual abilities. A person who can understand any sentence in a given language must be capable of harboring a corresponding propositional attitude, thus securing systematicity of thought in the sense at issue.

102 See the discussion of the distinction between generative and discriminative models in (Kiefer and Hohwy in print), §1.2.
encoded, information that would not be available to a language learner (who wasn’t already able
to parse the relevant sentences).

Socher et al (2011c) aim to solve this problem by defining a variation on the recursive
autoencoder (henceforth, “RAE”) model that discovers tree structures from input sentences,
rather than only discovering good vector representations for predetermined tree structures.
Given only the concatenated \( k \)-dimensional input vectors for words in a phrase or sentence
(which may be initialized using embeddings previously derived from corpus-based learning, as
discussed above, or else randomly), the network’s job is to simultaneously construct a parse
tree and minimize the reconstruction error at each node (as in the standard RAE).

The strategy employed by Socher et al to discover parse trees is, essentially, one of
brute force: the network searches for the binary branching tree, out of all possible trees over
the input words consistent with its prior decisions, that allow it to best minimize reconstruction
error. The procedure is as follows: first, the network uses its weights to compose the first two
words in the input sentence, deriving a hidden-layer encoding for a “candidate” node. It then
computes the error when this candidate node is used to reconstruct its children. The two-word
window then slides to the right by one word. This procedure is repeated, keeping a record of
the error for each candidate, until the window reaches the end of the sentence. A new string is
then defined by replacing the two word-vectors whose candidate node minimized the
reconstruction error with the representation computed for that node.\(^\text{103}\) Then, the entire process
repeats. At the last step there will be only two input nodes, and in this case only one possible
parse tree. The overall training objective is to minimize the sum of the reconstruction error at
each node.

\(^{103}\) E.g. \([w_1, w_2, w_3] \rightarrow [p_{1,2}, w_3]\), where, following (Socher et al 2011c, p.154), \(w_n\) is the vector for word \(n\)
and \(p_{i,j}\) is the phrase vector for the node whose children are \(w_i\) and \(w_j\). A node higher in the tree might
have a label such as \(p_{1,\{2,3,4\}}\).
The parsing decisions thus made can be expected to be reasonable, as can be shown by a bootstrapping argument: if the network’s encoding weights are already any good, then they will yield lower reconstruction error when applied to genuine constituents (i.e. those the network has “seen”, and similar ones) than when applied to ersatz constituents. The better the weights, the more strongly this will be enforced, so learning should produce a good set of weights as well as a good set of parsing decisions. Initially, of course, the weights will be random, and the lowest-level parsing decisions may be expected to drive learning: the weights will be adjusted so as to lower the reconstruction error of the word nodes.

The authors mention two ways in which this model, though of sound conceptual motivation, might be deficient (Socher et al 2011c, p.154). First, the accurate reconstruction of nodes with many children might be more important than the reconstruction of leaves (since subsequent reconstructions based on the former will be affected by any inaccuracy). This can be avoided simply by weighting the reconstruction of nodes closer to the root more heavily in the training objective. A second, more subtle technical problem is that the Euclidean distance error term between two vectors can be minimized simply by minimizing the magnitudes of the vectors, which the network is free to do since it learns its own representations. The authors avoid this problem by enforcing a constraint of unit norm on the phrase node representations.

Both problems can also be solved by reverting to the “unfolding” architecture of the RAAM, as described in (Socher et al 2011a), where reconstruction error is computed by decoding a node’s entire subtree and backpropagating from the error at the “leaf” nodes. This unsupervised learning method actually produces phrase reconstructions superior to those of the “standard” RAE (see Socher et al (2011a), Table 2 and discussion). Nonetheless, I have focused on the model in (Socher et al 2011c) instead because it is employed within a system that explicitly learns to construct parse trees (the “unfolding” model learns word and phrase
vector embeddings, but does not learn to parse and is deployed by the authors only for the purpose of paraphrase detection).

I think the goal of learning to parse “from scratch” is the right one. The RAE has the virtue of constructing its own parse trees without supervision. It is not constrained \textit{a priori} to produce only parse trees that are valid according to some generative grammar—however, given that there are generative principles underlying the language-production processes that yield corpora, and that the autoencoder attempts to learn a maximally efficient generator for observed data, it should be possible in principle to at least approximate\textsuperscript{104} the actual generative grammar for the language in question (cf. the bootstrapping argument mentioned above).

The brute force strategy employed by the network is, however, less than ideal in terms of psychological plausibility. Keeping a running record of candidate parse vectors for each two-word window in an input sentence may become prohibitive with sentence length, and moreover, the idea that the entire sentence must be observed before a single parsing decision is made, and then re-observed to make the next decision, stretches credulity a bit. This “greedy” strategy does have the merit of efficiently searching the space of parse trees by only considering those consistent with prior decisions on each pass. But a solution to unsupervised parsing that took better advantage of the parallel distributed processing in connectionist networks would be preferable.

In fact, the authors don’t end up testing the architecture just described as such. Rather, they add a second, supervised training objective: using a database of phrases and sentiment labels (from among five categories) provided by humans, they train an additional classifier on the relevant constituent vectors that attempts to predict the sentiment of each phrase. This learning is supervised via backpropagation from the correct answer. The result is a

\textsuperscript{104} “Greedy” algorithms like the one employed here make the best local choices, but are not guaranteed to find the global minima of cost functions.
“semi-supervised” learning system that combines both the reconstruction and ranking scores as the target function for optimization, where the relative weight of these two factors is determined by a free hyperparameter. The authors report then-state-of-the-art performance on sentiment analysis tasks, with a relative weighting of the reconstruction versus categorization error of 0.2 and 0.8, respectively. Thus, the system as actually employed, and the results thus obtained, do not show much about the effectiveness of the unsupervised parsing.

There is a principled reason that the authors resorted to semi-supervised learning in this case, despite their noble goal of learning to parse from scratch: as discussed in connection with (Collobert et al 2011)’s vectors above in §3.5, antonyms (like “good” and “bad”, or “Jesus” and “Satan”) tend to occur in similar contexts yet have opposite affective valences, so unsupervised word vectors do not work well in practice for sentiment analysis, at least within currently available systems. As argued above, it may be hoped that distributionally informed VSMs will eventually overcome this problem, but in any case sentiment analysis, while important for commercial applications such as social media analysis, is not exactly a paradigm NLP task, and is not obviously terribly relevant to the issues at hand in this chapter.

### 4.3 More powerful tree-shaped models

In this section I’ll describe the other recursive models I mentioned above, and end with some brief reflections. Many of the efforts of Socher and collaborators are aimed at discovering a more powerful composition function than a single set of weights (which, apart from capacity considerations, is not at all sensitive to the syntactic types of the input words/phrases). One possibility for achieving this, explored in (Socher et al 2012), is to learn a set of composition functions, in effect one for each pair of lexical items. Socher et al achieve this by learning a
matrix as well as a vector for each word: since matrices can be used to transform vectors, the matrix for a word at a node can be multiplied by the vector for a word at its sibling node, yielding a vector for the parent node (in fact, Socher et al’s proposal applies a weight matrix to the concatenation of the outputs of both possible matrix-vector operations for each pair of words).

I do not consider this model in more depth here because, while it is in a way the purest example yet of a wholly vector-based representational system that accommodates rule-based compositionality, learning a novel composition function for each word pair seems to cut against psychological plausibility and against the efficiency considerations that are essential to most interesting mechanisms for unsupervised learning. In any case, this proposal shares the feature of the other models discussed in this section that is, for me, a “deal-breaker”: it employs thoroughly supervised learning in the form of error backpropagation from class labels.

The model of (Socher et al 2013b), the Recursive Neural Tensor Network (RNTN), uses a tensor that takes the input word vectors as arguments, in addition to a simple weight matrix, as the recursively applied composition function (as Socher et al note, this model is equivalent to the vanilla RecNN considered above when the tensor is populated entirely by zeros). In effect, the tensor, which is learned during training, acts as an array of different possible composition functions, all of which are simultaneously applied to the word pair: “Intuitively, we can interpret each slice of the tensor as capturing a specific type of composition” (Socher et al 2013b, p.6). This model, while achieving state-of-the-art results on sentiment analysis and making the correct call in many subtle, syntactically sensitive cases,\textsuperscript{105} is uninteresting for my purposes because, yet again, it employs supervised learning from class labels.

Socher et al (2013a) have also developed a “compositional vector grammar” (CVG) that finds a happy medium of sorts between the last two proposals discussed: rather than a single

\textsuperscript{105} For example, it correctly takes the effect of negation high in the parse tree into account even when many positive words are present, e.g. in “This film is not an excellent or original example of its genre”.

123
composition function or a unique one for each pair of terms, the CVG applies a weight matrix for composition that depends on the syntactic categories of the child nodes. The result is a nice hybrid between classical and statistical NLP: each word is represented by a discrete syntactic category label (which is used to select the composition function) as well as a high-dimensional embedding in a semantic space.

This model performs competitively with state-of-the-art systems (where success is measured in terms of precision and recall of the correct parse for Wall Street Journal sentences), outperforming almost all then-extant models including the Stanford parser mentioned earlier. Importantly, the CVG was able to learn to make the correct attachment decisions for sentence pairs like “He ate spaghetti with a fork” versus “He ate spaghetti with a sausage”, which are entirely driven by semantics, whereas the Stanford parser failed on this example. However, as in the previous models, it relies on feedback from the correct trees for learning. Beyond this, it employs a standard context-free probabilistic grammar to narrow down the search space before the fine-grained semantic vectors are used in a second pass.

A somewhat different model that combines the basic idea of recursive neural networks with the finest in contemporary connectionist technology is the “tree-shaped LSTM” proposed in (Tai et al 2015) and independently by (Zhu et al unpublished). This model differs from standard recursive nets in that, instead of generalizing a generic recurrent neural network (RNN), it generalizes (in the same way) a more sophisticated recurrence-based network architecture, the Long Short-Term Memory (LSTM) network (Hochreiter and Schmidhuber 1997; see also Karpathy 2016, pp.24-25), which is used today in most state-of-the-art sequence processing applications, including those under discussion here. Tai et al show that this kind of network beats alternative recurrent approaches at predicting human judgments of the semantic relatedness of sentence pairs. Zhu et al achieve similar results on sentiment analysis.
LSTMs, relative to RNNs, introduce structural complexity, and potential departures from biological plausibility, that may be relevant to the thesis of this dissertation. In the LSTM, each parallel computational unit contains its own memory functionally insulated from the rest of the network via multiplicative input and output gates. Subsequent models introduce a memory decay “gate” as well (Gers et al 2000). This complexity is motivated by difficulties in optimizing simpler RNN models, not by considerations of biological plausibility or theoretical elegance. As Le et al (unpublished) point out, it is unlikely that the LSTM is the optimal solution to the problems that motivated it, though it beats the available alternatives and still seems to be the best-performing extant model for sequence processing.

The LSTM achieves its advantage over RNNs in virtue of moving in the direction of Classical systems in at least the following broad respect: each unit exhibits internal complexity that is harnessed to improve performance. While the benefits of LSTMs over generic RNNs have in general to do mainly with an improved memory system, the boost in performance in this case is due to a certain emergent property of the combination of LSTMs with tree topologies: due to the adaptively learned input and output gates at each node, the network can learn to “ignore” certain constituents and pay more attention to others (see Zhu et al, Fig. 1). This is potentially useful, since for example the head verb in a complex verb phrase may in many cases be more informative about sentence meaning than its sibling.

Next, the model in (Socher et al 2010) begins with the “greedy” approach to parse tree discovery of (Socher et al 2011c), but rather than letting reconstruction error serve as the learning signal for parsing decisions, it computes a “score” for each candidate phrase node, and learns weights so that high scores are assigned to correct merger decisions (i.e. examples of correct parses are used in a form of supervised learning).
The authors test this model, as well as various modifications of it that increase performance, and achieve precision and recall close to the then-state-of-the-art version of the Stanford parser (Klein and Manning 2003) on sentences from the Wall Street Journal. Unfortunately, the basic model is already less psychologically plausible (due to the tree-supervised learning) than that in (Socher et al 2011c), and each modification, while boosting performance, moves the models even further from psychological reality, by using explicit syntactic and semantic category labels, as well as global search techniques in the space of all possible trees instead of greedy approximations.

I have introduced this model not because it is particularly competitive with the other approaches discussed in this section in terms of empirical results on NLP tasks, but because its generalization in (Socher et al 2011b) has the interesting property of being (successfully) applicable equally to the “parsing” of images and of natural language, and is thus powerful in a different way. This model differs from the one in (Socher et al 2010) mainly in that, instead of taking word sequences as inputs, it takes either word sequences or (pre-processed) images as input, together with an “adjacency matrix” that tells the network which elements of the input are adjacent to one another (this information is trivial to derive from input sentences but not from images, since a “sliding window” approach cannot be used for irregularly shaped 2D elements).

Just as in (Socher et al 2010), the network attempts mergers of every pair of adjacent input elements to derive a parent-node representation, whose score is computed. Learning proceeds in a supervised way via backpropagation from the true nodes in the relevant tree structures, to maximize the scores of accurate parse decisions. In the case of visual input, the input images are pre-segmented into broad regions, and an accurate parse is one that assigns sub-regions to the correct broader region (for example, assigning all the visual parts of a church to the same higher-level “church” node). Because the network constructs binary trees but the
order in which sub-regions of an image are merged is arbitrary, there will usually be more than one valid “parse” for a given image.

Using this model, Socher et al achieve 78.1% accuracy in labeling each pixel in input images with its corresponding class label (e.g. a label for which object in an image a pixel belongs to—here it is assumed, as is true in almost all cases, that each pixel belongs to only one object). This approach beat its contemporary competitors on this task by a few percentage points. The same network also performs well (though not quite as well as benchmark systems) at discovering the correct parse for English sentences (see Socher et al 2011b, §6.4 for details), and moreover does so on the basis of limited information compared with standard approaches. While the network is thus so-so as a language parser, the vision task is a rather challenging one in computer vision: whereas most widely publicized contemporary achievements of deep learning in computer vision concern only an image labeling task (e.g. assign the single discrete category to an input image that best fits it, or a distribution over such categories), the “scene understanding” task attempted here requires the network to recognize multiple object instances, as well as to segment the image into regions corresponding to each, down to the pixel level.

For now-familiar reasons, I think we should ultimately look to different approaches to learning if we wish to model psychological processes using connectionist nets—in addition to relying on pre-specified trees for training as in (Socher et al 2011c), this model also relies on pre-segmented images. The latter is not a serious worry: a module could be swapped in that can learn to segment images in an unsupervised way (e.g. Carson et al 2002), and in more realistic models, off-the-shelf, hand-engineered features could be dispensed with. But my interest in this kind of model, in any case, mainly lies in its proof that a domain-general connectionist processor applicable both to language and vision is possible—a thread that I’ll take up in the next section.
Apart from its intrinsic interest and for the concreteness it lends to the discussion, I have indulged in the foregoing feast of model details in order to offer an abductive argument, of sorts, for the overall type of connectionist solution to the problem of systematicity that is slowly coming into focus: a non-Classical approach to compositionality based on (a) distributed vector space representations, in which (b) position in the vector space (i.e. the internal structure of the vector representations) maps onto semantic relatedness, and in which (c) principles of composition can be at once rule-like and empirically informed. The argument in question is simply that some of the best-performing parsers in the world today (the RNTN and tree-shaped LSTM in particular) are direct conceptual descendents of the humble RAAM system proposed by Pollack (1990),\textsuperscript{106} in direct response to Fodor and Pylyshyn's systematicity challenge.

I have, however, thanks to a kind of puritanism about unsupervised learning, found these models inadequate as solutions to the challenge by my own lights. This means that the empirical argument that I'm suggesting to be implicit in the foregoing sections ought to be far more compelling to typical proponents of Classical architectures than it is to me, since the typical Classicist places less weight than I do on the learnability of representational structure via connectionist algorithms. Insofar as the issue is just whether a connectionist architecture can, in virtue of the structure (or lack thereof) of its representations, give rise to systematicity, and necessarily so, the answer seems to be a resounding “almost certainly”.

But for my money, this connectionist explanation of systematicity is incomplete if it succeeds only in an unrealistic learning regime. In the remaining sections of this chapter I'll look at an approach to connectionist parsing that does not require tree structures to be supplied

\textsuperscript{106} “Humble” because its typical number of parameters could probably be counted in the hundreds or thousands, rather than millions.
I begin with an extant model and then consider, somewhat speculatively, a variant that takes advantage of recent developments in connectionist theory. Curiously, this approach, like the one just discussed, involves the transposition of techniques typically used for computer vision to the domain of natural language processing. These models suggest that domain-general representational systems may be responsible for natural language processing, and so are of course friendly to the thesis that language-like structure presents no specific challenge for connectionism.

4.4 Applying computer vision techniques to language

Two perennial debates in cognitive-scientific circles concern the domain-generality (or lack thereof) of cognitive mechanisms and their innateness. Carruthers (2006) argues for a “massively modular” approach to the mind, according to which brains enable intelligent behavior by implementing a large collection of evolutionarily endowed special-purpose systems. This is in a way a radical extension of the basic modularity thesis espoused in (Fodor 1983). This theory of mind sits well with the Classical appeal to a level of analysis at which mental representations can be treated as language-like formal structures, transformed according to a delimited set of rules that operate largely independently of their semantic interpretations.

The tack taken by contemporary connectionists is, not surprisingly, about 180° removed from this view, at least in some ways. Here is a quote from (Hinton 2005) that I find compelling:

\[ \text{[quote from (Hinton 2005)]} \]

---

107 See also the “skip-thoughts” model (Kiros et al 2015) discussed in 6.3 below. I do not include this model here because it is not a parser per se, but it may provide a principled approach to the wholly unsupervised learning of a vector composition function.

108 It should be noted, however, that Fodor’s modularity thesis was offered as a way of contrasting those aspects of the mind that can be usefully characterized as modules with the more theoretically intractable “general cognition” that looms in the background, and is the locus of the most impressive features of human cognition. There is perhaps some tension between this perspective and the tight relationship between syntax and the systematicity of inference posited in (Fodor and Pylyshyn 1988).
Our perceptual systems make sense of the visual input using a neural network that contains about $10^{13}$ synapses. There has been much debate about whether our perceptual abilities should be attributed to a few million generations of blind evolution or to a few hundred million seconds of visual experience. Evolutionary search suffers from an information bottleneck because fitness is a scalar, so my bet is that the main contribution of evolution was to endow us with a learning algorithm that could make use of high-dimensional gradient vectors. These vectors provide millions of bits of information every second thus allowing us to perform a much larger search in one lifetime than evolution could perform in our entire evolutionary history.

This argument is offered not in the service of some sort of radical, blank-slate empiricism, but rather to motivate the view that we should expect to find powerful domain-general learning principles that let experience do most of the work. The discovery of a single style of neural network-based learning applicable both to natural language processing and to sensory processing more generally is thus desirable not just on purely a priori grounds of parsimony, but because it constitutes a step in this direction.\footnote{Relatedly, in a series of meta-analyses, (Hamrick et al 2018) find that both first and second language abilities seem to rely on the same domain-general learning mechanisms.}

A major success story of 21st-century connectionism, one that restored the place of artificial neural networks in the mainstream of applied AI after decades of relative marginality, was the use of convolutional neural networks (CNNs—see LeCun and Bengio 1995) to solve problems in computer vision. CNNs differ from vanilla feedforward networks in that they involve the application of the same set of small filters (i.e. synaptic weight matrices, usually followed by a nonlinearity) to small patches of the input image. This architecture was inspired by the discovery of cells that act as oriented gradient filters, replicated across space, in early vision.

In addition to this biological motivation, the CNN architecture reduces the number of parameters that need to be learned for a task (since the weights for each filter are shared across its replicated instances), and builds a useful a priori assumption about the structure of the input data into the architecture (namely, that similar visual features may appear in many
different spatial positions within images). This architecture is referred to as “convolutional” because the replication of the same filter across image patches is equivalent to the mathematical operation of convolving the filter with the image (considered as a function from pixel position to pixel intensity, potentially in higher dimensions for color images).

Very deep (multi-layer) CNNs trained on huge amounts of data advanced the state of the art in computer vision starting around 2012 (Krizhevsky et al 2012), and remain the preferred method in that field. Importantly, in addition to their depth, effective CNN models are also “wide”: they make use of many different replicated feature detectors at each layer. Typically, these feature detectors “see” each of the three color channels in a color image. The resulting filters, which in standard approaches are learned from data in a supervised way, end up capturing features like oriented edges, textures, and simple curves, and higher-level features like faces in deeper layers. Some filters (without prior programming to this end) wind up being “color-agnostic”, while others respond selectively to patterns in a given color channel (see again Krizhevsky et al 2012). Dimensionality reduction is typically achieved in CNNs via special-purpose “max pooling” layers that pass forward only the maximum value in each cell of a grid overlaid on the feature map at a given layer.

In (Kalchbrenner et al 2014), this approach to computer vision is deployed on a natural language processing task, resulting in a network that induces “a structured feature graph over the input sentence” (p.656). The authors use this network to obtain then-state-of-the-art results on sentiment analysis and question classification tasks, again (as in the case of Collobert et al 2011, discussed earlier in §3.5) beating out systems that rely on hand-engineered features. Importantly, these tasks involve classifying entire input sentences.

---

110 See Kalchbrenner et al (unpublished) for a very impressive large-scale model that is also based on CNNs, and yields state-of-the-art results on character-level language modeling in the context of translation.
where long-range dependencies are often important for classification, so the success of the model demonstrates that decent sentence-level representations have been learned.

It is useful to consider first at a relatively abstract level how the machinery of CNNs is applied to natural language in this model. First, the sentences are represented at the input layer of the network by combining vector space embeddings for the words that occur in them. Specifically, sentences are represented as $N \times M$ matrices, where $N$ is the dimensionality of the word embeddings, and $M$ is the number of words in the sentence. Thus, the input sentence representations are like the input images fed to vision networks: the values of the individual entries of the word vectors are treated in effect as pixels, with the semantically significant dimensions of the vector embeddings treated like color channels.

In this context, sliding a filter over the input amounts to computing a feature for every $n$-gram in the sentence, where $n$ is the size of the convolutional filter. Although the authors do not discuss the following possibility, this way of representing and processing sentences builds in the potential to learn higher-level features for $n$-grams that are sensitive to particular dimensions in the input word vectors (i.e. “semantic” filters), as well as filters that respond to patterns across many dimensions (“syntactic” filters).

The model does not make use of an explicit unsupervised learning procedure to extract word vector embeddings, and instead relies on backpropagation from the sentiment analysis task, through the other parameters of the network, to the word embeddings, beginning from a random initialization. Nonetheless, the network’s performance suggests that the embeddings learned in this way were successful (as were those learned similarly in (Milokov et al 2013a), which initially inspired the later unsupervised approaches such as skip-gram).

---

111 Somewhat confusingly, the authors list the feature representations in their networks as “unsupervised vectors” (p.662). I take it that they mean to contrast their system, which learns its own representations to suit the demands of the task, from other systems that use pre-engineered feature representations.
Like standard convolutional networks for vision, the network in (Kalchbrenner et al 2014), which the authors call a “Dynamic Convolutional Neural Network” or “DCNN”, uses pooling layers between successive convolutional layers, together with a fully-connected layer at the top that “sees” information from the entire layer below. Unlike typical models used for vision, however, the authors employ a $k$-max pooling layer that selects the $k$ most highly-activated features at each layer and feeds them to the next layer (where $k$ may vary depending on other parameters such as input layer size—see §3.3 of the paper).

Local connectivity is of course a reasonable constraint in vision networks: the max pooling operation in effect feeds forward the most salient visual features at a given location in the image. In the context of NLP, since long-range dependencies may matter, this constraint would be an obstacle. The $k$-max pooling operation avoids this while still achieving dimensionality reduction by selecting only the most salient features active across the whole sentence representation that it receives as input.

Note that the entire sentence is represented at each layer, first in terms of its word vectors, then in terms of the most salient $n$-grams, then in terms of the most salient collections of $n$-grams, and so on up to the fully-connected layer at the top, with single nodes (within each dimension) in higher layers representing phrases at the layer below. The network thus implements a form of compositionality (in the sense discussed in §3.6) for VSMs. Given this, it is perhaps not surprising that it ends up constructing, for each input sentence, parse-tree-like representations whose leaves are the word embeddings and whose root is the single vector at the fully-connected layer at the top.

It is worth dwelling for a moment on the features of these trees. Though, as I have argued, the nodes in such trees can be expected to capture both strictly syntactic information and shades of meaning, the structures induced are not $a$ priori constrained to resemble parse
trees like those posited in generative grammars. Among other things, while the DCNN’s trees will most likely have structural features in common with such standard parse trees, there is an important difference: because of the multidimensional way in which input words are represented, the parse trees themselves are multidimensional. In the setup used in the DCNN, the sub-trees in each dimension are joined before the final $k$-max pooling layer using a “folding” operation that simply sums the vectors in adjacent rows to further reduce dimensionality. Without this operation, the subtrees would be joined only at the root node (see Kalchbrenner et al 2014, p.660).

Although the DCNN is tested only on relatively simple categorization tasks (rather than more demanding syntactic labeling or parsing tasks) and although its tree structures bear only a loose resemblance to Classical syntax trees, it constitutes an important proof-of-concept for the thesis that structured representations can be not only implemented in connectionist networks of a rather standard kind, but learned “from scratch”, without appeal to implausible brute-force learning mechanisms (though in this case, via supervision from a downstream task objective).

4.5 Hinton’s critique of CNNs

The network just considered is a striking illustration of the erosion of lines between language processing and sensory processing enabled in some connectionist models. Apart from the global as opposed to local character of the pooling operation and the optional “folding” layer in the latter model, CNNs induce the very same sorts of tree structures over their input images in the context of computer vision models used for object recognition. The success of

---

112 The points made here mostly compare vision and natural language processing (where the aspects of the latter that depend on vision, such as visual character recognition, are abstracted away—presumably, the NLP networks under consideration model a downstream process that may also take audition as input). I assume there are no serious obstacles to extending this approach to other modalities.
this model indicates that applying computer vision techniques to NLP may be not only a good idea from the point of view of theoretical psychology, but an effective approach in practice.

In this section, I’ll consider some rather abstract reasons to suppose that despite this, CNNs in particular may not be the best available approach within the family of connectionist models. In fact, the drawbacks of CNNs to be canvassed here, pointed out by Geoff Hinton and colleagues in a diverse series of publications (Hinton et al 2010, Hinton et al 2011, Sabour et al 2017), do not concern NLP in particular, but apply to CNNs across the board. These considerations motivate a new approach to connectionism that makes the role of distributed representations in facilitating the construction of parse-tree-like structures transparent. I’ll discuss this approach in the following section.

First, consider that connectionist networks are graphical models of a sort. In many cases they are Bayesian networks, i.e. directed, acyclic graphs (DAGs) with associated marginal and conditional probabilities (Pearl 1988). As discussed in (Williams 2018, §4.2), graphical models allow for a kind of compositionality. A graphical model with $n$ binary-valued nodes representing propositions, for example, can represent (and assign probability to) $2^n$ possible situations. More structured models allow for more specific types of composition. For example, in the Helmholtz machine (Dayan et al 1995, Hinton et al 1995), unsupervised learning of the weights leads to progressively more abstract representations as computational distance from the input layer increases. The Helmholtz machine implements a stochastic generative model in its top-down weights, wherein hierarchical representations can be constructed by sampling states for high-level parent nodes (given their marginal probabilities), sampling states for children based on the states of the parents, and so on down to the bottom level of the model.

---

113 This interpretation of the nodes is employed, for example, in (Hinton and Sejnowski 1983), though that model is not a DAG.
(such representations can also be inferred from data via the bottom-up connections). The generative and bottom-up models are both (in this case stochastic) DAGs.

But as (Hinton et al 2000) point out, in general, DAGs are a poor model of image-generating processes, because multiple parent nodes may jointly influence children, but (assuming non-transparent objects), each image region should be assigned to at most one “cause” (i.e. one “parent” object of which the region is a part). For the same reason, they may be expected to fail as generative language models (since each constituent should be parsed as belonging to exactly one constituent at the next-highest level of hierarchy). Note that the CNN model of (Kalchbrenner et al 2014) doesn't implement a single-parent constraint, nor is it guaranteed to deliver only parse trees licensed by a standard generative grammar.

Further, as (Hinton et al 2011) note, the representations employed by CNNs, particularly at the “max pooling” layer, are not ideal as models of object recognition in human brains. This is because they attain abstraction at a price: higher-level representations of object categories lose information about the fine-grained features represented lower in the hierarchy. The standard CNN represents objects at the category level via a single real-valued node that represents the presence, or not, of a class instance in an image. This representational scheme seems psychologically implausible, given that object recognition presumably relies on much richer high-level representations.

This representational feature of CNNs also entails a serious inefficiency. CNNs (and simple neural network models in general) handle abstraction by learning single higher-level nodes that respond to particular patterns of node activities in lower layers, akin to the receptive fields of individual biological neurons. Higher-level nodes respond to patterns in the first-order pattern detectors, and so on. But since each pattern of pixels corresponds to just a single viewpoint on any given object, the same will be true of the higher-level “abstract”
representations. By contrast, a single representation for an abstract category, such as the concept CAR, should apply equally to cars perceived from various viewpoints.\textsuperscript{114}

It is still possible to derive truly abstract representations with CNNs (otherwise they wouldn’t work), but this is accomplished only by learning very high-level units that respond indifferently to related features active anywhere in the layers below. In most CNN architectures including AlexNet (which represented one of the first “breakthrough” moments for CNNs—see Krizhevsky et al 2012), the final object category labels are the outputs of fully connected layers at the top of the processing hierarchy, which “see” the entire input image and respond just in case a suitable collection of high-level features have been passed through the “max pooling” layer below. Thus, the object recognition strategy implicit in the average CNN is something along the lines of the following: \textit{If many cat-diagnostic features are detected at once with high confidence, label the image ‘cat’}.\textsuperscript{115} There is thus a danger that the success of CNNs at their appointed task depends on the fortuitous exploitation of a domain regularity that admits of exceptions: typically, any image that includes sufficiently many cat-identifying features is an image of one or more cats. The network needn’t (usually) explicitly register the precise higher-order relations among such features in order to get the correct answer.\textsuperscript{116}

The apparent inability of (at least many) CNNs to capture information about the \textit{number} of objects of a given type in an image (see Karpathy 2016 p.343) suggests that they might not be constructing object representations in anything much like the way humans do. A more vivid illustration of this limitation is available in the kinds of images that CNNs yield when “run in

\textsuperscript{114} The same applies to abstract category representations proprietary to a given sensory modality, in case one distinguishes these sharply from concepts proper.

\textsuperscript{115} This isn’t quite fair: distinct high-level categories may share low-level ones like eyes, fur, etc, and some features such as dog ears may be diagnostic for non-cathood.

\textsuperscript{116} It may be argued that this sort of heuristic is the bread and butter of statistical methods like those used in machine learning and computer vision, but there is a difference between the epistemic risk inherent in any statistical inference and a case like this in which there is a clear qualitative gap between the evidence exploited by the algorithm and that to which human judgment is responsive. However, what makes up that gap may well be more statistical information of a different sort, as argued in the ensuing paragraph.
reverse” and used as generative models. As is by now notorious, such images (produced by activating a high-level category node and backpropagating all the way to the image pixels) resemble hallucinations in which local visual features of objects (most saliently, eyes, noses, and other animal parts) are accurately rendered, but with only scant regard for their higher-order statistics. For example, beginning with an arbitrary image as a “seed” and backpropagating from the “dog” category label will result in an image with the low-spatial-frequency profile of the input image, but constructed, in effect, from dog parts. Thus, despite the ability of CNNs to learn a kind of hierarchically organized representational scheme, accurate categorization seems to be driven disproportionately by the recognition of conjunctions of specific low-level features.

Finally, note that since CNNs employ layers of scalar-valued feature detectors, the representations of individual words and n-grams in each dimension of the trees constructed by the DCNN model of (Kalchbrenner et al 2014) are “localist”, single-node representations rather than distributed representations. The concatenation of the nodes at each layer may be viewed as a distributed representation, but this representation is of the whole sentence at each layer, and while each layer also has a “depth” dimension corresponding to the dimensions of the input word embedding vectors, these representations are not tied across dimensions. Thus, nodes in the constructed “syntactic” trees are useful only because of the functional role they play in the broader network. They do not, individually, contain fine-grained information about the lower-level constituents they govern—that is, in the terms of (Hinton 1981), they possess no “direct content” (cf. §1.2 above).

117 For details, see the post by the Google AI research group at https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html (accessed 8.15.2018).
4.6 The capsules approach

As argued in (Hinton et al 2000), by restricting the space of DAGs actually selected for the interpretation of any given input, parse trees can be “carved out” of the nodes activated in a DAG (Hinton et al 2000). As Hinton puts it (Hinton et al 2000, §3), for the case of image processing: “Given a DAG, the possible parse trees of an image are constrained to be individual [trees] or collections of trees where each unit satisfies the single-parent constraint, with the leaves being the pixels of an image.” The result of processing will thus be a parse tree proper. Similar reasoning, of course, applies to neural net models of language, where parse trees perhaps enjoy a richer history. I now discuss a proposal for a next-generation type of connectionist network called a “capsules” network (Hinton et al 2011, Sabour et al 2017, Hinton et al 2018). It enforces the single-parent constraint by appeal to distributed representations at each node of a parse tree, solving both problems raised for the CNN-based approaches considered in the previous section at once.

I’ll begin with a brief overview. As defined in (Hinton et al 2011) and elsewhere, a "capsule" is a subset of neurons within a layer that outputs both an "instantiation parameter" indicating whether an entity is present within a limited domain and a vector of "pose parameters" specifying the pose of the entity relative to a canonical version. The parameters output by low-level capsules are converted into predictions for the pose of the entities represented by higher-level capsules, which are activated if the predictions agree, and subsequently output their own parameters (the higher-level pose parameters being averages of the predictions received). This scheme was designed specifically to explain how parts could be intelligently associated with wholes in part-whole hierarchies within a connectionist network.

The “capsules” approach (see Figure 8) groups layers not into nodes with activities represented by scalar values, but into mini-modules called “capsules”, which yield vector-valued
or matrix-valued (see Hinton et al 2018) outputs (and in most cases take vector-valued inputs as well). Weight matrices play the role of individual weights between nodes in ordinary neural nets.

Figure 8: Bottom: schematic layout of one type of capsules network. Boxes with circles indicate vector-valued processing units (capsules) and shaded regions represent weight matrices connecting capsules in distinct layers. As discussed in (Sabour et al 2017), capsule dimensionality increases in each layer in the model shown here, while the number of capsules per layer decreases. Top (left): an ordinary neural network whose structure corresponds to that of the capsules network below, where circles are scalar-valued nodes and edges are single weights. Inset: two ways of representing existence with capsules. Inset top: capsules (a) and (b) have pose vectors with similar orientations but different lengths (darker circles indicate higher scalar unit activities). In the representation of (Sabour et al 2017), capsules (a) and (b) describe the same entity, but capsule (a) represents it as more likely to exist. Inset bottom (follows Fig. 1 in Hinton et al 2011): a separate logistic unit “p” represents the probability that the entity described by the pose vector exists. Its value is used to multiplicatively gate the capsule’s output.
This is a natural generalization of the standard connectionist/deep learning paradigm. We may think of a capsules network as identical in form to a traditional connectionist network at a coarse level of description, since, like the nodes in a traditional network, each capsule’s input and output is a function of its local connectivity. However, vector-valued “nodes” can obviously encode much more information than do binary or real-valued nodes.

Thus, one of the problems posed for CNNs above is solved simply via this representational choice: rather than a single scalar value to indicate the existence (or probability) of a highly specific feature in the input, a capsule can represent the existence of an entity (by its activation), and the specific features of the object to be represented (by its vector of “pose” parameters), yielding a single representation whose properties covary with those of the particular instance represented. Efficiency is also facilitated because the invariant knowledge of part-whole relationships for an entire class of entities is encoded in a single weight matrix (Hinton et al 2011). In addition to yielding far more flexible representations, Sabour et al (2017, p.9) claim that this proposal “eliminates the binding problem”.

The difference between capsules nets and standard connectionist nets shouldn’t be exaggerated: in many ways, a layer of $N$ concatenated capsules with $M$ nodes each is identical to a standard layer with $N \times M$ nodes, and capsules may be viewed as special cases of distributed representations. However, while in ordinary networks, active units simply broadcast their states indiscriminately to every higher-level unit to which they’re connected, capsules networks employ a procedure that exploits the distributed representations at each capsule, together with top-down feedback, to more intelligently route information.

---

118 This, at least, in the model of (Sabour et al 2017), where the magnitude of the vector of pose parameters represents the probability that the associated entity exists. (Hinton et al 2018) abandon this elegant idea on the grounds that it leads to an ad hoc cost function, and revert to the proposal in (Hinton et al 2011) in which existence is represented via a separate logistic unit (see Fig. 8 inset above).
The most natural way to enforce the “single-parent” constraint in a vanilla connectionist net would be to include a mechanism for competition among the nodes at any layer, such as inhibitory within-layer connections. A much more sophisticated method is available when the nodes are vector-valued, however: as discussed in (Hinton et al 2011), (Sabour et al 2017) and (Hinton et al 2018), measures of vector similarity (or clustering algorithms) can be used to ensure that a capsule corresponding to a higher-level entity in Layer 2 receives activity only from lower-level capsules in Layer 1 whose predictions for the “pose” of the higher-level entity match (see Sabour et al 2017 and Hinton et al 2018 for two different ways in which such “dynamic routing” may be implemented). This may be expected to enforce the single-parent constraint because high-dimensional coincidences are very unlikely (Hinton et al 2018, p.1), so each “child” node in layer $L$ should have at most one “parent” node in $L+1$. The dynamic routing proposal is inspired by the model in (Olshausen et al 1993), which focuses on somewhat different aspects of the idea, including its biological plausibility.

A capsules architecture also points the way toward a solution to the dilemma concerning abstraction mentioned earlier. As discussed previously, one reasonable use of hierarchical representations in neural networks is to enable dimensionality reduction. In autoencoders, for example (Hinton and Salakhutdinov 2006), the dimensionality of layers typically decreases as the processing hierarchy is ascended. Capsules networks can retain this feature without sacrificing fine-grained information about the poses of visual entities by dividing layers into progressively smaller collections of higher-dimensional capsules. In connection with this, consider the following, from (Sabour et al 2017, p.2):

For low level capsules, location information is “place-coded” by which capsule is active. As we ascend the hierarchy, more and more of the positional information is “rate-coded” in the real-valued components of the output vector of a capsule. This shift from place-coding to rate-coding combined with the fact that higher-level capsules represent more complex entities with more degrees of freedom suggests that the dimensionality of capsules should increase as we ascend the hierarchy.
We then have a coarse-grained system of abstract representations in effect layered over more fine-grained representations of precise object features. Here, the representations increase in dimensionality as the hierarchy of representational layers is ascended. However, there is still a mechanism for enforcing dimensionality reduction: in effect, there is sparse coding at the level of groups of neuronal units. This layering is what “eliminates the binding problem”. Sabour et al (2017) and (Hinton et al 2011) note that this encoding has the consequence that only one entity of a given type can be represented at a precise location in the input array at a time, but argue that human perceptual systems actually exhibit this limitation (cf. Pelli and Tillman 2008).

So far I’ve been describing the virtues of capsules systems in a way intended to be agnostic as to their domain of application, but the networks were devised mainly as replacements for CNNs, and as such are mainly intended as tools for computer vision. In the remainder of this section I’ll briefly discuss some of their features as general computer vision models, consider their potential (and so far barely explored)\(^\text{119}\) use in natural language processing, and briefly consider differences between the two domains.

Beginning with a standard input layer for computer vision, it is shown in (Hinton et al 2011) how first-layer capsules that represent basic pose parameters (e.g. position, scale, skew, rotation, etc) of visual entities can be learned in an unsupervised way from pixel intensities. From there, higher-level capsules can be activated that correspond to larger visual entities of which the entities represented at lower levels are parts. The weight matrices that map lower-level to higher-level capsules constitute, in the case of vision, transformations that map the specific features of the low-level entities to specific features of higher-level ones: for example, an eye detected at a certain location with a certain orientation is translated into a prediction for a face (in the same location) with a corresponding orientation.

\(^{119}\) Though see (Wang et al 2018, Wedajo et al 2018) for recent applications to sentiment analysis.
Although not generative models themselves, capsules networks for vision are importantly related to the rationale for generative models like the Helmholtz machine (Hinton et al 1995, Frey et al 1997): the idea of factoring object representations into, (a), representations of invariant relationships between objects and their parts and, (b), representations of object features that vary with viewpoint, was, like the Helmholtz machine, in part inspired by the idea that computer vision is essentially “inverse computer graphics”. Typically, images of visual scenes are rendered on the basis of viewpoint-invariant, hierarchical representations of objects. And the efficiency of capsules representations viz-a-viz CNNs sits well with the idea that efficiency should in general be an important constraint on mental representation.

Research on capsules is still in its infancy, and empirical results are accordingly limited. However, the convolutional capsules network proposed in (Hinton et al 2018) performs twice as well as the best-performing extant CNN on the smallNORB shape recognition database, using 15 times fewer parameters (2.7 million → 310 thousand). The (likewise convolutional) model described in (Sabour et al 2017) also performs well on the very challenging task of segmenting highly overlapping digits (see §6 of that paper).

What about the case of NLP? Although I have ruled it out of court for relying too heavily on an oracle that provides tree structures, it is useful to compare the capsules proposal to the compositional vector grammar (CVG) discussed in (Socher et al 2013a). There, each word is, again, represented by a label for its syntactic category as well as a VSM vector that captures fine-grained semantic information. The nodes in the parse trees computed by a CVG have the conventional phrase tags associated with generative linguistics (NP, PP, etc), but are also each

---

120 This idea, along with some of the criticisms of CNNs listed here, is explored in a video lecture of Hinton’s at https://www.youtube.com/watch?v=rTawFwUvnLE&feature=youtu.be (accessed 8.15.2018). See also the early proposal in (Zemel et al 1990).
121 Indeed, the proposal for dynamic routing in (Hinton et al 2018) makes use of the Minimum Description Length principle, in place of the dot-product-based scheme in (Sabour et al 2017).
122 One might almost argue too well. A casual look at the examples suggests that many human beings could not perform this task nearly as well as the network did.
associated with a vector. The more purely connectionist approaches to recursive networks discussed above are very similar to this, but they do not represent syntactic categories explicitly, relying on the ability of VSMs to capture syntactic as well as semantic regularities. Capsules might provide a middle road, in that distinct capsules could keep track of distinct syntactic categories while retaining rich semantic information at each node, and yet trees could be constructed without explicit supervision. Capsules thus may afford a realistic, purely unsupervised approach to parsing—but this idea, as it stands, remains speculative.

Despite what the domains have in common, there are differences between visual “scene parsing” and NLP that might call into question the wisdom of applying a capsules architecture in the latter case. One is that there are, intuitively at least, many more types of object than there are types of phrase (at least, as represented in most theories of grammar). There are also fewer dimensions of variation in the pose of a visual object than there are dimensions of variation in meaning of a term or phrase. If words are assumed to be represented by high-dimensional vector space embeddings, then words, as well as phrases and sentences, are certainly “more complex entities with more degrees of freedom” relative to most visual objects. It is possible, then, that natural language processing begins where the capsules architecture ends: by the time words are represented as such, we may already be dealing with representations in which most information is “rate-coded” rather than “place-coded”.

A second, related difference is that, while sentences are plausibly processed one word at a time (with possible exceptions for common, overlearned phrases, which might be represented as Gestalts), freeing the lower-level nodes up for representation of new words at each step, visual scenes involve the simultaneous processing of information across the entire visual field. The apparent difference here between vision and natural language might be narrowed by appeal to the fact that vision actually involves sequential foveation-based sampling
(see the intro to Sabour et al 2017), but apart from sheer biological plausibility, it would make sense for vision to use replicated feature detectors and a kind of “place coding” at early stages. It’s possible that NLP could make limited use of this feature of capsules as well, for example by using peripheral vision to take contextual information into account when making parsing decisions (see e.g. the “context-sensitive” variants of the model in Socher et al 2010).

Despite these caveats, there is an indirect reason for supposing that capsules architectures might model natural language processing well, if they are good models of vision: if there is indeed a single, domain-general process for unsupervised representation learning (Bengio et al 2012), then (given that it is otherwise biologically plausible, e.g. that neurons might plausibly organize themselves into capsules-like representations) it is better to posit the more powerful vector-valued nodes of capsules networks across the board.

Before moving on, I’ll revisit a pivotal question posed in §1.2 above: do the hierarchically organized representations in neural networks like those I’ve been considering function in a way that licenses the description of them as akin (functionally, not just superficially) to syntax trees? That depends on how it is that syntax trees are meant to function in processing. That cognitive processes are sensitive to the explicit syntactic structure of mental representations is of course one of the defining commitments of Classical architectures.

In general, the structure of the trees computed in these models, whether spatially extended, relatively enduring structures, as in the capsules proposal and the DCNN model of (Kalchbrenner et al 2014), or structures implicit in the transitions between processor states, as in the proposals based on recurrence, do not seem to be implicated in processing except insofar as they serve to compute rich vector representations at the root node. In the DCNN, for example, the word vectors, n-gram vectors, and higher-order n-gram vectors (e.g. n-grams of lower-level n-gram features, corresponding to more abstract syntactic and semantic categories)
are not used except to compute the final sentence-level vector representation fed into the classification algorithm. And it has been one of my aims in the discussion so far to argue that explanations of higher cognitive function needn’t after all appeal to such explicit structure in sentence-level representations. That said, a connectionist cognitive architecture is at liberty to use any constituent representations computed along the way to the sentence level, and the phrase-level node vectors computed by the tensor network in (Socher et al 2013b) are used by the authors to perform a fine-grained sentiment analysis task.

In the next chapter, I abandon the topic of parsing in neural nets specifically, and consider closely related connectionist models of other high-level cognitive functions. I argue that these models point again to a uniform computational basis for natural language comprehension and perception more generally. I begin, however, with some highly abstract reflections on how concepts, propositional representations and various processes defined on them may be mapped onto aspects of vector space representations, against the backdrop of the survey of neural network-based approaches to vector compositionality just completed.
5.1 Concepts and propositions

The discussion of vector composition methods at the end of Chapter 3 bordered on philosophical deep waters, which are the subject of this section and the next. To start, in §3.6, the idea was briefly aired of deriving phrase-level vector space representations using a composition function that maps input representations to output representations in a different vector space. This is in some ways the more natural representation to use if one is thinking Classically, since representations of sentence-level and of word-level contents would be kept cleanly separated. But there are powerful considerations that tell against this approach.

One is that the problem of exploding dimensionality renders systems like the tensor product impractical, as already discussed. One may perhaps want distinct spaces for words and sentences, but tensor products would yield distinct spaces for each level of depth in a parse tree. Apart from sheer capacity concerns, and ignoring the ad hoc nature of the fixes proposed to this system to allow iterated composition (Smolensky and Legendre 2006), this seems a terribly inefficient way of representing meaning, even if we don’t worry about recoverability constraints. The dimensionality of the space would depend on the deepest tree the system is capable of processing, which means that most sentences would be represented using the same small fraction of the available neurons. Moreover, if we want in the end to argue that composite vector representations are computed by RNNs (as seems appealing given the need for recursion, as well as the work surveyed in the previous chapter, and further research to be discussed below), the output vectors must be of the same dimensionality as the input vectors.\footnote{To require precisely the same dimensionality across all inputs and outputs may be to build an artifact of computational modeling into one’s cognitive theory. It’s certainly possible that only a somewhat variable portion of available neuronal resources, even within some target population, are recruited at any}

\footnote{To require precisely the same dimensionality across all inputs and outputs may be to build an artifact of computational modeling into one’s cognitive theory. It’s certainly possible that only a somewhat variable portion of available neuronal resources, even within some target population, are recruited at any}
The idea that sentence- and word-level representations exist in the same vector space has, however, a number of interesting and *prima facie* untoward philosophical consequences, many of which may seem at first glance so absurd as to rule any such approach to mental representation out of court. In this section I’ll discuss these rather abstract issues, and argue that the philosophy of mind enforced by the type of connectionist cognitive architecture in question is, on the contrary, in fact defensible and independently motivated.

One feature of the kind of approach to “compositionality” (I use scare quotes to appease hardline Classicists) under discussion is that it does not result in contributing vectors being literal constituents of a resultant phrase-level vector, for the same reason that the superposed vectors in Smolensky architectures can’t be expected to display the constituents they encode on their sleeves. One might argue that phrase and sentence vectors would contain the vectors for contributing expressions in some mathematical, non-Classical sense, but the considerations about unique decomposition voiced in (Fodor and McLaughlin 1995—see §2.3 above) seem pretty clearly to scuttle any meaningful algebraic notion of constituency. One may as well stick with an obviously true and less controversial description of the mechanism at work: word- and sentence-level representations in a VSM of this sort would exhibit the computed-from relation rather than the constituent-of relation.  This much, I take it, was established in Chapter 2.

Second, and more controversially, the fact that sentence-level, phrase-level and word-level representations in this type of system “live” in the same vector space suggests an erosion of the sharp boundaries normally thought to obtain between the semantics of sentences and the semantics of subsentential expressions (see Baroni 2013, p.512). This distinction has given stage of processing. The heart of the idea however is that the same neuronal substrate (represented abstractly as a vector space) is used to represent simple and arbitrarily complex contents. This may be so even if the effective dimensionality of the relevant vectors fluctuates somewhat.  

124 Of course, as discussed earlier in connection with VSAs, the class of practically relevant decompositions can be delimited by appeal to the condition that input representations be recoverable from the representation of a complex—though in this case there is no generic “unbinding” operation. But in any case, again, computational derivability just isn’t (Classical) constituency.
played an enormously important role in traditional truth-functional approaches to semantics, as Baroni notes. One way of framing the issue: words are typically thought to represent objects, properties and relations, while sentences are thought to represent states of affairs.

In a compositional VSM of the kind under consideration, it is not clear that such distinctions are well-motivated. Of course, given that sentences must be distinguished from words, it must be the case that no vector in the relevant space represents both a word and a sentence. The word-representations, sentence-level representations, and any intermediate phrase-level representations must be distinct. However, it would be a requirement on phrase- and sentence-level representations that they be near the representations of their Classical constituents in the vector space: sentences have meanings that are closely related to those of the words that occur in them, after all (this is, more or less, the whole idea behind compositionality). This suggests that the word and sentence-level representations will not be neatly partitioned into different regions of the vector space. And indeed, as noted by Gayler (2003) and discussed above, vector addition at least is a similarity-preserving operation, resulting in phrase vectors similar to their constituent vectors.

Of course, in some cases the addition of a given word or phrase would decisively alter the meaning of an entire sentence (as in “It is not the case that $p$”). This will change the magnitude of the resulting vector (with respect to the input) along some dimension, but the latter can still be expected to be similar to the input vectors along many other dimensions. It may be argued that this consideration provides a route via which sentence-level and word-level

---

125 This way of putting things is slightly misleading: it could be that words in this framework correspond to Quinean “one-word sentences” (Quine 1960), in which case a word vector could also be said to represent a sentence. But in this case, we’d rather say that words are just one-word sentences. In any case, I trust that my meaning here is clear. See the following section for more on the Quinean idea.

126 A proposal exactly along these lines, for the contribution that ‘not’ makes to a complex expression, is discussed in §6.4 below.
representations could be partitioned in the space: a dimension (or some subset of them) could be reserved for marking this distinction.

This is an intriguing possibility, but unless the difference between word- and sentence-level representations is explicitly encoded in the input or important for the training procedure, it is unclear that such a partitioning would arise. More importantly, representations that differed only on a hypothetical word versus sentence axis would have to share semantic properties along all other dimensions, so the line between sentence-level and word-level representations would be blurred in any case. Since there is no principled distinction in the system under consideration between vectors that represent words and those that represent sentences, any point (or perhaps region—see the next section below) in the vector space may as well be characterized either as a representation with propositional content or as one with word-level content (e.g. as corresponding to a concept, in a familiar sense of that term). 127

This may seem to be decisive grounds for rejection of VSMs of this sort. It may be argued that such representations imply, absurdly, that sentence-level and word-level meanings are interchangeable. But this just gets the metaphysics wrong: dogs are not states of affairs. Moreover, word-level and sentence-level meanings play different roles in psychological processes. It seems obvious, for example, that understanding a sentence is not the same kind of thing as grasping the content of a constituent that is not itself a sentence.

On the contrary, I take the fact that the VSMs under consideration collapse the distinction between sentence-level and word-level meaning to be among their finest features, because it’s independently plausible that in the end, all mental-representational content, at least

127 It is also possible to maintain that while arbitrarily complex phrases are stored in a common vector space, distinct word-level (as well as perhaps character-level and some phrase-level) representations exist en route to this hub, extracted from lower-level perceptual events. Indeed, this is perhaps to be expected on many of the models presently under consideration. This does not significantly alter the picture being painted here, so long as word-level meanings are also represented in the common space.
at the level of amodal, abstract cognition, is propositional. There are several reasons for this, which I’ll attempt to lay out in the remainder of this section.

First, an appeal to intuition: what does “understanding ‘dogs’” (see Fodor and Lepore 2001, p.366) amount to other than being able to recognize dogs, say things about them, reason about them, etc—in short, the same kinds of things that would result from understanding sentences about dogs? Of course, thinking about ‘dogs’, the word, is different from thinking about the sentence ‘Dogs are good’. But this difference concerns the contents of two thoughts, and shouldn’t be confused with the issue at hand, which is the difference between thinking that employs DOGS (the concept) and thinking that employs DOGS ARE GOOD (the thought).

Next, one might appeal to a fallacious argument from authority: there is a long tradition in philosophical semantics (though perhaps one that has had negligible impact on formal semantic theories) according to which judgments or thoughts (“sentence-sized” contents) are the atoms of meaning. Brandom (1998) attributes this view to Kant and Frege, and adopts it himself. Quine (1960) famously argued that the only conceivable evidence for subsentential structure is the set of antecedently assumed truth-values of a speaker’s sentences (similar arguments apply to beliefs, intentions, and thoughts). His notion of the “one-word sentence” is evocative in this context. And Carnap (1947, p.7) asserted that only sentences have “meaning in the strictest sense, a meaning of the highest degree of independence.” Support for the “sentence-first” conception of linguistic meaning can be further adduced by appeal to the work of Zellig Harris (1968), discussed in Chapter 3 (cf. §3.1 and §3.5).

---

128 In the sense of being “sentence-sized”, concerned with selecting among states of affairs in the sense of (Stalnaker 1976), concerned with truth-conditions, etc—not in terms of involving syntactic constituent structure, of course.

129 Fodor would probably have objected here that one should not draw metaphysical conclusions from epistemological premises. But the emphasis should be on conceivable (or possible) evidence. The problem Quine points to is that nothing could determine subsentential structure if the truth-values of sentences don’t.
It’s in any case plausible that mental representation is never only of objects or properties *per se*, but always of facts involving them. What would it be just to represent Douglas, full stop? One represents Douglas, in any actual case, by thinking about him, seeing him, imagining him, and so on, and thinkings, imaginings, and perceivings have propositional contents. Berkeley (1710) may well have endorsed some such view, since he warned against the confusions of supposing that there are purely abstract ideas (he doesn’t even allow that one can represent triangularity without representing a particular triangle, or shape without color, and a concept’s representing a property without its also representing a particular, or the converse, would be very pervasive and deep kinds of abstraction).

Supporting this line of thought is also the old truism that really, reference is something *people* do with words, not something words do of their own accord (*mutatis mutandis* for concepts), whereas *being true* really is plausibly a property of sentences (or at least of utterances). What cognitive science definitely needs is some way of explaining one’s ability to refer to John using the term ‘John’. It is not necessary that this explanation proceed by way of the assumption that a primitive semantic relation of reference obtains between ‘John’ and John, or between JOHN and John. My ability to refer to John could be coextensive with my capacity to represent a large collection of propositions. Importantly, those who push this Quinean line needn’t deny that reference exists, only that it’s primitive or determined in an atomistic way (we can even say that the word ‘John’ refers to John if we like, but this fact will supervene on wider facts about the cognitive system including the reference of other terms).

There is a kink in my exposition of these ideas thus far: one may have supposed that the argument from the common vector space for words and sentences shows only that all content is *indifferently* conceptual or propositional, not that all content is propositional. To this, I will discuss two brief replies before moving on to less abstract matters. First, given that all
contents can be characterized propositionally, and given the independent arguments just
rehearsed in favor of propositional contents, it is tempting simply to reduce conceptual content
to propositional content. A concept, for the purposes of compositionality, is a representation of
a property that combines with other representations of properties and representations of
particulars, and perhaps other ingredients, to yield a propositional content. This role could be
played by other propositions, which do represent particulars (as instantiating properties) and
properties (as instantiated by particulars). It’s not clear why concepts must represent only
properties or only individuals in order to do their jobs. And it’s in any case plausible that we only
ever represent properties-instantiated-by-particulars.

One might object to this view by appeal to reverse compositionality (see §1.1 above). If
(relatively) semantically primitive expressions must contribute their whole meanings to the
semantically complex representations in which they figure, the above account of the relation
between concepts and the propositions whose meanings they determine will not be available.
But in order to motivate the compositionality and systematicity of conceptual abilities (see Evans
1982, §4.3) that’s primarily at issue here, it’s not necessary for the “primitive” expressions to
contribute their whole meanings to the complex. It would be enough if understanding the
complex suffices for one to understand the primitive expressions, and vice-versa (so, for
example, being able to think JOHN LOVES MARY suffices for you to be able to think about
John and about love and about Mary, in other contexts, specifically in other thoughts, should
there be any). This condition seems to be met in VSMs.

Note, however, that it may also be possible to characterize all intentional content as
conceptual. As Baroni (2013, p.517) mentions, it has been proposed that sentences should be
conceived of as essentially equivalent to nouns in their representational properties (there is
something intuitive about this, as reified states of affairs are entities of a sort). Moreover,
thoughts could be viewed as merely particularly complex concepts. In order to flesh out these ideas, however, it’s necessary to return to some relevant details of vector space models.

5.2 Words as regions, concepts as operators

As we’ve seen, in VSMs derived from distributional data, each word corresponds to a point in a semantic vector space, whose coordinates are given either directly by its co-occurrence statistics or by the magnitudes of component vectors in a lower-dimensional space derived from the co-occurrence statistics in some way (via SVD, neural networks, etc). According to the further proposal I’ve just been discussing, more complex phrases up to full sentences will be mapped, via some composition function, to points in the same space.

It was argued in §2.9 that a key difference between VSMs and the more classically based VSA models is that the former make use of continuous “semantic spaces” in which nearby vectors correspond to similar meanings, while the latter in effect store a collection of discrete items in a memory, whose relations to one another only in some cases involve similarity. The idea that proximal vectors have similar meanings has figured prominently in the arguments so far rehearsed, but so far I have not exploited the continuity of the spaces in VSM models. This property, I’ll argue, turns out to be important in understanding how such vector space representations might be deployed in cognitive processes.

First, I will suppose that while words and phrases map precisely onto dimensionless points in a real-valued vector space under idealization, vectors close enough in the space will

---

130 In fact, continuity is not an essential property of such models: raw word count vectors, for instance, could be used to define a VSM using integer-valued vectors. And in fact, the mechanisms I describe in this section could perhaps be implemented in a “quantized” vector space with discrete-valued vectors. The required property is “density”: that the space between word-level representations is also meaningful. However, any workable version of the VSM proposal will appeal to spaces of reduced dimensionality, as argued above, and by far the most natural and effective ways to achieve this result in continuous spaces.
also be close enough in meaning to serve for most purposes. If the ‘horse’ vector is, say, \textbf{horse} = [9, 1.2, -3.4, 0.01, 2], then in moving to coordinates [8.93, 1.1, -3.4, 0.02, 2], I will, I assume, still represent a horse. In fact, there is a stronger reason to allow wider variation in the vector representations corresponding to a given expression. The phrase ‘running horse’, for instance, is a representation of a horse. So the vector computed for this phrase by a system of vector composition ought also to be close in the vector space to \textbf{horse}, along most dimensions.

Indeed, the latter point seems somewhat in tension with the idea that word meanings correspond to points in the space. It is perhaps better to think of words as mapping to extended regions centered on the relevant points. Katrin Erk (2009) develops a proposal along these lines. She first discusses a family of proposals for a specific and local type of composition rule: in contrast to the usual “summation vectors” (i.e. the vectors derived for each word from the distributional data, as I’ve been discussing), “The occurrence vector for a word in a specific sentence is typically computed by combining its summation vector with that of a single context word in the same sentence, for example the direct object of a target verb” (Erk 2009, p.107). Erk then shows that the applicability of certain simple inference rules involving paraphrase\textsuperscript{131} can be modeled if we take the meanings of words to be represented by regions in the VSM space bounded by their possible occurrence vectors.\textsuperscript{132}

It is tempting to identify occurrence vectors with phrase vectors. An occurrence vector for ‘horse’ in the sentence ‘The horse raced past the barn’ would simply be a representation with the content HORSE, inflected to reflect the particular horse discussed in this sentence. But notice that this description at least partly characterizes the content of the sentence as well.

\textsuperscript{131} The inferences she discusses are cases of semantic entailment—see §6.2 below.
\textsuperscript{132} Waskan (2010) questions whether VSM representations will prove of any use in applications that depend on semantic understanding of specific contextual relationships among words in a sentence (there, his target is LSA, but the criticism is equally applicable here). Occurrence vectors may go some way at least toward quelling this worry.
Thus, we might think of the sentence-level content as (represented by) a region centered on a point that lies in the intersection of the regions corresponding to the constituent words.

An apparent problem here is that typically, occurrence vectors are computed as simple sums or averages of the input vectors, whereas, as discussed in the previous section, reasonable sentence-level representations must be sensitive to word order. But I am not primarily interested here in the notion of occurrence vectors. We could just as well define regions corresponding to word meanings in terms of whatever (presumably more complex) composition function is used to construct phrase representations. The meaning of a word would then amount to its contribution to the meanings of the sentences in which it figures, as has long been argued, for example, by proponents of conceptual role semantics (see e.g. Block 1994).

Indeed, the real question at issue is whether it is reasonable to suppose that phrase and sentence vectors will lie within the intersection of their constituent word or phrase regions, consistently with the assumption that syntactically distinct sentences are given different representations even if they contain the same lexical items. I see no a priori obstacle to this, given suitable regions, but I leave this as a topic for future research.

Apart from the issues about representation and compositionality that I’ve just discussed, it is useful to look at this kind of semantic model from the point of view of processing. As we’ve seen, applications of VSMs don’t typically compute precise word vectors, but use some form of approximate addressing to access words (e.g. finding the nearest neighbor to the computed vector). Translated into a computational-psychological process, this would involve a quick vector operation (e.g. matrix multiplication) followed by a local search. This convergence behavior may be important for completing the kinds of analogies discussed in (Milokov et al 2013a, c): thinking of the male analogue of a queen may involve actually performing a vector
operation\textsuperscript{133} and then finding the nearest match to the result. Borrowing an idea from the analysis of dynamical systems (and neural networks in particular), we can perhaps think of the representations in question as point-attractors in the vector space, where regions correspond to attractor basins.\textsuperscript{134}

One of the best-known connectionist-inspired theses in the philosophy of mind is Paul Churchland’s account of concepts as attractors in vector space (Churchland 1989).\textsuperscript{135} Pete Mandik describes the proposal as follows: “Churchland proposes to identify concepts with attractors in hidden unit activation space...The network’s concept of dogs is a region of activation space, the center of which determines a dog that would be perceived as the prototypical dog.” (Mandik 2009, §4). As Erk (2009) notes, human conceptual abilities have often been modeled in terms of feature vectors, for example in prototype- or exemplar-based theories of concepts (see Prinz 2002 and the following section). We might then think of the vector-space representations corresponding to words and phrases as, simply, concepts.\textsuperscript{136}

On the account being developed here, “concepts” would be taken broadly so as to include sentence-level contents. At this point, it is important to address a basic question about such a proposal: given that simple concepts corresponding to words are attractors in the

\textsuperscript{133} It may be possible to subsume cases of analogical reasoning like this under the same mechanism that is used to compute phrase-level representations. For example, finding the solution to the analogy MALE : KING :: FEMALE : ____ might involve searching for the concept FEMALE KING. The initial superposition of the constituent concepts would be high in energy due to its implied inconsistency and hence low probability (cf. Hopfield 1982, Hinton and Sejnowski 1983, and the remainder of this section), so that the network would subsequently settle on the previously learned QUEEN concept. As discussed above, Milokov et al use simple vector arithmetic to complete the analogies, but this isn’t necessary (though one would want whatever composition function is used to yield similar results in these cases).

\textsuperscript{134} Once again, there are ties here to the notion of content-addressable memory—see (Hopfield 1982).

\textsuperscript{135} Though I clearly share the Churchlands’ interest in connectionist models of cognition (see also Churchland 1986), I have not much discussed their work here because my aim is to show how the intentional states posited in commonsense psychology might be realized in neural networks, not to replace intentional attitude descriptions of mind with neuroscientifically informed ones.

\textsuperscript{136} Erk herself distinguishes on conceptual grounds between the “co-occurrence space” that figures in VSMs and the space of concepts in human psychology, on the grounds that the former contains “mixtures of uses” rather than “potential entities” (her §3). However, the two could nonetheless could turn out to be (as they say) \textit{a posteriori} identical, since an entity could be picked out by a maximally specific description.
network’s state space, and that phrase-level representations are treated as (regions centered on) points intermediate between these attractors, how are phrase-level representations maintained, e.g. why doesn’t the network simply fall into the attractor basin of whatever concept the phrase-level representation is closest to?

As Horgan and Tienson (1996) stress, attractor behavior in neural networks is not all-or-nothing. Typically, there will be multiple forces vying for control of the network at any moment, and in this context each such force has a semantic interpretation. When I entertain the thought whose content is expressed by the sentence “That horse is a good skateboarder”, the set of co-active concepts may maintain a vector space representation that is (thanks to the tension induced, so to speak, by many different nearby attractors) held at a position in the vector space intermediate between the stored concepts. Indeed, this is just “parallel constraint satisfaction”, a hallmark connectionist computational principle, combined with the idea of concepts as attractors. Of course, as in the cases of syntactic transformations and inference discussed earlier in connection with VSAs, we may suppose that overlearned or oft-tokened complexes of existing concepts create a new concept.

Some nuance can be added to this picture by noting that even representations for never-before-seen phrases constructed on the fly (such as, perhaps, “skateboarding horse”) may be attractors relative to nearby points in the space. An alternative, potentially richer way of looking at attractor behavior is in terms of the “energy landscape” of a network’s state space (see e.g. Hinton and Sejnowski 1983, Hinton 2005). Roughly, low-energy states are more probable than high-energy states and correspond to more (logically and probabilistically) consistent sets of representations.\(^{137}\) Since the probability of a given configuration of the hidden

\(^{137}\) It is assumed that the probability of a representation being tokened is closely related to the probability assigned to the relevant content by the system (see, again, Hinton and Sejnowski 1983). Defending the simplest version of this way of representing probabilities is challenging for many reasons (to take just one: I may intuitively be much more likely to think of a unicorn than a “unipig”, while supposing both animals to
units in a network depends on its input (including any input from lateral or recurrent
connections), the energy of various states can be changed dynamically by changing the inputs
(over short timescales) or by changing the weights (over longer timescales).

Such landscapes may have detailed topological features, so that for example, although
SKATEBOARDING HORSE is itself an unlikely concept to entertain, and so high-energy in an
absolute sense (e.g. if asked to picture a skateboarding animal, a horse wouldn’t be one’s first
thought), there will still be a point in the state space that is an attractor given the constraint that
SKATEBOARDING and HORSE are to be combined into a single representation. For example,
I find it more natural to imagine a horse traveling sideways with a skateboard under each pair of
legs (front and back), than for example a horse standing on a single huge skateboard, or on two
elongated ski-like skateboards. Of course, the descriptions I’ve just given could be used to
enforce further constraints, yielding the complex concepts I’ve just entertained.

What of the point, raised at the end of the previous section, that any content in the
system can be thought of either as conceptual (in the traditional sense, i.e. as word- or
phrase-like) or propositional (sentence-like)? One somewhat clumsy way of accommodating
this would be to appeal to one-word sentences (Quine 1960). But I am not sure, again, which
propositional attitude HORSE corresponds to. There is a way of thinking about these matters
that skirts this issue: rather than thinking of words as mapping onto mental representations that
have contents on a par with sentences, we can think of them as stimuli that constrain mental
representations (Elman 2014, §4.2):

...suppose that one views words not as elements in a data structure that must be
retrieved from memory, but rather as stimuli that alter mental states (which arise
from processing prior words) in lawful ways. In this view, words are not mental
objects that reside in a mental lexicon. They are operators on mental states.
From this perspective, words do not have meaning; rather, they are cues to

be equally (im)probable). Further defense of this thesis is beyond the scope of the present discussion,
but the idea that coherent representations are generally more likely to occur than incoherent ones is
certainly intuitive. Thanks to Eric Mandelbaum for pressing me to be explicit on this point.
meaning...This scheme of things can be captured by a model that instantiates a dynamical system. The system receives inputs (words, in this case) over time. The words perturb the internal state of the system (we can call it the ‘mental state’) as they are processed, with each new word altering the mental state in some way.

I see no reason to have to choose between this “perturbation” model of the contribution of words to meanings and a memory access model, provided the “memories” are construed as attractors in a neural net’s state space. But otherwise, I find Elman’s description of things satisfactory.

In particular, note that while words may be “cues” to meaning, so may visual impressions, memories, hunger pangs, etc. In short, influences of all kinds might act as soft constraints on what is represented in a vector space, where the vector space in which cognitive-level representations reside is perhaps one realized in a highly interconnected cortical region (Damasio and Damasio 1994).

It is important to consider how the idea that word-level representations are learned from co-occurrence counts might interface with this picture. In the VSM models discussed above, for example (Milokov 2013a,b,c) and (Collobert et al 2011), the word vectors are implemented as matrices, where the matrix row corresponding to a word is selected via multiplication with a one-hot encoding of the word. We may suppose that in a more psychologically realistic model, learned mappings from sensory inputs caused by words and utterances to high-level neural codes (i.e. activity vectors) would play this selecting role.

Importantly, instead of simply retrieving a stored “word vector”, perception of a word in a context would alter the energy landscape of the relevant vector space, so that the word’s contribution to the overall represented content is added to whatever was already being represented, modifying the existing content. Incorporating this idea necessitates only a slight but crucial revision to the simple account of concepts as attractors in state space: concepts are instead operators on state space. This may in fact be cast just as a generalization of the
simpler idea. Suppose for the moment that vector addition is the relevant composition operation. Then the point in state space corresponding to a word is simply the result of applying the word’s characteristic operator, \(+word\), to the zero vector. Similar accounts could be used with more complex composition functions, provided that they yield the identity mapping when applied to the “initial” (null) state of the system. This conception of word meanings as operators may be important in modeling logical operations, as I’ll discuss later in §6.4.

The idea that the energy landscape is influenced by much more than just the perception of words presents no special problem: other sensory inputs, as well as endogenous influences like long-term memory and so on, can add their own constraints to the state space without interfering with those imposed by the words. Indeed, connectionist computational models that function in essentially this way actually work to model core psychological processes, as I’ll discuss shortly. First, I consider in-principle arguments against any account of concept composition like the one just outlined.

5.3 The prototype theory of concepts revisited

In this section I’ll further motivate the idea that whatever compositional approach is employed in tandem with VSMs to attempt to explain systematicity, it should be one in which composition functions themselves are empirically learned. This brings me to the second major Fodorian challenge to connectionist-style modeling of cognition that I’ll consider in this dissertation. According to Fodor and Lepore (Fodor and Lepore 1996, 2001), the compositionality of possession conditions for concepts entails that concepts can’t be stereotypes, prototypes, or anything else statistical. They also, by a similar argument, (allegedly) can’t be uses or inferential roles (Fodor and Lepore 2001, p.365). I’ll consider the
arguments against concepts as prototypes given in (Fodor and Lepore 1996) in some detail over the next few sections.

The specific version of the prototype theory of concepts that Fodor and Lepore criticize claims that having a concept is a matter of having the capacity to represent a prototype (e.g. a paradigmatic or statistically normal case), “together with a measure of the distance between the prototype and an arbitrary object in the domain of discourse” (Fodor and Lepore 1996, p.263). The account of concepts given in the previous section is clearly in the spirit of this description in important respects. The criticism, quoted at length below, is that productivity would not result from this theory of concepts (p.263):

Prima facie...the distance of an arbitrary object from the prototypical pet fish is not a function of its distance from the prototypical pet and its distance from the prototypical fish. In consequence, knowing that PET and FISH have the prototypes that they do does not permit one to predict that the prototypical pet fish is more like a goldfish than like a trout or a herring, on the one hand, or a dog or a cat, on the other. But if prototypes aren't compositional, then, to put it mildly, the identification of concepts with prototypes can't explain why concepts are productive.

This argument sounds convincing prima facie, but it presents no serious problem for the prototype theory.

First, it is just not true that knowing the prototypes associated with PET and with FISH does not allow one to predict that the prototypical pet fish is more like a goldfish than like a dog or a cat. Suppose that the PET prototype is a cat or dog, and that the FISH prototype is something like a trout. It stands to reason that when the concepts PET and FISH are composed, one will have to move some distance from the paradigmatic pet, since a pet fish is a fish. More subtly, it might stand to reason that one will have to move some distance from TROUT as well, since smaller, more colorful fish might make better pets for the average person.
The foregoing makes the assumption that the composition function is more or less rational, but I don’t see why this should be ruled out a priori. However, there is a more basic reply available: Fodor and Lepore are not letting the composition function itself (as opposed to the concepts) do any work. The motivating thought behind the argument is that “a gold fish is a poorish example of a fish, and a poorish example of a pet, but it’s quite a good example of a pet fish” (p.262). But why expect the prototype for the complex to be identical (or even particularly similar) to the prototype for either of the contributing concepts? On any theory of compositional meaning, the meaning of PET FISH is not identical to that of PET or of FISH. The argument seems to assume that the composition function is a maximally simple one.

The issue Fodor and Lepore are really concerned with, however, is that any such composition function for prototypes would seem to depend on empirical knowledge (Fodor and Lepore 1996, p.265). Even if the pet fish example could be accommodated using clever quasi-deductive reasoning as suggested above, there are doubtless other examples that will serve to make the point—consider the prototype for EIGHTEEN-WHEELED VEHICLE). Apart from violating typical Classical assumptions about the autonomy of the machinery of concept composition with respect to empirical matters, this may seem to preclude systematicity, since the function from the prototypes of simples to the prototype of the complex is therefore importantly context-dependent.

But the way in which the semantics of concepts contribute to the semantics of many semantically complex expressions may be highly context-dependent, even if the meanings of the concepts themselves are not context-dependent, and even if semantic composition isn’t context-sensitive across the board. The reply to Fodor and Lepore’s worry given in (Prinz 2012)

---

138 Fodor and Lepore (1996) seem to treat this fact—that composition is sensitive to reasons, and to logical relations in particular—as decisive evidence in favor of the idea that a concept contributes its “logical form” to a complex rather than its prototype (p.264). But why should a prototype theory of concepts have to deny that logical relations matter for composition?
is, in my view, the correct one in broad strokes: There is a default prototype determined \textit{a priori} for every complex concept, sufficient to ensure the facts of systematicity and productivity, but world knowledge may determine the prototype of a complex concept when it is available.

Learned departures from a default prototype violate the letter of compositionality, since the meaning of some complex expressions is no longer a function just of the meanings of relevant concepts and the way in which they’re composed (at least, the composition function may change from case to case, and may depend on world knowledge). But we need to explain the fact that the prototypical pet fish is a goldfish anyway. If this version of the prototype theory can do so without sacrificing anything essential to explaining the psychological facts about productivity and systematicity, so much the better for it.\textsuperscript{139}

Prinz elaborates this idea by appeal to a model of concept composition proposed in (Prinz 2002) encompassing three stages: retrieval, composition, and analysis (hence, the \textquotedblleft RCA\textquotedblright{} model). On this account, we first look for a stored exemplar of the concept (including one that’s \textquotedblleft cross-listed\textquotedblright{} for the components of a complex concept). If that doesn’t yield anything reasonable, we compose the input concepts using minimally smart heuristics. Then, we introduce additional features if needed to \textquoteleft{}clean things up\textquoteright{} at the analysis stage. Prinz (2012, §21.4) assumes a rather \textquotedblleft{}dumb\textquotedblright{} default composition process:

Using compositional algorithms, one would represent a pet fish as a medium sized, furry, scaled, quadruped, that lives both in a body of water and in the home, and gets taken for walks. Of course, this absurd, Boschian compound will immediately be discarded by any one with a bit of background knowledge. And that’s just the point. If a compound is familiar, there is no need to use a compositional procedure.

\textsuperscript{139} Fodor and Lepore consider this proposal, but they reject what seems to me to be a strawman version of it. For one, they still insist that the \textquotedblleft{}default\textquotedblright{} prototype for a complex concept must also be a prototype for each constituent concept. They also suppose that all complex concepts that depart from the default prototype must be idioms. But a more sensitive composition function needn’t yield the first result, as discussed above. And it needn’t yield the second either, because if the default prototypes are already in the vicinity of the empirical truth, one can conceive of the effects of learning not as wholly overriding the default procedure, but as simply biasing the composition function in those cases (perhaps by changing a prior over the space of prototypes)—see below.
The RCA model is not intended as a competitor to extant theories of concept composition in the psychological literature (see e.g. Smith et al 1988), but rather as a schematic theory that captures the essentials of any such proposal. However, I think a proposal that is at once more minimal and more informed by neural processing models can be given, according to which “default” composition is accomplished by the same mechanism that brings empirical knowledge to bear where relevant.

I suggest that all three stages of RCA can be taken care of by the energy-based dynamics of a vector space. First, “search” for a “stored exemplar” is no different than construction of a new exemplar (e.g. composition). Search is just letting the network settle into an energy minimum with the input concepts “clamped” so as to constrain the neural network’s state. Of course, energy minima correspond to “stored” concepts, but if no previously learned concept exists that satisfies the constraints imposed by the constituent concepts, the network will still be pushed into a uniquely determined region of the state space, which is an energy minimum given the clamped concepts, as discussed in the previous section. Since the state space will have been sculpted by empirical knowledge, such knowledge can naturally be brought to bear on the composition process. In fact, this unifies the “retrieval” and “analysis” functions in Prinz’s model: knowledge of specific categories like pet fish can strongly bias the composition process, while general background knowledge can be used to reject candidate constructed prototypes when there is no prior knowledge of a complex category.

There may be extremely difficult cases that amount almost to puzzles, such as Prinz’s PAPER RAINCOAT. I am open to the possibility that in such cases, some “analysis” strategy takes over that is in excess of the usual energy minimization search, though it might also just be the same process iterated (see Hinton’s remarks on intuitive versus rational inference in (Hinton
1990)—cf. §1.3 above). And, moreover, it’s important to keep in mind that in a neural network-based approach, the composition function could be quite complex.

The latter may be important in the context of a somewhat different sort of objection that is a major concern of Fodor and Lepore’s (1996, §3): it seems that the semantics of truth-functional connectives cannot be understood in terms of prototypes. The prototype of NOT RED, if there is one, is pretty clearly not the prototype of NOT somehow combined with the prototype of RED, on any casual reading of “prototype” (what would the prototype for NOT be, anyway?). More generally, prototypes, like vector space models (cf. §3.6), seem a good fit for substantive terms (nouns, verbs, adjectives, adverbs), but not as obviously a good fit for function words (logical connectives, but also e.g. prepositions, articles, and quantifiers).

It’s important to look closely at how this kind of example of concept composition differs from PET FISH. Putting things extensionally, the main difference is that the set of pet fish is the intersection of the set of pets and the set of fish, while the set of things that aren’t red is of course not the intersection of the things that are red and the things that are not. A similar point can be made in an intensional idiom: pet fish have the properties of pethood and fishhood, while not red things don’t have the property of being red (and it’s debatable whether they have the property of being not).\footnote{Worries of this kind are discussed in (Baroni and Zamparelli 2010).} \footnote{Notice that this kind of example poses a problem for any simple definition of the “semantic constituency” relation that was initially used to motivate the compositionality thesis. This matters, because as discussed in §1.1, the bare idea that the semantics of some expressions are a function of the semantics of others is not sufficient to get us compositionality in any interesting sense. I won’t press this point, however. I assume that systematicity is a genuine phenomenon, even if the set of semantic relations on which it depends can’t be characterized so succinctly.}

Relatedly, we must be sure that our model addresses a problem posed for simple associationist theories of word meaning by Hayley Clatterbuck.\footnote{Hayley Clatterbuck, “Associationism and Hierarchical Structure”, talk given at the 2018 Southern Society for Philosophy and Psychology meeting, San Antonio, TX.} What exactly is keeping the network from selecting a PET FISH prototype that isn’t even a fish, if the PET prototype
influences processing enough? That is: we want the input concept PET to act as a hard filter
on the potential features of the PET FISH concept, not just as one premise among others that
figure in a probabilistic inference. Here, there is a crucial choice to be made between
thoroughly probabilistic architectures that merely make such a possibility vanishingly unlikely,
and those that build in discrete mechanisms to prevent it in principle.\textsuperscript{143}

I am not sure whether the prototype theory of concepts can be defended against this
latter type of objection. Fortunately, I am not (at least primarily) concerned to defend that theory
as such here. The proposal that concepts correspond to operators on a vector space survives
these objections. The word ‘not’ will, after all, have a well-defined effect in a VSM,
corresponding to its place in a distributional model on the simplest approach (or to a dedicated
function, as in the hybrid approach discussed in §3.6). It seems likely that logical connectives
and content words will function very differently in this space, and it is an open empirical question
whether the mechanisms described so far can accommodate the function of the former.
However, I save further discussion of this issue for the sections on logic in chapter 6.

### 5.4 Machine translation and the language of thought

In the remainder of this chapter, I take up the thread at the end of §5.2, and examine
connectionist models that appeal to the idea that multiple information channels may converge
on a single common vector space housing an evolving mental representation that is perturbed
by sensory inputs and used to generate motor outputs. These models implicate natural
language processing\textsuperscript{144} in more varied and intriguing ways than the parsers considered in

\textsuperscript{143} For more on this theme, see Chapter 6 below.

\textsuperscript{144} For a thorough introductory survey on the topic of neural NLP, whose scope extends well beyond this
chapter, see Goldberg (2016).
Chapter 4. Some of them, such as Google’s neural machine translation system, discussed in this section, are among the most successful models of capacities usually associated with higher or “general” intelligence to date.

A background goal is to further bolster the idea that connectionist cognitive architectures can support systematicity. The reasoning is that if a connectionist network models hallmarks of general intelligence, for example by successfully translating arbitrary natural language sentences, it therefore exhibits a kind of de facto systematicity in its representations.\textsuperscript{145} Whether it does so in a way that is closely related to the human case is a further question, but the models themselves provide an existence proof of sorts that Classicism is not the only game in town. The networks described here attempt very specific but still cognitively demanding tasks: translation, cross-modal mappings of various sorts (e.g. natural language description of the contents of a visual image), and question answering. In the following chapter, I also consider connectionist models of reasoning.

I begin with a discussion of neural network-based approaches to automated translation. The basic approach to translation shared by the most successful recent neural network models (see especially Google’s recent efforts in Wu et al, unpublished and Johnson et al 2017) is nicely summarized by Vinyals et al (2015, p.1): “An ‘encoder’ RNN reads the source sentence and transforms it into a rich fixed-length vector representation, which in turn is used as the initial hidden state of a ‘decoder’ RNN that generates the target sentence.” Google’s approach uses a deep version of this strategy (8 layers in each direction, plus an additional attention network, as discussed below).

\textsuperscript{145} I assume, perhaps pivotally, that ‘representation’ is monosemous across the contexts of psychology and computer science, even if \textit{mental} representation is a privileged genre. I lack space to argue this point here, but see (Kiefer and Hohwy 2017) for an account of representation in terms of structural similarity that would seem to work across both domains (as well as in the case of natural language, given the distributional hypothesis—cf. §3.1).
As discussed previously, a recurrent neural network (RNN) is a type of artificial neural network whose “hidden layer” activities at a given time, $h_t$, are a function both of current input $x_t$ and the hidden layer activities at the previous time step, $h_{t-1}$ (see e.g. Karpathy 2016, §2.4.3, for a good, minimally technical introduction to RNNs). In the simplest RNN models, the function in question is a sum of linear transformations of $x_t$ and $h_{t-1}$, followed by a nonlinearity $g$ (i.e. $h_t = g(W_{xh}x_t + W_rh_{t-1})$, where $W_{xh}$ is a “synaptic” weight matrix mapping $x_t$ to an $h$-dimensional vector, and $W_r$ is a recurrence matrix connecting $h$ to itself over time). Google’s translator uses the “vanilla” version of the LSTM, discussed in connection with parsing in §4.3.

The kind of system just briefly described underlies the translation algorithms implemented in Google Translate as of this writing. I leave it to the reader to judge the quality of the resulting translations, but it seems uncontroversial that translation quality received a major boost when Google switched to neural network-based approaches a few years ago. Certainly, moreover, if there were a more effective approach on offer, Google would use it.

Several special features of Google’s approach to translation are worth mentioning, before discussing the general representational principles underlying neural machine translation. First, the most common and perhaps the most natural way to use an RNN to model sequences is to encode the sequence one element at a time, letting the final layer of the RNN accumulate information from each time-step, yielding a single fixed-length vector that represents the sequence. The system presented in (Wu et al unpublished), by contrast, saves the top-level RNN encoder features at each time-step, so that a list of fixed-length vectors is used to

---

146 I.e. it is based on a simple RNN, not a RecNN.
147 It is important to keep in mind, of course, that not only sheer accuracy or quality of translation is relevant here, but also memory and time constraints. In general, the predictive performance of statistical models can always be enhanced by averaging across an ensemble of models, including potentially models that bring to bear all sorts of hand-coded domain-specific knowledge. But ensemble averaging would not be viable for Google’s system, which must return results within seconds. Although neural network approaches tend to allow for faster inference than some other statistical approaches, Google’s system still makes use of approximations to speed things up, such as reduced precision arithmetic (see Wu et al unpublished, §6).
represent the encoded sequence, rather than a single compressed representation (i.e. the last element in this list).

Second, Google’s approach, like many models in this domain, uses an additional “attention network”, allowing the decoder network to focus on aspects of the encoded input (the sequence of vectors just discussed) predicted to be useful for the task at hand at each time-step of decoding. In addition, they use “residual” or “skip” connections (He et al 2016) during both encoding and decoding, which pass input directly from, e.g., layer $L_n$ to layer $L_{n+2}$, and add it to the transformation of the input computed by the previous layer $L_{n+1}$. This technique is motivated by theoretical considerations about optimization, allows very deep networks to be trained successfully, and is also biologically inspired (since real cortical connectivity is not limited to a simple chain of adjacent processing regions, even in the feedforward direction).

Finally, Wu et al employ a bi-directional recurrence function in the first layer of the recurrent encoder, i.e. a combination of a standard RNN that steps through the sequence of words from left to right, and a “reverse” RNN that steps through backward starting from the end of the sentence (Schuster and Paliwal, 1997). This feature of the Google model in particular seems wildly unmotivated from a psychological/biological point of view, since we normally begin to understand sentences as soon as we start hearing them, and we certainly do not in general build up mental representations of sentences by reading them in reverse. It is arguable that translation, however, normally involves first hearing an entire sentence (or, at least, phrase), then formulating an equivalent sentence or phrase in the target language. There is probably nothing psychologically or neurally plausible about the bidirectional RNN model, but the more general suggestion that the output of the translation process depends on information about the entire source sentence is reasonable, and could be mediated by some other mechanism.
One feature of this network may be important in the context of the philosophical issues at large in this dissertation: it seems as though, in this system at least, the hidden-layer representation is in effect a set of concatenated representations of each constituent. Is this, then, an essentially Classical system? I think not, for two reasons. First, though concatenating the top-level LSTM states in this way produces optimal performance, many similar systems perform well by using only the final state computed by the recurrence, as we’ll see below. Second, the concatenated representations in this network are not Classical constituents: rather, they are representations of the entire sequence processed thus far (which, given the bi-directional encoder at the first layer, always includes more than one lexical item), biased toward the most recently processed words.

The overall gestalt that this model presents, of a network building up a vector-space representation of the input sentence in one language, via recurrence, and using that very same representation to generate a sentence in the output language one word at a time, has immense intuitive appeal and squares with the “perturbation” model of meaning discussed in §5.2 and in (Elman 2014). One appealing property of this picture is of course its sheer simplicity, at least in outline. Another is that the network seems to build up a language-agnostic internal representation of the sentences it processes.

This possibility was explored most explicitly in (Johnson et al 2017), which proposed a system for multilingual translation employing the same architecture discussed in (Wu et al unpublished), which was designed for a single language pair. Simply by adding an output-language-identifying token to input sentences and a linguistically diverse dataset for training, it was shown there that a system could learn to translate between multiple languages using the same architecture. Moreover, forcing the system to use a single model to translate multiple languages in this way facilitated translation between novel language pairs. Somewhat
surprisingly, this model was able to perform “zero-shot translation”: accurate translation from $L_1$ to $L_2$ even when no such translations were part of the training data. This demonstrates the viability of the RAAM (Pollack 1990) approach to composition: while one might worry that using the same set of weights to capture more data might stretch the capacity of the network, it in fact aids generalization, which in this context amounts to systematicity.\footnote{Of course, this only works given a large enough model, which the one under consideration surely is.}

Since accurate translation depends on sensitivity to the semantics of a sentence (which in turn depends, at fine grains at least, on understanding the roles of words with respect to one another captured in syntactic relations), and since the output of the encoding process is used directly to condition the language model for the output language, it seems that the representation built up at the highest layer of the LSTM in these models must capture whatever meaning both sentences have in common. This sounds like one of the holy grails of artificial intelligence and cognitive science: a computational model of internal language-neutral representations of propositions, i.e. thoughts. Indeed, Geoff Hinton (anecdotally, at least) popularized the term “thought vectors” for such language-neutral representations.

In a post by Google’s AI blog about the “zero-shot” translation system,\footnote{Available as of 8.17.2018 at https://ai.googleblog.com/2016/11/zero-shot-translation-with-googles.html.} researchers inspected a low-dimensional projection of the vector space encodings of translated sentences, and found that sentences from different languages with similar meanings were mapped to similar parts of the vector space.\footnote{Moreover, though this was not included in the report, it is also presumably the case that multiple input-language sentences receive similar vector space embeddings. It would be odd indeed if the network managed to encode semantic similarities across languages while failing to do so within a language.} Interestingly, these “synonymous” sentences from different languages, though close together in the space, are slightly shifted with respect to one another (it is also worth noting that each sentence is represented by a set of nearly contiguous points in the space, since the high-level encoding is a list of vectors, as discussed above). The Google
team concluded that the network had invented its own “interlingua”, a headline picked up by several popular articles on the subject.

A philosopher reading about these systems would be hard-pressed not to be reminded of the LoT hypothesis. A common vector embedding space for multiple languages does not on its own amount to a language of thought in the Classical sense. However, an amodal, language-neutral semantic representation whose contents can be expressed in sentences of natural language is in itself of very considerable interest for cognitive science and the philosophy of mind. All of this suggests, in any case, that statistical approaches to translation in general, and neural network-based approaches in particular, are fairly well-positioned to form the core of elegant explanations of this aspect, at least, of human NLP capacities.

That said, the Google approach to translation, while effective, has its drawbacks. One such drawback, from my point of view, is that it relies on a purely supervised learning procedure: given a pair of training sentences, error is backpropagated from the output sentence representation to the input sentence representation, adjusting the weights along the way in proportion to the error signal so as to improve the ability of the model to predict output given input sentences. This is a drawback from the point of view of psychological realism because the representations learned (both at the highest layer of the LSTM and in the intermediate layers) are tailored specifically for the translation task. A language model for the output language must be implicitly learned (since good translations must be syntactically well-formed), as well as an encoding for input-language sentences, but presumably real speakers have an independent understanding of both languages antecedent to the translation task. Certainly, at any rate, they don’t learn to translate by beginning with no knowledge of language at all and being shown sentence pairs.
Moreover, though the Google system discussed in (Johnson et al 2017) does demonstrate the somewhat remarkable property of being able to perform zero-shot translation, thus showing that the vector representations learned by this method facilitate some degree of generalization, it is unknown how well the representations in this network would perform if detached and used for other tasks relevant to natural language processing, and to cognition more broadly (i.e., it is unclear whether these particular representations, at least, would functionally earn the title of “thought vector”). However, results such as those in (Milokov et al 2013a), in which good embeddings at the word level were learned as byproducts of a supervised task, give some cause for optimism.

Finally, though concatenating the RNN-derived encodings of a sentence at each time step preserves more information than using a single compressed RNN representation and works well in practice, this is probably not a psychologically realistic approach as the sentence representation grows in dimensionality with sentence length, is highly redundant, and accordingly does not make use of one of the most appealing properties of RNNs: their natural compression of temporally extended sequences into a single vector representation.

### 5.5 Multimodal connectionist nets

I now discuss a set of connectionist computational models that may be construed as embodying the assumption that there is a common high-level representational space shared across linguistic and sensory processing systems (see Jackendoff 1983), and across different sensory modalities, and that the sharing of this space has important functional consequences (cf. the conclusion of §5.2 above). The connection between these models and the translation
system just examined will be taken up in the next section. I begin by discussing a series of experiments by (Ngiam et al 2011) on multimodal Restricted Boltzmann Machines (RBMs).

RBMs are neural network models that are “restricted” in that, as in the Helmholtz machine (Hinton et al 1995, Frey et al 1997), within-layer connections are not allowed. Layers are fully connected to one another via symmetric connections. As discussed briefly in §3.5, RBMs are ideal for unsupervised learning from data using an algorithm such as contrastive divergence (Carreira-Perpiñán and Hinton 2005).

As (Ngiam et al 2011) discuss, the simplest approach to using deep learning to model a situation in which two sensory processing pathways intersect is to simply concatenate inputs from both modalities and feed them simultaneously to the network. For example, if one hopes to train a model that simultaneously processes video and audio data, frames of video and slices of audio may be concatenated and used as the input to a hidden layer. Ngiam et al pursued this approach, using several databases containing video of people pronouncing digits 0-9 and letters A-Z with accompanying audio. They found that “learning a shallow bimodal RBM results in hidden units that have strong connections to variables from [an] individual modality but few units that connect across the modalities” (p.2): information from the two modalities fails to mix significantly in the network’s hidden layer vector space. This, the authors chalk up to the fact that “the correlations between the audio and video data are highly non-linear” (p. 2).

---

151 As described in (Hinton 2010), RBMs consist of layers of stochastic binary units (cf. the model of Hinton and Sejnowski 1983), but they can also be used with other sorts of units—here, the authors use real-valued units for the inputs.

152 In the experiments considered here, audio samples are represented by slices of a spectrogram.

153 That is to say: if one were to attempt to draw a boundary in $n$-dimensional space ($n$ in this case being the number of pixels in the video input) around just the video datapoints that tended to occur along with a certain audio datapoint in the multimodal input, this boundary would be highly curvy. Part of the motivation for deep learning methods in general is that they extract relatively abstract representations of features that are more simply related to one another than the raw data points might be (that is to say: the vehicles of the explicit higher-level representations bear simpler mathematical relations to one another than do the vehicles of lower-level representations). Ideally, high-level representations that indicate the presence of a certain feature are linearly separable from those that do not (as in a simple scalar-valued node that acts as a feature detector).
To produce representations that meaningfully span modalities, then, one might try composing a deeper network by training a shallow network on each modality separately before combining the streams. Ngiam et al attempt this by first training an RBM each on the video and audio datasets separately, then feeding the learned features into a common representational layer as before, using the outputs of the previous RBM as inputs to the next in a “stack” (as in Hinton 2007). As the authors put it, “By representing the data through learned first layer representations, it can be easier for the model to learn higher-order correlations across modalities. Informally, the first layer representations correspond to phonemes and visemes and the second layer models the relationships between them” (Ngiam et al 2011, p.3).

The authors experiment with several variations on this idea across different learning and task settings. I will consider only one of their models in depth here: the “bimodal deep autoencoder”. The authors begin by training the two-layer network just considered one layer at a time, using the contrastive divergence algorithm. This “extracts” a set of representations that

These explicit representations can be used to drive feature-specific processing. In particular, they make the extraction of yet higher-level features easier. There may be a simple linear relation, for example, between the presence of certain place-coded feature representations relevant to face detection and the activation of the face-representation itself (mirrored, of course, in the correspondingly simple linear relationship between the things represented, when cast for example in terms of probability or confidence): the probability that a face is present is perhaps a weighted sum of the probabilities of features in the right spatial relationships to one another.

This point about the extraction of linearly separable representations is crucial to understanding how deep neural nets work, though it is tangential in the present context (hence the footnote). One of the motivations behind the capsules framework (§4.6), for instance, is that once objects are represented in terms of feature vectors, transformations can easily be applied directly to these representations, leaving it to the fixed mapping from abstract representations to features in pixel space to determine the consequences of this transformation on the appearance of objects.

The ease of working with such abstract features is also illustrated in the work of (Radford et al unpublished). They show that randomly walking through the vector space of the latent variable (e.g. hidden layer) of a generative neural network for images of bedrooms produces a continuous stream of more or less realistic rooms. By contrast, randomly walking through pixel space would produce mostly noise. This is because valid room-images “live” in a (highly curved) subspace embedded within the total space of possible collections of pixels. The hope, in training a deep neural net, is that a (necessarily non-linear) mapping will be found between this pixel space and the (usually) lower-dimensional, linearly related representations in the latent variable space. These latent variables, in a generative setting, can be used to generate all and only those states of the pixel-space representation that look like faces. See also the interpolation studies in (Bojanowski 2018), Figure 5.
capture important features of the inputs, as just discussed. The authors then use the weights thus learned as the starting-point for learning an auto-encoder that is trained to reconstruct both audio and video data.

Interestingly, the representations learned by this network, when used to classify pronounced syllables on the basis of audiovisual cues, reproduce the McGurk effect observed in humans (McGurk and MacDonald 1976). In the McGurk effect, subjects visually presented with a speaker enunciating the syllable /gal/, paired with audio of a speaker enunciating /bal/, subjectively perceive a /dal/ sound. When the bimodal autoencoder's hidden-layer feature representations are used to train a three-way classifier for these sounds, the classifier predicts /gal/ and /bal/ accurately for consistent audiovisual input, but predicts /dal/ when presented with a visual /gal/ and audio /bal/ (these syllables were not present in the audio or visual training data). The authors also show that under some conditions, features learned from both audio and video data yield better results when classifying spoken letters/digits presented in a single modality than do features learned using that modality alone. The moral seems to be that a genuinely integrated amodal representation is learned that combines information from multiple modalities, in ways that plausibly model cross-modal influences known to occur in human perception.

Work by Nitish Srivastava and Ruslan Salakhutdinov (Srivastava and Salakhutdinov 2014), again on multimodal deep RBMs, explores a similar approach. They more explicitly link these ideas to relevant psychological function, and to the interaction between language and vision in particular (p. 2949):

> Useful representations can potentially be learned...by combining the modalities into a joint representation that captures the real-world concept that the data corresponds to. For example, we would like a probabilistic model to correlate the occurrence of the words 'oak tree' and the visual properties of an image of an

---

154 I will not pause to describe this model in detail, and only discuss its main results and theoretical motivation. The model is similar to that in (Ngiam et al 2011), but it involves more independent layers for each modality before joining, and it is trained as a Deep Boltzmann Machine (Salakhutdinov and Hinton 2009), an undirected generative model, rather than as an autoencoder.
oak tree and represent them jointly, so that the model assigns high probability to
one conditioned on the other.

It’s not that this is such a radically new idea. Obviously, what you see influences what you
think, what you say, and so on. What’s perhaps surprising, if these models are on the right
track, is that no complicated domain-specific processing is required to mediate these influences.
The idea of fixing the state of a high-level amodal or multimodal representational space using
one modality\textsuperscript{155} and using this representation to condition the generation of representations in
another, combined with the idea that each processing channel can be used to shuttle
information in both directions (i.e. as both a “recognition” and a “generative” model—see Frey et
al 1997), may help explain a wide variety of psychological functions.

For example, consider a model architecture devised by Yee Weh Teh, Geoff Hinton, and
colleagues (depicted in Figure 6 of Hinton 2005—the essentials are reproduced here in Fig. 9).
The architecture combines a directed acyclic graph in the lower layers (a belief net) with an
undirected “associative memory” in the top two layers, as well as bottom-up weights for
approximately inferring the posterior under the model’s generative distribution given a data
vector. In the work just referenced, a joint generative model was learned of images of
handwritten digits and their labels. Inferring the class label from a visually presented digit is
accomplished by supplying the relevant pattern at the input layer, computing states successively
for the hidden layers up to the highest layer, then performing alternating Gibbs sampling\textsuperscript{156} using
the top two layers to activate the correct label unit. This method achieved an error rate of
2.25\% on the MNIST handwritten digit dataset, and can be lowered further to 1.25\% by

\textsuperscript{155} In this paper, as in others, language is treated as its own modality, on the grounds that images and
linguistic representations, in this case “sparse word count vectors” (p.2950), have very different statistical
properties.

\textsuperscript{156} I.e., values are fixed for the penultimate layer and used to conditionally sample states of the top layer,
then these top-layer states are used to sample states of the penultimate layer, and so on, repeated as
many times as desired. Samples drawn after more iterations are closer to the model’s true generative
distribution.
computing the exact free energy of configurations of high-level code and label units, and other tricks.

Figure 9 (After Fig. 6 of Hinton 2005).
White boxes: the combined directed/undirected generative model proposed by Teh and Hinton, where each box represents a layer of processing units. The largest layer at the top defines a joint distribution over labels and states of the penultimate hidden layer, which can be used to initiate a top-down pass to generate states of the input layer (“fantasies”). Grey boxes: a second (potentially multilayer) network may be used to derive the “labels” associated with inputs in the first network. Thus, modalities may be used to supervise one another.

This model is essentially a prototype for the kind of multimodal model I’ll discuss next: instead of a realistic model of a distinct sensory pathway, a set of 10 label units were used to model digit classes. But in a more realistic system in which two sensory processing streams
merge at the highest layer, “Different modalities can act like soft labels for each other” (Srivastava and Salakhutdinov 2014, p.2950).\textsuperscript{157}

If one of the generative models is a language model, this simple scheme can be used surprisingly effectively to model a variety of interrelated psychological functions, which are also useful in technological applications. First, a text cue can be used to select a high-level code which is then used to generate an image. This models the process via which mental imagery is generated based on a cue (as in, e.g., “imagine a vase of flowers”). It can also be used to retrieve stored images: the high-level feature representations caused by the text cue can be compared to the high-level features activated by images to find the closest match. The authors show that this kind of technique can be used in their network to retrieve pictures of cars and purple flowers using cues like “purple, flowers” and “automobile”.\textsuperscript{158}

Second, an image can be used in the same way to generate a snippet of text. This allows for a more realistic form of object recognition than that performed by most computer vision models (for instance, vanilla CNNs—see §4.4 above): instead of just activating a scalar-valued label that the experimenter conventionally associates with an object category, natural language descriptions that identify the category of an object or scene can be generated by conditioning on the hidden-layer state induced by an image. The authors again show that a simple version of this process can be accomplished in a deep Boltzmann machine: word labels can be retrieved that match input images (see Figure 5 of the paper).

\textsuperscript{157} This, incidentally, is part of the reason that I do not see a deep distinction between supervised and unsupervised learning, particularly as models become more complex, with more input channels.\textsuperscript{158} This network does not include a full language model or a generative model for images, but its fundamental principles of operation are the same. It is a more fleshed-out version of the kind of proposal floated in (Hinton 2005), but still one step away from the kind of full cross-modal inference I speculate about here.
5.6 Larger-scale multimodal models

I now consider a few models that show that this idea can be employed in the context of more realistic language models that generate (potentially novel) descriptions of input images, rather than single-word labels. In the words of Vinyals et al (2015, p.1):

Being able to automatically describe the content of an image using properly formed English sentences is a very challenging task...a description must capture not only the objects contained in an image, but it also must express how these objects relate to each other as well as their attributes and the activities they are involved in. Moreover, the above semantic knowledge has to be expressed in a natural language like English, which means that a language model is needed in addition to visual understanding.

The model in (Vinyals et al 2015) achieves image captioning by using an LSTM network to generate sentences, just as in the machine translation models previously discussed, but instead of conditioning the model on the recurrently encoded representation of a source language sentence, the model is conditioned on the output of a convolutional neural network (CNN) applied to the input image. Rather than translating from one language to another, this model translates between images and a natural language.

In slightly more detail: first, an image is fed to the network, and used to compute an abstract representation of the image. This abstract image-representation (a set of high-level, abstract visual features that would typically be fed into an $n$-way classifier for $n$ image categories in standard computer vision applications) is then used as input to the LSTM, which generates a distribution over output words. Sampling from this distribution, the first word in the output sentence is chosen, which is then passed as input (using a vector space word embedding) to the next round of processing. This word is combined with the existing state of the LSTM and used to output probabilities for the second word of the generated sentence, in a way that depends both on the input image and the first word—and so on until an “end-of-sentence” token is chosen as the next word.
Using this model, the authors consistently drubbed the competition according to several established measures of quality of generated captions (such as BLEU score). However, while (at least then) state-of-the-art, this system lacks diversity in its generated sentences: analysis revealed that 80% of the sentences ranked first as image captions had been previously observed verbatim in the training data.

Andrej Karpathy (Karpathy 2016) and Li Fei-Fei (Karpathy and Fei-Fei 2015) have done important related work in this area. On the vision side, they use a pre-trained CNN to compute abstract features for input images and specific image regions, and learn a mapping from these features to a multimodal embedding space. To represent language, they use bi-directional recurrent networks (as discussed in connection with the Google translation system above), beginning with “one-hot” word representations that are used to index rows in a matrix containing word embeddings (just as in the skip-gram model). Each word is represented, thanks to the recurrence, in a way that takes its surrounding context into account.

The joint (multimodal) embeddings are learned via a supervised procedure: using a training set of image-sentence pairs, the parameters in the model (including the word embeddings)\(^\text{159}\) are trained so as to maximize an image-sentence “alignment” score, which is a simple sum of the dot products of image-region and word vectors (see Karpathy and Fei-Fei 2015, Eq.8). The result, of course, is that image regions and words that apply to them are represented by nearby vectors in the joint embedding space. The authors then use these embeddings to train a generative model that produces sentences given input images. Fig. 10

\(^{159}\) In fact, the authors initialize the word embeddings with vectors learned using the skip-gram and CBOW models discussed above, but report little performance difference vs. using random initialization. Rather than impugning the quality of the skip-gram model, which we’ve already seen to be effective in practice, this indicates the power of the supervised learning procedure the authors subsequently employed.
below, for example, shows one case in which an appropriate sentence was generated that was not previously observed in the training set.\footnote{It was noticed independently by Richard Brown and Eric Mandelbaum that this man is in fact most likely \textit{tuning} a guitar. This may be the most devastating feedback I received on this dissertation.}

Performance of this model was consistently good (see paper for scores on standard metrics in this area, as well as several more examples), though there were also many failure cases, and again, only limited generalization from the examples observed in the training set. For example, though the description in Fig. 10 was not seen during training, the phrases “man in black shirt” and “is playing guitar” occurred many times each, and overall, 60% of generated sentences were found to have occurred \textit{verbatim} in the training data (see Karpathy and Fei-Fei 2015, §4.2).

In (Socher et al 2014), a different approach is taken that combines the multimodal architecture discussed here with yet another variation on the recursive neural nets discussed above: a “Dependency-Tree Recursive Neural Network” (DTRNN) is used to map input words

\textbf{Figure 10} (From Fig. 6 of Karpathy and Fei-Fei 2015). An image accurately captioned by the model. © 2015 IEEE.
to the common sentence-image embedding space. In this work, the focus is more on using sentences to retrieve relevant images than on image captioning, but it is shown that this network can be used to “search for sentences” by image as well.

In the latter half of this chapter, I have relied on a handful of examples from a huge literature to indicate something of the variety of functions that can be subserved by the “encoder-decoder” architecture I’ve been discussing. I consider one further model of a quite different task, that is not strictly a case of multimodal embedding. The model of (Yin et al 2016) generates natural-language answers given questions in natural language and an auxiliary knowledge database, and works very similarly to (and was inspired by) the translation and image-sentence alignment models just considered. In those models, sentence-generation begins by conditioning on information either from an image (in the latter case) or a sentence in a different language (in the former). In the Yin et al model, the question is mapped via a bidirectional RNN to a hidden layer representation, which is combined with information from the long-term database to condition an RNN that is used to generate the answer.

I conclude this chapter with some brief reflections on the architectural idea embodied in the proposals just discussed (viz, a common multimodal embedding space). First, consider again the fact (cf. §3.3) that identical types of vector arithmetic can be used to capture semantic relations in the case of both natural language and images (see Milokov et al 2013a, Radford et al unpublished, and Bojanowski et al 2018). This suggests, in addition to a domain-general processing style that applies to both vision and language (cf. §§4.4—4.6 above), a common type of representation, and associated form of systematicity, operative across the two domains.

Vector space representations corresponding to the meanings of linguistic expressions can be expected to exhibit these systematic relations thanks to corpus-based unsupervised learning together with the distributional hypothesis, and the models just discussed, such as that
in (Karpathy and Fei-Fei 2015), are trained explicitly to map images to points in the space close to their verbal descriptions. But why should hidden-layer codes for images be expected to exhibit the same systematic semantic relationships in purely visual generative models?

Radford et al (unpublished) combines the idea of adversarial generative learning (Goodfellow et al 2014), in which a generator network is trained to produce samples and a discriminator network is trained to distinguish them from real images, with a CNN architecture. As in (Goodfellow et al 2014), the generator works by learning a mapping from randomly drawn noise vectors $Z$, which serve as the hidden-layer codes, to images $X$. (Bojanowski et al 2018) use the same idea, but dispense with the adversarial training, and instead minimize the reconstruction error when members of $Z$ are used to generate members of $X$. The result is that $Z$ ends up, thanks to the learned mapping to $X$, serving as a compressed vector space model of the image manifold (e.g. dimensionality $d = 100$ for vectors $z$ and 4096 for images $x$ in Radford et al’s model). We can then apply the efficiency argument discussed in connection with the RAAM system in §4.1 above: in order to support the generation of many different images using only 100 (or whatever) latent variables, the space $Z$ must be used efficiently, so that similar vectors in $Z$ produce similar images. Relatedly, since such a compressed image model must capture salient dimensions of variation in the set of training images in its latent variables, it is to be expected that the latent variables will be linearly related to relevant image

---

161 As the authors note, the main difference between this model and an autoencoder is that instead of first learning an encoding $f(x)$ of the inputs $x$ and then learning a decoding, a mapping is instead learned directly from random vectors $z$ which are themselves learnable free parameters. This allows the model to access solutions out of reach for autoencoders, since the latter are constrained to use hidden-layer encodings that are functions of $x$ (see Bojanowski et al 2018, p.602).

162 Bojanowski et al also limit the representation space by drawing the random $d$-dimensional vectors $Z$ from a Gaussian, such that samples are overwhelmingly likely to fall within the unit hypersphere in $d$ dimensions (e.g. Euclidean norm of each $z$ is $\leq 1$). The restriction to a ball prevents the norm of a vector from being maximal along two dimensions at once, in effect encoding a prior according to which the strong presence of a given feature in an image excludes other features.
features (while being nonlinearly related to the image pixel space), so that variations in the variables correspond to proportionate variations in the features (cf. fn.153 above).

The systems just discussed model processes in which information from one sensory modality is brought to bear directly on a task involving another. But it may be wondered how this common representational space could be shared effectively in more complex cases. For instance, I may visually perceive one thing while thinking about another. To enable this, some sort of “multiplexing” (as it is discussed in the context of communications engineering) must be involved that allows for distinct signals to be carried in the same channel simultaneously, without confusion.¹⁶³ To accommodate this, we may perhaps borrow the idea of vector superposition from VSAs (see e.g. §2.1).

There is a worry here, however: the way things were described in §5.2, at any rate, only one proposition can be represented in a given vector space at a time. Words and other input cues are, according to the description in (Elman 2014), to be conceived of as perturbing a previously existing representation. It may be possible to devise some system in which some signals perturb while others are merely superposed, but a simpler solution is ready to hand: perhaps when I am thinking about one thing and doing another, the total representational content of my mental state at that moment is simply a conjunction of both contents.

The phenomenological and functional differences between this kind of case and a case in which I simply entertain a conjunctive thought may be chalked up to the unrelatedness of the conjoined contents. There are intermediate cases that may be explained by degrees of overlap between, say, perceived and imagined contents: imagining a cat scampering on the desk that I’m looking at is more like seeing a cat scampering on the desk than is, say, imagining a cat

¹⁶³ Imaging studies on multitasking (Marti et al 2015) suggest, however, that simultaneous mental processes can be carried out in parallel only for a short time.
scampering on a desk while I’m riding the subway. This is a topic that I intend to explore in future research.
6.1 Inference, association and vector spaces

I turn now from language processing, broadly construed, to a look at how connectionist networks might model the pillar of higher cognition: reasoning. The question whether, and if so how, representations within connectionist systems can be used to subserve reasoning is close to the heart of Fodor and Pylyshyn’s concerns about connectionism, and as such it has come up already under several guises (see especially §§2.6, 2.8, 5.2, and 5.3). Before diving once more into the details, I’ll address the basic and pervasive worry that connectionism is just mathematically sophisticated associationism, and therefore necessarily inadequate as a model of higher cognition (cf. Chomsky 1959).

One might reply to this charge simply by denying that connectionist architectures must be associationist. However, consider again the skip-gram model of (Milokov et al 2013c), which I’ll take for the moment as a proxy for a large host of related views discussed in this dissertation that appeal to vector space representations. There, word-level representations were learned by backpropagating from the prediction error incurred when the presence of one word is used to predict the presence of another. Predictive connections between the representations of words that occur in the same contexts in the corpus are strengthened, while connections between those that don’t undergo a form of extinction ($p(w'|w)$ is increased if $w'$ occurs in the context of $w$ in a given training example, and is decreased otherwise). That this training regime is a matter of associative learning is, I think, clear.\footnote{Provided, at least, that these probabilities are put to use in the natural way in a cognitive system that makes use of the learned word embeddings: viz., tokening $w$ is likely to lead to the tokening of $w'$ in proportion to $p(w'|w)$. This is, for what it’s worth, how word embeddings do tend actually to be used in connectionist language models.}
There are indeed some ways in which the functionality of these types of systems transcends standard associationist principles. It was argued in §1.3 that vector space representations predictably lead to systematicity in a way that mere associations do not. Instead of a single associative link, each word-level representation in a VSM encodes associations of varying strength between that word and every other word in the vocabulary, corresponding to the corpus statistics. And in systems that use recurrence to construct phrase-level representations, rational predictions may be made for novel items, as in the RAAM system (§4.1) and Google’s neural machine translation system (§5.4). This clearly goes beyond classical associationist learning per se: because of the semantic density of the vector space, the hardware guarantees that if some associations hold, others must as well. Vector space representations in effect allow for interpolation: because representations for previously unobserved items are implicitly defined given their semantic similarity to known items together with the structure of the space, associations involving these items are implicitly defined as well.

The difference between this system and a paradigmatically associationist one is, however, perhaps a matter of degree. Even classical associationism presupposes a type of generalization in that multiple observations contribute to modifying the same associations despite differences of detail. Some capacity for abstraction, and thus an implicit similarity metric, is assumed. Put differently, associations are defined over discrete contents, with requisite categorical distinctions presupposed. What connectionist models perhaps show is that within-category generalization and cross-category abstraction can both be subserved by the same mechanism, a representational space whose structure relates categories to one another implicitly as well as relating instances. Yet this structure remains an extrapolation from observed statistical relations. For these reasons, I assume going forward, at least for the sake of argument, that connectionist learning and representation is indeed associationist.
If this is so, the argument that an associationist system cannot account for higher cognition must be addressed directly. The groundwork for this argument has been laid in previous chapters, in which I attempted to show that at least very many cognitive functions that may have seemed beyond the reach of an associative treatment, such as translation and parsing, are well modeled by connectionist networks, and to explain the representational principles whereby this is possible. But there remain arguments, including more or less a priori arguments rooted in Classical assumptions about cognitive architecture, motivating the view that inferences cannot be associations. Someone convinced by such arguments might suppose that purely associationist systems must fail as accurate models of the mechanisms of human cognition, no matter how closely they may approximate its input-output profile. In the remainder of this section, my task will be the largely negative one of rebutting such arguments. In the next section, I’ll argue that a conception of inference as essentially tied to rules responsive to Classical constituent structure cannot encompass induction, and that in any case a superior alternative is to take formal logical rules as special cases of semantic entailment.

As discussed in (Kiefer 2017), Paul Boghossian, following (Harman 1986), offers an intuitive characterization of inference as a “reasoned change in view”: a process “in which you start of with some beliefs and then, after a process of reasoning, end up either adding some new beliefs, or giving up some old beliefs, or both” (Boghossian 2014, p.2). Of course, this definition of inference is only as illuminating as one’s antecedent understanding of the phrase “process of reasoning”, which is arguably the heart of the notion of inference.

One way of attempting to define reasoning (and thus, inference) is by appeal to logic: reasoning, one may argue, is the process of transitioning from one thought to another in a rule-governed way, where the rules in question are those prescribed by some system of logic (see Quilty-Dunn and Mandelbaum 2017). A bare appeal to logic to define reasoning may lead
to circularity, particularly if one is open-minded about which is the relevant system of logic, but
one may hope to pin down the relevant notion of reasoning by appeal to formal systems of a
certain sort. Accordingly, Quilty-Dunn and Mandelbaum unpack their notion of inference as
follows: A “bare inferential transition” (BIT) between mental representations \( A \) and \( B \) is one that
results from the application of some rule sensitive to the constituent structures of \( A \) and \( B \).

For present purposes, what matters about this definition of inference is that it serves to
drive a wedge between inference and association, assuming (as Quilty-Dunn and Mandelbaum
do) that associative transitions are not structure-sensitive rule-based transitions of this sort.
Accompanying this way of distinguishing inferential transitions from associative ones is a claim
about how such transitions may be modified: inferential transitions can be affected by the giving
of reasons, counter-arguments, and the like, while associative transitions can be modified only
by counter-conditioning or extinction. “Roughly, if S associates \( A \) and \( B \), one breaks that
association not by giving good reason not to associate \( A \) and \( B \), but rather by introducing \( A 
\) without \( B \) and \( B \) without \( A \). A link between two concepts that cannot be affected by any amount
of extinction or counterconditioning is \textit{ipso facto} not an associative link” (Quilty-Dunn and
Mandelbaum 2017, p.5).

I’ll begin my reply with this second way of distinguishing inference from association. The
problem I see here is that modification of a transition by counter-conditioning is a case of
modification by the presentation of evidence. Thus, the dichotomy implicit in “not by giving good
reason not to associate \( A \) and \( B \), but rather by introducing \( A \) without \( B \) and \( B \) without \( A \)” is a
false one. Observing \( A \) without \( B \) (or \( B \) without \( A \)) \textit{is} a good reason not to associate \( A \) and \( B \) (or,
at least, to decrease the strength of an existing association).

If this isn’t obvious, consider that if the mental attitude involved in the representation of a
given content is assertoric, and assuming that attitude is preserved across inferential transitions
(as seems reasonable), assertorically tokening A will tend to lead one who harbors the
association in question to assertorically token B. But if A may occur without B, then in effect
assuming that B is the case whenever one independently takes A to be the case is likely to lead
to a false belief. By presenting A without B, one is in fact giving evidence that B may occur
without A, and thus a good reason indeed not to assert B whenever A is asserted. Moreover,
a link between concepts not responsive to any amount of extinction seems unlikely to deserve
to be called an inference, either: if no sample of P’s without Q’s is large enough to convince me
of the falsity of P → Q, my belief in the latter hardly seems rationally revisable. So both
inferences and associations fall on one side of the relevant distinction among possible mental
transitions.

This does not establish that counter-conditioning and evidence-based modification of
dispositions to infer are just the same thing. Perhaps there are cases of the latter that are not
cases of the former. Quilty-Dunn and Mandelbaum suggest one candidate: if I “learn” that
bananas are actually red and that I’ve been the subject of an elaborate hoax about their color, I
will (ceteris paribus) cease to draw relevant inferences, and this does not seem to be a case of
counter-conditioning or extinction. If counter-conditioning is just one type of evidential
modification of mental-state transitions among others, then perhaps it can still be used to argue
for a joint in nature between association and inference, though on slightly shifted ground. But,
on reflection: how could I become convinced, on the basis of some evidence, that bananas are
not in fact yellow, without being presented, either directly or indirectly, with a situation in which
bananas are not yellow (and thus without being affected by some counter-conditioning)?

---

165 In fact, assertoric attitude is inessential here. Even if one is merely considering hypothetical relations
between states of affairs, one would not (ceteris paribus) want one’s mental model of a situation to
involve a connection between A and B if one has evidence that there is no such connection. But
assertoric attitudes serve to make the point most simply and vividly.
Let me unpack this. The most obvious way in which I could learn of the true nature of bananas would involve being presented with a red banana. But even if I am merely informed that there is such a banana verbally, by a person whose testimony I consider reliable, and along with corroborating evidence, such as a description of the means of my previous deception about the color of bananas, this will lead me to think about a situation in which there is a red banana. I am invited to at least entertain (and perhaps to assertorically employ, if I believe my source) the concept BANANA without the concept YELLOW, which ought to weaken the relevant association precisely by, in effect, counter-conditioning (at the level of representations).

To elaborate just slightly, drawing on Hume's paradigmatic associationist theory of mind (Hume 1748): Hume supposed that contiguity, resemblance, and cause and effect (which amounted for him, roughly, to constant temporal conjunction) were the relations that tended to lead to associative links (Mandelbaum 2017, §3). On one way of thinking about association, these relations obtain in the first instance among the associated objects (properties, states of affairs, etc) in the world, i.e. among the contents of the relevant representations. But presumably the mechanism by which the representations actually get recruited into an associative structure involves isomorphic relations obtaining among the representational vehicles themselves. This suggests that what really drives associative learning is not, strictly speaking, the degree to which two environmental events are statistically correlated, but the degree to which their representations are so correlated.

---

166 I assume that the cat is by now well out of the bag, but it is perhaps worth mentioning here that contrary to an at least erstwhile popular misconception, strength of association from A to B does not depend on frequency of past co-occurrence of A with B, but rather on degree of statistical dependence of B on A (Rescorla 1988). What this amounts to is that if B occurs often without A, then it will not become strongly associated with A even if it always occurs when A occurs.

167 In making this claim, I disagree wholeheartedly with Gallistel and King (2010, p.196) about the anti-representational nature of associationist theory. In any case, even an anti-representational associationist would need to posit internal states that mediate hierarchical associations, as discussed in (Rescorla 1988), upon which the argument given here could trade.
The difference between paradigm cases of evidential modification of beliefs and such association-forming (and breaking) mechanisms as conditioning is further obscured once it is acknowledged that conditioning needn’t be a gradual process involving many exposures to a given stimulus pair. This heads off the objection to what I’ve just been saying that being informed about the redness of bananas will (if I believe my source) instantaneously change my belief, whereas counter-conditioning would take a while.\textsuperscript{168}

Here is Rescorla (1988), from a review of then-recent developments in the theory of Pavlovian conditioning and ways in which they buck common misunderstandings about it: “It is a commonly held belief that Pavlovian conditioning is a slow process by which organisms learn only if stimulus relations are laboriously repeated over and over again...However, this view is not well supported by modern data. Although conditioning can sometimes be slow, in fact most modern conditioning preparations routinely show rapid learning”, including one-trial learning (cf. §6.6 below).

What of apparent counterexamples to the rationality of association? Isn’t it just clear as day that associative transitions, even between propositions, have nothing to do with reasons, since I may associate TRUMP IS PRESIDENT with THE SUN WILL EXPLODE (Quilty-Dunn and Mandelbaum 2017, p.5)? If so, how can I claim that counterconditioning is a case of evidential responsiveness? But I submit that these cases are of a piece with cases in which spurious evidence is encountered. The apparent counterexamples involve unusual circumstances or scenarios deliberately contrived to mislead. But of course, updating of beliefs by evidence is likely to fail (to lead to arbitrarily absurd beliefs) in these same circumstances.\textsuperscript{169}

\textsuperscript{168} One may suppose that the rapidity with which my beliefs are altered is proportional to the assumed reliability of the source of evidence that bananas are actually red, with first-hand perception and the testimony of reliable witnesses being at the top of the list. See the next paragraph on rapid (counter-)conditioning.

\textsuperscript{169} One might object that I can certainly come to associate two propositions without believing either. Thus, the comparison to cases of misleading evidence fails. But while the cases are certainly different, I
The arguments just given attempt to undermine the distinction between conditioning and the presentation of evidence as means for modifying mental-state transitions.\(^{170}\) The first, more basic way of distinguishing inference from association appealed to the sensitivity of inference to constituent structure. We’ve seen that connectionist processing \textit{is} sensitive to the structure of internal representations, though not in general to its Classical constituent structure, and it may be added that it’s common ground between connectionism and Classicism that processing is sensitive to \textit{content only} by way of being sensitive to structure. One could insist that genuine inference requires Classical constituency, in which case I’d be tempted to reply that genuine inference in that sense may not exist. In any case, I will look in the next section at reasons to doubt that inference \textit{can} be defined by appeal to Classical constituency.

I’ll consider one more argument in favor of the view that association and inference are distinct sorts of processes. Consider the fact (Reverberi et al 2012, cited by Quilty-Dunn and Mandelbaum) that subjects presented with a statement of the form ‘If \(p\) then \(q\)’ had \(q\) primed when presented unconsciously with \(p\), but not vice-versa. Quilty-Dunn and Mandelbaum adduce this as evidence that inference may occur unconsciously, but it might also be argued that this asymmetry can’t be explained in terms of association, since \(p\) and \(q\) should be more or less equally associated with one another. However, one would expect the patterns involved in such highly stereotyped syllogisms to be overlearned, even if the mechanisms underlying inference were purely associationist.\(^{171}\)

\footnotesize
\(^{170}\) I hedge here because one might still suppose that conditioning proper always involves direct presentation of counterexamples to the conditional corresponding to any association (i.e. ‘If \(A\) then \(B\)’), while other forms of evidential persuasion may involve only causing one to token the relevant internal representations, without \(A\) or \(B\) necessarily being present. This is a difference, but I don’t think it’s much of a peg on which to hang an inference/association dichotomy.

\(^{171}\) This depends on the assumption that associations are not by nature symmetrical. I won’t argue that point here, except to stress that the types of transitions to which I’m really committed are those that occur in connectionist networks, and those transitions are surely one-way in the general case. See (Kiefer 2017) for further discussion.
6.2 Induction and semantic entailment

I now consider some more substantial reasons for doubting that inference is a natural kind whose nature involves sensitivity to constituent structure. One problem is the type of inference Quilty-Dunn and Mandelbaum (2017) call "semantic entailment". How can we explain the typical transition from ‘This apple is red’ to ‘This apple is coloured’, since it doesn’t seem to be of the right form to activate any reasonable syntactically-sensitive logical rule? Quilty-Dunn and Mandelbaum stick to their guns and claim that unless the transition involves a suppressed conditional linking the first statement to the second, turning it into a syllogism, it is not inferential. If it so happens that the transition is based on an association between RED and COLOURED, for example, then it is simply an associative and not an inferential transition.

This consequence strikes me as a *reductio* of the inference/association dichotomy. It seems clear that common sense would count transitions like that from ‘This apple is red’ to ‘This apple is coloured’ as inferential so long as they were systematically employed in thought in the right way. This is not to say that it is impossible that such a transition could turn out not to be inferential, but what would be decisive here would be whether the mental transition were part of a contextually intelligible mental process or rather some kind of accident.

Further, consider what the difference between an associative and a BIT-style explanation of semantic entailment amounts to. On the BIT account, I must explicitly represent a premise such as ‘If x is red then x is coloured”. On the associative account, I directly transition from ‘This is red’ to ‘This is coloured’ in virtue of associating these propositions. But it is also plausible, especially if the association between these propositions is mediated by an association between the constituent concepts RED and COLOURED, as Quilty-Dunn and Mandelbaum suggest, that the fact that this associative transition exists amounts to an implicit encoding of the
content “If x is red then x is coloured”. There may be nothing to choose between these two
stories.

In any case, compositional structure cannot be used to define inferential transitions in
general unless the definition also covers inductive inference. It seems highly unlikely a priori
that syntactic rules governing simple inductive inferences will serve to distinguish them from
associations (see Kiefer 2017, §2.4). There is a long history of proposals, however, according
to which the evidential relations at work in inductive inference are weakened or generalized
forms of deductive consequence (e.g. Carnap 1950, Reichenbach 1949). Quilty-Dunn and
Mandelbaum align their account with a recent variant of this sort of proposal (Goodman et al
2015), which offers a computational framework that combines Bayesian inference with classical
symbolic processing, yielding a probabilistic version of the LoT hypothesis—see §6.5 below).

Broadly, the perspective on induction that Quilty-Dunn and Mandelbaum endorse, and
that naturally complements the BIT account of inference, is that induction is essentially a matter
of the application of logical rules, where the addition of probability operators to the formal
language accommodates the uncertainty associated with ampliative inference. However, I don’t
think that the way in which probabilistic inference is modeled in these sorts of systems (or any
workable alternative) supports the view that inductive inferences are formal operations sensitive
to (Classical) constituent structure.

We can examine how such simple inductive inferences would be implemented in a
PLoT, taking as a guide the particular formal language (Church, a stochastic lambda-calculus)
discussed in (Goodman et al 2008, 2015). The core mechanism for probabilistic inference in
Church is the inclusion of stochastic functions, which enable a program to branch not
deterministically but according to the sampled value of a random variable. Standard logical,
rule-based inferences could be accomplished in Church just as they would be in any typical
formal logic. Boolean truth-functions have their usual definitions, and various operators such as and, or and not can be used to construct functions of arbitrary symbols. In this way, the semantics of the logical connectives can be understood in terms of simple programs that map inputs to outputs in a truth-functional way.

Probabilistic inferences, on the other hand, depend on stochastic functions. They are handled in terms of conditioning: a prior probability distribution, combined with a particular piece of evidence (a condition), yields an output distribution. What Church provides in this regard is a clever method (called “rejection sampling”—Goodman et al p. 6; see also QUERY in Freer et al 2014) of sampling from the relevant conditional distribution (the posterior, in Bayesian terms) by combining the sampling of random variables according to the prior distribution with standard if-then branching routines (i.e. conditionals of the form treated in deductive logic).

In more detail: the key operation in Church for defining conditioning is the query function. query takes as arguments definitions of the terms that appear in it, a prior probability distribution specified in those terms, and a condition-statement, also specified in terms of the relevant definitions, that acts as the bit of evidence on which the resulting distribution is to be conditioned. Rejection sampling involves sampling from the prior distribution, feeding the result as input into the condition or “predicate”, which returns TRUE or FALSE depending on the value of the sample, and returning the sampled result once the predicate “accepts” the sample (i.e. returns TRUE).

This process of conditioning on a distribution, via rejection sampling, is identical in its input-output profile to standard Bayesian belief updating. There are alternative ways of modeling such updating computationally that depend only on straight Bayesian mathematics, without the detour through rejection sampling. Thus, we are in a position regarding inductive
inference that is analogous to that arrived at with respect to semantic entailment: while it is possible to understand inductions in terms of the application of formal rules, it is not clear that doing so yields any theoretical advantage, as against understanding them as, simply, associative connections, as I'll try to show over the next few paragraphs.

We could model the inference from 'It is raining' to 'The streets are wet' in Church, something like as follows:

```
(query
  (define is-raining (lambda () if (flip 0.1) True False)
  (define street-is-wet (lambda ()
                      If (is-raining) (if (flip 0.99) True False) (if (flip 0.02) True False))
  street-is-wet
  (= is-raining True))
```

This short program samples a truth-value for is-raining from a prior distribution (arbitrarily set to a 10% chance of rain here), uses this to determine the value of street-is-wet (which is taken to have p(0.99) when it is raining, and p(0.02) otherwise), and accepts the sampled value of street-is-wet only if it is consistent with is-raining being true.

Alternatively, we could model this inference, in a Bayes net, by sampling values of A (= "it's raining") and B (= "the street is wet") from the distribution corresponding to the following graphical model:

```
A   B
O———O
p(A) = 0.1
p(B|A) = 0.99
p(B|~A) = 0.02
```
One way of implementing this would be to use a stochastic neural network like that described in (Hinton and Sejnowski 1983), whose representations surely lack constituent structure.\textsuperscript{172}

As in the case of semantic entailment, I would argue that “It’s raining” → “The streets are wet” would pre-theoretically be considered an inference regardless of the mechanism employed. But in this case, the difference between the mechanisms is even more slight. In the Church program, there is, in the definition of street-is-wet, the probabilistic analogue of an explicit representation of the conditional “If it is raining, then the streets are wet”. But, functionally speaking, the same thing is accomplished in the Bayesian network by the directed edge from A to B together with the annotated probability. And of course this annotation is just an artefact of our description of the Bayes net: if implemented as in (Hinton and Sejnowski 1983), all the work would be done by the strength of the hardware connection from node A to node B.\textsuperscript{173}

Most importantly, however, for simple inferences such as this, based on learning from experience, the transitions in question will not be purely formal as modeled under the PLoT hypothesis, but instead will depend on the specific semantic values of the propositions related. The distribution over street-is-wet is defined in the Church program in terms of a logical if-then statement, but it also appeals explicitly and ineliminably to is-raining.\textsuperscript{174} Moreover, this is not an accidental feature of how induction would be modeled in Church. Relations of conditional probability are just not formal relations in the sense at issue. They are thus more like the local “rules” in terms of which semantic entailments could be characterized, for example that one may infer ‘--is coloured--’ from ‘--is red--’ in more or less any context.

\textsuperscript{172} To be clear: the model in (Hinton and Sejnowski 1983) is not a Bayes net in the sense of (Pearl 1988), but for simple Bayesian inference with a single piece of evidence the difference doesn’t matter.

\textsuperscript{173} Plus bias terms, to account for the probability of A and the effect of ¬A on B.

\textsuperscript{174} One could of course reference is-raining via some other symbol, but this does not affect the argument.
In fact, this idea points to an alternative way of understanding inference: we might take relations of conditional probability to be basic and to encompass semantic entailments, and construe the rules of formal logic as stemming from special cases of semantic entailment in which the semantics at issue is that of the logical vocabulary. This may sound outlandish, but there is reason to suppose that it can perhaps be pulled off if the semantics of logical expressions are modeled as operators on a vector space, as briefly discussed in §6.4 below.

Once Classical compositionality is given up, there remains, I argue, little theoretical motivation for distinguishing inference from association as a distinct type of internal process. This doesn’t mean that inference can’t be defined in a satisfactory way, however. Inferential transitions between thoughts (or other propositional attitudes) may be thought of as simply those regular causal relations between such states that tend to be truth-preserving, both actually and counterfactually (Kiefer 2017). Thus, the transition from A to B is an inference just in case (a) A and B are truth-apt, (b) B tends to be true just in case A is, and (c) the transition is rooted in some more or less stable disposition to token B upon tokening A.

The force of the qualification about counterfactual conditions is that an inference from \( p \) to \( q \) may count as truth-preserving so long as \( q \) would be true in all (or perhaps most) situations in which \( p \) were true, regardless of whether \( p \) is in fact true, or could be. This definition of inference could be cast in terms of rules, but the “rule” in question is not a formally, syntactically specified rule like those in formal logic systems. The rule would rather be something like “always preserve truth in transitions from one mental state to another”. Of course, the rules of formal systems like the propositional calculus are singled out because processes that operate according to them tend to respect this more general semantic Ur-rule.

The reasons that the rules of formal systems like the propositional calculus or first-order logic may be supposed to be truth-preserving is given in the soundness proofs for these
systems—technicalities aside, they rely on considerations about the truth-functionality of connectives, and in the case of first-order logic, the notion of models, truth under an interpretation, and truth under every interpretation. In each case, the reasons we can expect the rules to be truth-preserving rely on our antecedent understanding of key logical terms like ‘and’, ‘or’, ‘all’, and ‘not’. Given the understanding of these terms, the proofs are all but trivial.

Associations, on the other hand (those at least that are associations between propositions, as opposed to concepts like SALT and PEPPER) may be expected to be truth-preserving in worlds like those in which they were formed. This is simply because, given principles of associative learning, any association that has actually been learned must correspond to a genuine regularity in the world (those associations that don’t correspond to regularities would, given a large enough sample, be eliminated, in essence via extinction). This is not obviously a lower grade of epistemic respectability than that enjoyed by induction.

The qualifier about sample size is important: associations would be perfectly truth-preserving, relative to a specified domain, under the assumption of unlimited sample size (which also rules out a biased sample), but it’s possible by sheer bad luck for spurious contingencies to be reinforced that are really just noise due to limited sample size, i.e. “overfitting” (see Hohwy 2013, p.44) in statistical modeling (or perhaps due to the presentation of contrived stimuli in psychological experiments—see Quilty-Dunn and Mandelbaum 2017, p.5), and there is no reason to expect associative transitions learned in this way to be reliably truth-preserving. Moreover, one cannot simply idealize away from the vagrancies of sampling without idealizing away from the explanandum: the experience of the living organism and its ability to adjust its behavior in light of that experience, which is of the essence of empirical
cognition. Thus, while the positive non-Classical account of inference just sketched could stand further development, I will for present purposes consider associations to be inferential to the degree that they approximate the standard of reliable truth-preservation embodied in deductively valid inferences. This squares with intuitions about induction: paradigmatic generalizations from experience are clear cases of inference, but listening to *All Along the Watchtower* while smelling boiled potatoes and thereafter expecting the two to co-occur is not.

I don't pretend to have offered decisive arguments against a Classical construal of inference in this section. Any further adjudication of these matters would, however, seem to depend on the overall theoretical virtues of one or the other approach to the nature of inference and attendant background theories. I will have more to say about the PLoT hypothesis and its relations to Bayesian networks in the remainder of the chapter, but I have said enough for now to indicate why I take my minimal account of inference, on which associative links may also be inferential, to be viable.

### 6.3 Inductive reasoning in connectionist systems and the skip-thoughts model

Over the past few sections I had quite a lot to say about inductive reasoning and its relation to association. Here, I will consider a few ideas about how inductive reasoning, closely related to the probabilistic flavor that permeates many connectionist approaches to the mind generally, may be implemented in neural networks. In the next section I will consider several proposals as to how deductive, rigidly rule-based forms of reasoning might be accommodated.

---

175 I thank Eric Mandelbaum for pushing back against an earlier, more cavalier draft of this section in which I wholeheartedly endorsed such idealization. This turned out, as Eric tactfully indicated, to be in tension with my implication in the following section that some mental transitions are not inferential.

176 For more on idealization, inference, and the possibility of error, see (Kiefer 2017), §2.2.
First, note that in a way, all of the connectionist models discussed in this dissertation (or virtually all of them) have inductive reasoning built in from the ground up: I have argued in effect that the associative transitions in such networks are rational transitions, and computation in connectionist networks just is the unfolding of many such associative transitions in parallel. So, in a way, inductive inference is naturally accommodated.

But what about cases of explicit, conscious reasoning, in which premises are entertained and conclusions drawn with the full awareness of the reasoning subject? It is tempting to say that the difference between these cases and generic cases of automatic, unconscious inference (such as those that I’ve argued are involved in perceptual processing—see Kiefer 2017) is simply one of consciousness, and thus to pass the question off to the nearest theorist of consciousness. This would, however, be to miss the opportunity to address an important question within the connectionist framework. Granted that high-level neural codes are inferred, by a process more or less similar to inductive reasoning, from “premises” encoded in sensory input channels. Apart from the ongoing influence of perception on these representations, how should we expect sequences of them to unfold over time?

One possibility is to take the idea of a neural language model, exploited in approaches to word representation like skip-gram, and apply it to the level of sentence-sized contents (i.e. thoughts or propositions). Recall that skip-gram vectors, though employed as a means of obtaining statistically informed word-level representations in which proximity in vector space entails proximity in meaning, were originally motivated by the useful word representations learned by neural language models trained to produce sentences in which word transitions conform to corpus statistics. Well, inductive reasoning, at least, might then be modeled in terms of the “prediction” of one thought or sentence, given another (cf. the discussion of inference and association in the previous section). So a viable connectionist approach to inductive reasoning
might involve learning a higher-level model of the relations between thought-vectors, a special case of the kind of hierarchical learning ubiquitous in the kinds of models I’ve been discussing.

As it happens, this kind of model, called “skip-thoughts” or “skip-thought vectors” (by analogy to skip-gram) was proposed in (Kiros et al 2015) as a way of learning generically useful sentence representations in an unsupervised way from data. That is: the skip-thoughts model is trained on sentence-level corpus statistics so that sentences are given vector representations that depend on their typical contexts, e.g. neighboring sentences. One may have thought that the statistics at this level would be too sparse to learn much, but the authors show that skip-thought vectors, once trained, perform servicably, not matching state-of-the-art (which in this case is a version of the tree-shaped LSTM discussed in §4.3) but getting close, especially for an unsupervised method. In (Le and Milokov 2014), a similar approach is applied to variable-length phrases, through sentences and up to paragraphs.

In fact, the plot is somewhat thicker than this. Kiros and colleagues are interested not just in deriving good sentence representations from corpus data, but in deriving a reasonable composition function that can produce sentence-level vectors from combinations of word- and phrase-level ones. They suggest the skip-thoughts learning regime as a way of achieving this: beginning with one’s preferred set of word vectors, a composition function can be learned so that the sentence-level vectors that it yields are predictive of their sentential neighbors.\textsuperscript{177} Thus, this principled approach to the unsupervised learning of sentence vectors also furnishes yet one more approach to vector compositionality (see Fig.11). Interestingly, the model is an instance of the “encoder-decoder” framework that was seen to be operative in machine translation and multimodal inference models (§§5.4—5.6).

\textsuperscript{177} This combinatorial system of representations avoids the problem of sparsity in sentence-level statistics: because sentence representations are derived recursively from word-level representations, every possible sentence has a representation (and the set of such representations evolves during learning to enable sentence-level predictions).
More importantly for present purposes, the skip-thoughts model may offer the skeleton at least of an account of induction: just as a neural language model using word embeddings can be used to produce sequences of words that satisfy distributional constraints, sentence embeddings could be used to produce chains of reasoning (or, at least, streams of thought) like those found in texts. This proposal is open to serious objections as an account of inductive reasoning, however. The problem is not that it’s so crazy to think that we may learn to reason from paradigm cases: this may well be how reasoning is learned, if it is. The problem is rather that sentence-level transition probabilities are probably a poor proxy for inductive inferential relations among the thoughts those sentences express.

Apart from questions of irrationality, it’s just not the case that the flow of most prose directly mirrors the flow of reasoning. A system trained on such prose would be more likely to
model *narrative* than reasoning, and narrative transitions, while they sometimes depend upon reasoning, depend on much else besides. Among other issues, the premises and conclusion of an inference might not always occur in close proximity, and sentences may be juxtaposed for rhetorical or dramatic purposes, or simply because of the necessity of serializing the contents of thoughts for verbal expression, in ways that have little to do with reasoning.

Thus, while I am happy to suppose that grammar could (despite conventional wisdom) be learned by imitation of one’s linguistic environment, it stretches credulity to suppose that we abstract rules of inference in a similar way from sentence-level transitions. Almost all of the word-sequences one hears are sentences, but learning to infer on the basis of sentence-sequences would require knowing in advance which sequences corresponded to inferences, to allow one to select out the right data for imitation.\(^\text{178}\)

A more reasonable approach is taken in (Socher et al 2013c), in which a connectionist net is trained to perform “common sense reasoning” over a database of simple triplets of the form \((\text{entity}_1, \text{Relation}, \text{entity}_2)\) such as \((\text{cat}, \text{has-part}, \text{tail})\) and \((\text{Bengal tiger}, \text{is-instance-of}, \text{tiger})\). The network learns (1) a vector representation for each entity and (2) a function that assigns a score to novel relations, based on those it’s been trained with. The training is very similar to that used in (Collobert et al 2011) to learn word vectors: the network is shown both real triplets from the database and corrupted triplets with words randomly swapped out for others, and is trained to maximize the scores of the former and minimize those of the latter. The authors begin with pre-trained word vectors (in this case, those advertised in Huang et al 2012) and, similarly to the “occasion vectors” discussed in §5.2 above, create new vectors for complex entities such as ‘Bengal tiger’ by averaging the relevant word vectors. These “entity vectors”

\(^{178}\) This does not impugn the procedure for deriving sentence embeddings in the skip-thoughts model in the least. What the authors test for is semantic relatedness, and this, unlike inference, we may suppose to be captured in sentence-level transitions. Table 2 in (Kiros et al 2015) demonstrates the quality of the learned sentence embedding space.
are then composed with tensors (which, in this case, function a bit like higher-dimensional weight matrices, computing several different multiplicative interactions between the vectors simultaneously), which are learned for each relation type. They achieve ~90% accuracy at classifying novel relationships as likely to hold or not via this method (where classification is accomplished simply by thresholding a graded score prediction).

I call this approach “more reasonable”, but does it model inference? Certainly it shows that a connectionist network can assign rational “credences” to unobserved propositions on the basis of observed ones (the representation of these facts is somewhat simple and contrived, but in light of the progress in connectionist NLP discussed above, this may be regarded as a contingent limitation of the particular model). This assignment has the epistemic profile of induction, but the network does not seem to model an inferential process: there is simply a static (once learned) prediction function, encoded in the network’s weights, that assigns a score to each proposition expressible in the network’s language.

I think, however, that the process of learning in this network, and in many connectionist networks, provides a serviceable model of inference in the sense outlined in Harman (1986). According to Harman, inference in general, construed as a psychological process (he denies that deduction is its own species of inference), involves the simultaneous updating of all of one’s beliefs, to yield a new set, in a way that maximizes coherence while being as conservative as possible. There are various considerations that motivate this account of inference (see Kiefer 2017 for discussion), but to mention just two here: it naturally accommodates confirmation holism (updating one’s stock of prior beliefs in light of a new observation is just a special case of inferring), and it can be formalized in Bayesian terms.

Neural nets like that considered in (Hinton and Sejnowski 1983) may be construed as models of inference in Harman’s sense. In this model, minimizing a global energy function is
tantamount to massively parallel Bayesian inference (cf. Kiefer and Hohwy in print). Minimizing
the energy is the same as maximizing the probabilistic coherence of the representations: \( p(p|q) = p(P = 1|Q = 1) \), where \( P \) and \( Q \) are the states of binary stochastic processing units and \( p \) and \( q \) are the propositions they represent, and the probability of each node being "on" is a function
of its contribution to the global energy function. The energy monotonically decreases as the
network runs, unless external input is added, so the joint probability of the overall collection of
hypotheses is maximized.

Moreover a similar energy-minimization analysis\(^{179}\) is applicable in principle to most
connectionist systems, supervised and unsupervised alike (LeCun et al 2016). Since the
energy contributed by a given representation is a function of both the local weights and the
activities of other units, inference from data given a model and learning of the model itself can
both be understood in terms of energy minimization, and therefore as inference in Harman’s
sense.

There is one deeper worry about the ability of networks like this to model reasoning:
especially, it is just Fodor and Pylyshyn’s concern about the systematicity of inference. Given a
finite set of nodes in a graphical model, it seems that only a finite set of fixed propositions can
be represented, let alone inferentially related. I have discussed several recursive vector-based
compositional systems of representation above, which should allow for the representation of an
indefinite number of propositions in a fixed-sized neural network, but the implementations of
Bayesian inference I’ve just discussed have been defined for individual nodes, not for vectors.
It is of course possible to extend this kind of analysis to vectors, and thus to distributed

\(^{179}\) It is not generally the case, of course, that connectionist networks employ binary stochastic nodes
taken to encode the truth-values of propositions. But the relation between the energy contributed by a
given representation and its probability remains the same across many different models. And even if one
doesn’t take the contents of the relevant representations to concern probabilities, the description of a
network in terms of energy minimization places constraints on the set of mental-state transitions.
representations, so that if compositional VSMs exhibit systematicity in their representations, they should be able to deliver the systematicity of inference as well. I’ll return to this topic in §6.8, after discussing the “Probabilistic Language of Thought” (PLoT) hypothesis below.

6.4 Deductive reasoning in connectionist systems

Deductive logic is the Classical system par excellence: formal, rule-governed, and “hard” rather than “soft” (probabilistic). Most humans don’t seem to engage in much deductive reasoning, at least consciously or explicitly, but it is clearly something humans can do and therefore something that an account of cognitive architecture must make room for. I have argued throughout this chapter that it’s possible to disentangle the principles required to understand very many forms of intelligence, including systematic representation and inference, from Classical assumptions. But if formal operations over linguistically structured representations turn out not to be of the essence of cognition, then deduction in a way becomes a “problem case” for alternative models, just as induction was for a formalist notion of inference.

As Smolensky (1988a) notes, there are roughly two ways (at least) to think about the relation between Classical-style, “hard” rule-governed behavior and the “soft” probabilistic behavior of connectionist systems, assuming that both types of system model important (and perhaps closely intertwined) aspects of the human mind. The first approach is to suppose that rigid, categorical rules form the basic substrate of cognition, and that probabilistic features of thought arise out of the confusion of very many such micro-rules: “if there are enough hard rules sloshing around in the system, fluid behavior will be an emergent property” (Smolensky 1988a, p.139). The second is to suppose that cognitive systems are fundamentally “soft”, but that layered constraints conspire to yield the appearance, in some cases, of “hard” rules.
I think the second perspective is by far the more natural one, given that by default, the relations between hardware processors in a connectionist system are probabilistic or at least graded rather than all-or-nothing, and that it’s fairly easy to see how graded constraints could be used to approximate discrete behavior (to take just one simple example, a stochastic binary unit that receives total summed input greater than 4000 will be astronomically more likely to be in the “on” than the “off” state). It’s also eminently possible to derive messiness from many rigid rules (for example, probabilistic features of populations of molecules may emerge from in-principle deterministic interactions), but it’s not clear what these rules might look like in the case of connectionist systems.

One might have thought that, actually, there is a very clear paradigm for deduction within neural networks, moreover one that lends credence to the second alternative just discussed: it’s been known since at least the proposal of (McColluch and Pitts 1943) that logical operations can be modeled by small collections of simple neuron-like processors with thresholds and weighted connections (indeed, McColluch and Pitts were among the pioneers of what later came to be called the “connectionist” approach to computational modeling of the mind). An OR gate, for example, can be modeled by a simple network of binary threshold neurons, where neurons $P$ and $Q$ are connected via equal positive weights to neuron $P \lor Q$, which has a threshold equal to each incoming weight. If either $P$ or $Q$ is “on”, the threshold is met and $P \lor Q$ is activated.

I think it’s pretty clear that this kind of proposal, as McColluch and Pitts intended it, can’t work at the level of psychological modeling, only because the representations that must figure as the premises and conclusions of arguments are presumably more complex than simple

---

180 There is room for quibbling here: binary stochastic networks, which model spike trains, involve definite (though probabilistic) transitions between discrete states. And in general, one can certainly think of the basic, micro-scale processes in connectionist nets as rule-governed, problematizing the distinction being assumed here. But I trust that the big picture Smolensky means to paint is clear enough.
binary threshold neurons—in a realistic system that accommodated the systematicity of inference (among other things), those premises and conclusions would correspond, as we’ve been discussing, to distributed representations (e.g. sentence embeddings), while the McColluch and Pitts neurons were supposed to be, well, neurons.\textsuperscript{181} A similar proposal could perhaps be advanced for distributed representations. A partial implementation of deductive logic in distributed representations, based on VSAs, was considered in §2.8 above. However, we’ve seen VSAs not to be very promising as general-purpose models.

A different possibility for implementing deductive logic in connectionist systems, and particularly in those that incorporate generative models amenable to top-down influence, may be found in the work of Philip Johnson-Laird on the psychology of reasoning. Johnson-Laird pursues the hypothesis that, rather than being a formal theorem-proving sort of process, reasoning involves constructing iconic “mental models” of situations. In (Johnson-Laird 2010a, b) and elsewhere, a wealth of evidence is adduced for this hypothesis, including the content-sensitivity of certain inferences, as well as the fact that the difficulty of an inference scales with the number of distinct mental models required to determine what follows from a set of premises, among other things.

This view of reasoning is a natural fit for connectionist systems that independently traffic in mental models. If we already have a rich, complex generative model for keeping track of states of the world, why not use some of its capacity to model a situation we are interested in reasoning about? Interestingly, Johnson-Laird points out that premises that lend themselves to vivid visualization actually \textit{impede} or slow down reasoning. He also points to fMRI studies showing that only certain inferences (involving visually salient content) strongly activate visual

\textsuperscript{181} Ironically, McColluch-Pitts neurons ended up being very influential in the design of logic gates for conventional serial computers (see von Neumann 1958), but have largely been left behind as ways of directly implementing logic in contemporary connectionist networks.
cortex. This suggests that the mental models employed in reasoning may use visual cortex but typically employ only higher-level, more abstract representations.

No doubt, the details of this sort of account of deductive reasoning need filling out, to a greater extent than I can attempt here. Johnson-Laird expands on the idea as follows:

When we reason, we aim for conclusions that are true, or at least probable, given the premises. However, we also aim for conclusions that are novel, parsimonious, and that maintain information...So, we do not draw a conclusion that only repeats a premise, or that is a conjunction of the premises, or that adds an alternative to the possibilities to which the premises refer—even though each of these sorts of conclusion is valid. Reasoning based on models delivers such conclusions. We search for a relation or property that was not explicitly asserted in the premises. Depending on whether it holds in all, most, or some of the models, we draw a conclusion of its necessity, probability, or possibility...When we assess the deductive validity of an inference, we search for counterexamples to the conclusion (Johnson-Laird 2010a, p. 18244; internal references omitted).

I have already discussed (in §5.3) the possibility that concept composition may be accomplished by a “search” in which a neurally implemented VSM is allowed to settle into an energy minimum subsequent to receiving the relevant words as inputs, and (cf. §2.7, fn. 39, §2.8, and §5.2, fn. 126) the similarity or perhaps indistinguishability, in the models under consideration, between syntactic composition and inferential transition. In this case, we may imagine the search for counterexamples described here to operate via a mechanism similar to that posited for concept composition: a model is constructed for a premise (e.g. “Either it’s raining or I’m at the baseball game”) via vector composition, and modified by addition of a subsequent premise (“I’m not at the baseball game”). The result, if the system is working properly, will be a model according to which it’s raining (cf. the account of negation just below).

It is worth noting that in general, and even apart from Johnson-Laird’s proposal, the content-sensitivity of inference bodes well for the types of connectionist systems of representation and inference we’ve been considering, as against a purely formal, syntactic model. For example, Johnson-Laird points out that from the premises “All the Frenchmen are
gourmets, and some of the gourmets are wine-drinkers”, people are prone to incorrectly infer “Some of the Frenchmen are wine-drinkers”, but from “All the Frenchmen are gourmets and some of the gourmets are Italians”, people are prone to infer nothing in particular (Oakhill et al 1989). Since inference is a thoroughly semantically permeated process in VSMs and many other connectionist systems, this kind of effect is naturally accommodated in those systems. By contrast, a model of mental processes as probabilistic programs can’t explain this kind of effect, except insofar as they appeal to purely statistical mechanisms, leaving formal inference rules by the wayside. But if the formal part of PLoTs is doing any work at all, surely it should be invoked in the case of deduction.

That said, there are outstanding challenges for connectionist accounts of logic, related to the challenges in accounting for the semantics of logical terms that I mentioned in §5.3. While I take this topic to be a fairly wide-open area of research, existing accounts are very suggestive. For example, there is a simple, effective way of modeling negation in VSMs. In (Widdows and Cohen 2015), a binary negation operator, \( \mathbf{a} \text{ NOT } \mathbf{b} \), is defined as \( \mathbf{a} - (\mathbf{a} \cdot \mathbf{b} / |\mathbf{a} \cdot \mathbf{b}|)\mathbf{b} \), where \(|\mathbf{x}|\) is the magnitude of vector \( \mathbf{x} \) and \( \mathbf{x} \cdot \mathbf{y} \), again, is the inner product (i.e. the sum of the products of elements of \( \mathbf{x} \) and \( \mathbf{y} \), \( x_1y_1 + x_2y_2 + \ldots + x_ny_n \)). Here, the first term in the multiplication reduces to a scalar, so the operation as a whole simply subtracts a scaled version of \( \mathbf{b} \) from \( \mathbf{a} \). This is a projection of \( \mathbf{a} \) onto the subset \( S \) of vectors in the VSM that are orthogonal to \( \mathbf{b} \), i.e. the “closest” vector to \( \mathbf{a} \) in \( S \).

On this account of negation, NOT doesn’t have its own dedicated point (or region) in the vector space. Rather, it maps some other vector representation (\( \mathbf{a} \)) to a new representation that “factors out” the negated term \( \mathbf{b} \): the resulting vector (the projection) will be maximally different from \( \mathbf{b} \), while preserving as much similarity to the base term \( \mathbf{a} \) as possible. Picturing this in low dimensions is easy, and somewhat helpful: the negated term \( \mathbf{b} \) corresponds to a vector
perpendicular to a plane onto which the non-negated term is projected. From the point of view of this plane, \( \mathbf{b} \) is “invisible” (see Fig. 12). We might think of this as modeling a situation in which the state of affairs corresponding to \( \mathbf{b} \) does not obtain.

![Figure 12: Vector \( \mathbf{a} \) is projected onto plane \( S \), orthogonal to vector \( \mathbf{b} \), using the NOT operator. Approximate rendering in three dimensions. To interpret the diagram, keep in view that \( \mathbf{b} \) is perpendicular to plane \( S \).](image)

This may perhaps be taken as a geometrical interpretation of the proposal for the concept NOT introduced by Kamp and Partee (1995): although NOT RED has no prototype, we can think of something’s not being red as its being as dissimilar as possible to red things (by contrast, the way in which prototypes function to determine extension in simple, non-negated cases is that things similar to the prototype fall into the relevant class). Fodor and Lepore criticize this approach, and rightly so, as a way to save the prototype theory of concepts, but it might yet be salvageable as part of a non-Classical account of the meaning of NOT.

It may be wondered why negation should be defined as a binary operation. The reason, I suspect, is that some nonzero vector must already be present in order for a meaningful
projection to occur (i.e. if we apply the binary NOT operator using the zero vector for $\mathbf{a}$, the result involves division by 0). This is related to the Classical characterization of negation as a function mapping the extension of a negated term to its complement. In order for negation, so characterized, to have meaning, it’s not sufficient that it attach to a term. It is implicit that there is also something else in the complement set. In a universe consisting only of cows, it’s unclear what NOT A COW would refer to (I curtail Heideggerian speculation about the concept NOTHINGNESS). But in a universe consisting only of brown and red cows, the extension of NOT BROWN can be modeled as the projection of cow onto the space orthogonal to brown.

It’s important to consider how this account of negation may be squared with the idea defended in §5.2, that concepts are operators on a prior vector space representation. There, I supposed each word $w$ to operate on the pre-existing representation in the space, $\mathbf{x}$, such that the contribution of the word to the overall representation can be modeled as $f(\mathbf{x}) = \mathbf{y}$, for example $\mathbf{x} + w$, where $w$ is the word’s associated VSM representation. It is possible that in cases such as NOT, and perhaps more generally, the operation performed on the prior representation depends as well on the subsequent word in a sequence. Indeed, it is hard to see how an operator like NOT could perform a transformation like that posited by Widdows and Cohen without making use of the next word vector in the sequence. It may be independently reasonable to allow for some “look-ahead” in the composition function generally, as the models discussed earlier that employ bidirectional RNNs in effect do (see again Wu et al unpublished), such that this account of negation would be naturally accommodated. This may also provide the resources for a superior account of composition involving relational terms. I have remained non-committal here as to the precise composition function, and have suggested that in any case it will likely be a complex non-linearity computed by a recurrent neural network.
6.5 The PLoT hypothesis

This brings me, finally, to a square confrontation with the “PLoT” (Probabilistic Language of Thought) hypothesis. This hypothesis is best evaluated against the background of the connectionist approaches to cognition just discussed. It aims to distance itself both from them and from the original, purely symbolic LoT hypothesis. Like the Helmholtz machine and other connectionist models examined above, the PLoT hypothesis is a Bayesian, probabilistic generative-models-based approach that appeals to stochastic, sample-based processing. Unlike those models, it supposes that the representations deployed in psychologically adequate generative models will be richly structured symbols of various sorts rather than simply vectors encoded in the activities of neuronal populations.

I’ll begin in this section with a brief account of the PLoT hypothesis, distinguishing some commitments that are central to the PLoT perspective to differing degrees, and in subsequent sections I’ll compare PLoT models to connectionist proposals along various dimensions.

The definitive claim of the PLoT hypothesis, I take it, is that concepts are (stochastic) programs in a probabilistic language of thought (Goodman et al 2015). I’ve discussed this idea already in connection with inference, but to briefly recap: according to this proposal, thought takes place in something like a formal language, but one supplemented with machinery for sampling the values of random variables (yielding something like Church (Goodman et al 2008), a stochastic lambda-calculus). This leads to a way of modeling cognition that combines the universal representational scope, and inferential power, of Classical symbol systems, with the flexibility and subtle, graded behavior characteristic of probabilistic forms of reasoning.

A related idea is that the human ability to generalize to new concepts on the basis of very few examples is best explained not by purely bottom-up statistical learning, but rather by appeal to a form of Bayesian inference that relies on powerful structured priors (Tenenbaum et
al 2011, Kemp and Tenenbaum 2009), possibly in conjunction with simpler statistical approaches (Salakhutdinov and Tenenbaum 2011). These two ideas are combined in (Lake et al 2015), the guiding idea there being that very efficient inference might rely on the induction of probabilistic programs from data. Third, it is sometimes further claimed that some such structures, or principles for generating them, may be innate (Kemp and Tenenbaum 2008), mirroring the apparent nativist implications of the original LoT hypothesis (Fodor 1975).

In (Goodman et al 2015), it is demonstrated that Church programs can model many different types of probabilistic reasoning convincingly. By defining symbols in terms of stochastic programs that may refer to other symbols, elaborate functional relationships and hierarchically related “concepts” can be constructed. Probabilistic reasoning depends on primitive operations of sampling from specified distributions, and such distributions can be conditioned on the outcome of other processes. As discussed in §6.2, Bayesian inference can be accommodated via “rejection sampling” or similar means.

Of course, at least a large part of this representational power derives, simply, from the fact that Church is a high-level programming language. The idea that concepts are programs (whose semantics is given in terms of their interpretation, e.g. the way in which the processor handles them) is interesting in its own right, and of course suggests a functional role semantics for such concepts. Goodman et al suggest that Church programs can be understood in terms of a “sampling semantics”: since branching behavior often depends on the value of a random variable, a program defines a distribution over possible computations rather than a single

---

182 In this respect, proponents of the PLoT proposal seem to be more on board, philosophically, with connectionists (cf. the talk of implicit definition in Hinton et al 2011) than were Fodor and the older guard of Classicists. Part of the ease with which some Classically-minded philosophers brush off the matters with which connectionists are concerned as so many “implementation details” stems, I suspect, from leaning too heavily on a distinction between syntax and semantics (which in turn, I suspect, may be bolstered by an externalist, atomistic semantics). See the following section for (a bit) more on this theme.
stream of computation. This, the authors argue, potentially yields a powerful form of compositionality (Goodman et al 2015, p. 4):

...if you have described a deterministic process compositionally in terms of the computation steps taken from start to end, but then inject noise at some point along the way, you get a stochastic process; this stochastic process unfolds in the original steps, except where a random choice is made. In this way a distribution over outputs is determined, not a single deterministic output, and this distribution inherits all the compositionality of the original deterministic process.

The authors further argue that this flexibility outstrips that of simpler approaches, like Bayesian networks. Although hierarchical Bayes nets may exhibit a form of compositionality (as discussed above), they are limited to inferences over the particular variables represented by their nodes (p.13). On the other hand, Bayesian networks, along with many other more complicated structures, can easily be defined in Church.

While (Goodman et al 2015) lays out the conceptual groundwork for a PLoT formalism, it serves more as a proof-of-concept than a demonstration. In that work, simple programs are defined that serve as toy models of how reasoning in realistic domains might be accomplished, and of how various forms of reasoning relate to one another (for example, one’s degree of confidence in the laziness of a tug-of-war player, independently inferred, might influence the conclusions one draws upon observing the outcomes of games that player participated in). But in a realistic setting, the variables involved would have to be much more richly interconnected and functionally specified in order to deserve their names.

The work in (Lake et al 2015) puts similar programs to work within a “Bayesian Program Learning” (BPL) framework. Lake et al define a hierarchical generative model for characters drawn from various alphabets. It begins with small strokes of a pen (defined as parameterized splines), combines these into larger strokes, and combines the larger strokes into characters, which are then rendered as images. After training on a dataset, the program can generate characters from existing categories, produce new examples given an exemplar (see Figure 5 of...
the paper), and produce new characters similar to those in novel “alphabets” (see their Figure 7). Tokens produced by the program are generally visually indistinguishable from those created by humans (as verified quantitatively by a “visual Turing test”—see Fig. 6 of Lake et al’s paper).

The BPL model can also be used to invert the generative model and discover the simple “motor program” (set of strokes) used to produce a character, with high accuracy. Most impressively, it can learn new “concepts”, as demonstrated by the ability to correctly generalize from just one example. The BPL model achieved 3.3% error on a 20-way one-shot classification task (i.e. correctly picking a new exemplar of a class, given only one previous example as a “training” set), while humans achieved 4.5% on average at this task, and a deep convolutional neural network optimized for the task achieved 8%. Importantly, the BPL still achieved comparably low error rates (around 4%) when only a small fraction of the training data was used, but the neural net’s error shot up to ~20% under these conditions.

Lake et al also note that, quantitative performance aside, deep learning models typically learn representations dedicated to just one task. The average CNN for image classification, for example, is not optimized as a generative model. As the authors put it: “A central challenge is to explain these two aspects of human-level concept learning: How do people learn new concepts from just one or a few examples? And how do people learn such abstract, rich, and flexible representations? An even greater challenge arises when putting them together: How can learning succeed from such sparse data yet also produce such rich representations?” (Lake et al 2015, p.1332). The BPL model seems prima facie at least to go further toward answering these questions than do standard connectionist models.

---

\[183\] This is a fair criticism of standard supervised classification networks, but as seen in Chapter 3, learned vector space word embeddings are good general-purpose representations, and such embeddings are by no means limited to the domain of natural language (cf. Chapter 5, and §§5.5—5.6 in particular).
One principle emphasized by Lake, Tenenbaum et al is “learning to learn”: programs induced from previous data can be used to more efficiently induce programs from future experience. However, more efficient learning does not come for free: the advantages of the BPL model over the CNN, for example, stem from powerful priors built into the model (as will be discussed in the following sections). While perhaps some of this prior information can be learned as well (see Lake et al 2015, p.1337), at least some of it is presumably built into the architecture innately (indeed, the local connectivity in CNNs is itself an “innate” prior built into the structure of the network).

While I do not take any kind of innateness claim to be essential to the PLoT hypothesis, innateness and the Language of Thought have long been friends. Fodor and Lepore (1996, p.351) say that the facts about compositionality reviewed in previous sections suggest that only two theories of lexical semantics are possible: (1) all lexical meanings are primitive, or (2) some lexical meanings are primitive and the rest are defined out of those. Add to this Fodor’s (Fodor 1975) argument that one needs “Mentalese” sentences in which to couch hypotheses about word meaning in order to learn a natural language, and, to avoid an infinite regress, one must take the primitive lexical items in Mentalese (including, famously, CARBURETOR) to be innate.

Kemp and Tenenbaum (2008) suggest a nativist hypothesis of a different stripe: they posit a “universal structure grammar”, whose form might be a meta-grammar that generates context-free graph grammars. Without going into excessive detail, a context-free graph grammar supplies rules for “growing” graphical models given very simple primitives (nodes, edges, and the like) and rules for adding these primitives to an existing graph. A meta-grammar would similarly generate graph grammars of this sort out of just a few primitives. The utility of this sort of system is clear: if Bayesian networks are a good idea, then a “grammar” that allows one to build them on demand is an even better one.
It should be noted that such an innate “graph grammar” may not be very far from the kind of nativism proposed by Chomsky (see e.g. Chomsky 2000) to explain language learning, since a generative grammar for syntax trees is a special case of generative graph grammars. The Chomskian and Fodorian innateness theses differ, however, in the following respect: the former supposes innate knowledge (or at least know-how) concerning how to parse sentences, the latter innate knowledge of meanings.

Of the two sorts of nativism, I think the former is more likely to survive the push toward data-driven modeling (and indeed, has been by far the more widely discussed sort in recent years). If there’s one thing that deep learning models have almost certainly gotten right, it’s that the brain plausibly implements powerful domain-general learning mechanisms to extract a hierarchy of modality-specific feature representations from sensory data (“representation learning”—see Bengio et al 2012). In general, Fodor doesn’t seem to have anticipated, or considered, the possibility of unsupervised learning of concepts, probably because he is committed (for reasons like those discussed in §5.3 above) to the view that concepts cannot be based on anything statistical.184

All parties to this debate in its contemporary form agree that learning is largely driven bottom-up by data. But proponents of PLoT question, in effect, whether (largely unsupervised) learning of representations can do all the relevant explanatory work on its own, and whether this kind of learning can account for the sorts of structures that plausibly constrain reasoning at higher, cognitive, amodal levels of representation. Given just a bit of innate structure, they propose, the bottom-up learning will work much more efficiently, and models incorporating such

---

184 It’s not entirely clear to me that Fodor need have rejected a statistical account of the learning of concepts, because such an account of concept learning might a priori be consistent with the view that concepts, once learned, do not contribute anything statistical to the meanings of complex expressions. That said, it’s not clear what a theory that divorces conceptual content from concept learning in this way would look like, or what would motivate it.
structure have a much better chance of capturing the way in which humans actually learn concepts (Tenenbaum et al 2011).

The old-fashioned Fodorian argument was that the only way in which we can imagine language-learning to work, at least presently, requires the positing of innate concepts (the “only game in town” argument). The argument just rehearsed, on the other hand, appeals to the idea that innate knowledge would simply make learning much easier. However, a large enough performance gap strongly suggests a difference in mechanism, and the data may indeed indicate that the process of training a neural net on millions of examples is not all there is to human concept learning.

6.6 One-shot and zero-shot learning

Space constraints allow me only one laudatory section on the PLoT hypothesis (the preceding one). I will now turn to some criticisms. In general, my complaint is that the PLoT hypothesis advertises a decisive “third way” between connectionist and Classical modeling, but what it in fact offers is a set of (very good) ideas some of which (the Classical ones) are obviously useful but not easy to implement in brains, and others of which are easy to implement in brains but (like stochastic sampling, program induction and learning to learn) are features of many connectionist models. That said, there are substantive issues in the vicinity. I begin with what I take to be a genuinely stiff challenge for connectionist models: one-shot learning.

To start, it is important to note that (Lake et al 2015) employ extensive hand-engineered machinery to achieve their one-shot classification results. In particular, to efficiently explore the space of possible generative programs for an observed character, they apply a heuristic

---

185 See e.g. (Rumelhart et al 1985 p.29).
procedure that outputs the most likely “parses” for the training image. These candidates are then used to parse test images by re-fitting the parameters that define tokens, given types, for each test image. The probabilities of the test images under the candidate parses can then be used to decide whether the test and training images belong to a common type.

The heuristic that finds the candidate parses takes about five pages to describe (see supplementary materials for Lake et al 2015, pp.11-15), and involves several subsidiary algorithms. First an image “skeleton” graph is derived from the raw image (a three- or four-stage process in itself), then a series of random walks are taken on this graph to generate the candidate parses at the level of strokes. A further process then searches for sub-strokes that may have composed the strokes in the candidate parses. Values of the variables defining the order, orientation, etc of the sub-strokes are then determined by an optimization procedure that maximizes the posterior probability of the image via a brute-force search, followed by a greedy fine-tuning phase. Random walks are then initiated for each candidate parse to estimate variability of the character type, and this estimate is used to approximate the posterior distribution over types and tokens for the test image.

The best-performing connectionist approach to the one-shot learning problem cited by Lake et al employs a “siamese convolutional neural network” (Koch et al 2015), which is a pair of identical CNNs whose top-layer feature representations are fed into a logistic (sigmoid) unit that outputs the probability that a pair of images fed to the twin networks are from the same class. The parameters of the networks are learned by backpropagation from the discrepancy between the prediction and the truth (identical classes, or no) for pairs of training images.

The inference procedure for one-shot classification in this model is: compute a single forward pass through the neural network for the training image paired with each of the test images, and choose the test image for which the network yields the highest output probability.
This process can be optimized using machinery designed for “mini-batch” learning to yield one-shot classification in a single forward pass.

This contrast is an example of why, despite the undeniably impressive results of (Lake et al 2015), my bet is still on deep learning systems as models of human intelligence. First, recall that the performance gap between these models is around 5%, and that both are close to human-level performance already. Second, as (Koch 2015, p.22) points out, the neural networks employed in this work are, while deep at 5-6 layers, shallow compared to cutting-edge models. More importantly, distributed representations within connectionist networks can be leveraged by more intelligent and still biologically plausible domain-general learning algorithms, as in Hinton et al’s work on capsules (see §4.6 above). That approach greatly improves on the efficiency of CNNs, and should enable better one-shot classification with a fraction of the training data.\(^1\) Finally, (Koch et al 2015) note that their model is trained on raw image data alone, while the model of (Lake et al 2015) is trained on images of characters as well as the stroke data for each character (in effect, the program used to generate the characters).

These considerations alone seem sufficient to me to militate decisively in favor of the connectionist models, given the vast advantage of the latter over the BPL system in terms of simplicity and, relatedly, transparent biological implementation (see next section). I’ll close this section by discussing a handful of connectionist models that naturally employ the multimodal embedding spaces discussed above (see §§5.5—5.6) to enable zero-shot learning (a pure example of “transfer learning”, in which knowledge from one domain is applied to another, by necessity). From there, one-shot learning should be a breeze.

\(^1\) It also arguably involves a concession to Classicism, since it imposes an additional layer of structure in the representations, though I would argue that this structure is just that made available by distributed representations in ordinary connectionist systems. The novel features of capsules networks (e.g. dynamic routing) are thoroughly in the spirit of connectionism as Fodor and Pylyshyn conceived of it.
Zero-shot learning is enabled, for example, in the model of (Socher et al. 2013d), as follows. First, distributed word vectors are learned as discussed above (§§3.3—3.5), creating a word embedding space. Images whose class corresponds to a word in the embedding space are then mapped to that space by learning a simple two-layer neural network that minimizes the squared distance between the image projection and the relevant word vector. From there, a two-step process is used to classify novel images (from classes unseen during training).

First, outlier detection methods are used to predict whether a novel input belongs to a previously observed class, or not. The rough idea is that an image should be classified as belonging to a novel category if its probability under each of the Gaussians centered on the known classes is under some threshold (where the variance is estimated from the set of training images with the corresponding label). Second, if the image is classified as novel, it is assigned to the label for unseen classes whose corresponding Gaussian maximizes its probability. Using this method, the authors obtain accuracy for zero-shot classification of up to 90% when two categories are held out from the training set. The package is a (perhaps crude) model of the process whereby an unseen object is categorized based on verbal descriptions.

Frome et al. (2013) take a similar approach, but improve on the model in (Socher et al. 2013d) in two important respects: (1) they scale up to a situation involving zero-shot inference over 20,000 unseen classes given only 1,000 in the training set (based on the ImageNet dataset), and (2) they replace the outlier classification method of (Socher et al. 2013d) with an elegant alternative: after pre-training of unsupervised word vectors (ala Milokov 2013b) in a

---

187 To be fair, the highest accuracy is obtained for the pair cat-truck, the lowest reported accuracy, around 50%, for the pair cat-dog, but this is still far above chance for 10-way classification.
188 As it happens, the interest of this kind of model (see also Frome et al. 2013) extends well beyond the problem of zero-shot learning. It demonstrates how visual grounding of certain concepts and descriptive aspects of meaning might interact. It is because the space of word embeddings is structured by linguistic corpora in a way that, as discussed above, implicitly captures the roles words play in the language and, via structural similarity, the world represented by the language, that the mapping learned by the network from images into the embedding space is sensible even for unseen categories.
language model and discriminative pre-training of a standard computer vision network, the classifier at the “top” of the vision network is removed, and the network is re-trained to predict the word vector associated with each input image (using a loss function that maximizes similarity between images and their labels, as in for example Karpathy and Fei-Fei 2015), but employing a contrastive ranking criterion conceptually similar to that in (Collobert et al 2011). The result is a system that leverages the multimodal embedding spaces discussed in Chapter 5 to provide a principled and effective approach to the prediction of unseen classes: the network generalizes from the mapping it has learned between example images and the word vectors for their class labels to “translate” between novel images and their descriptions. Performance on zero-shot classification was favorable compared to benchmarks, yielding a 10% top-5 hit rate for 1000-way zero-shot classification (versus 2% for competing systems), and a 2.5% top-5 hit rate on around 21,000 classes (where guessing yields an expected 0.0002%).

In the model of (Palatucci et al 2009), a strategy very like that discussed in (Tenenbaum et al 2011) is employed to classify novel items without cross-modal transfer. High-dimensional raw input is first mapped to a skeletal, relatively abstract feature-based representation, which is in turn used to predict class labels, given a database of knowledge about the mappings between the feature representations and the labels (which is richer than the input data). This scheme works because the first mapping, from inputs $X$ to semantic marker structures $S$, can be learned independently of the mapping from structures $S$ to labels $L$. Then, never-before-seen inputs will (in a good classifier) map to $S$’s that are near points in the semantic space that can reasonably predict categories, given the previously learned mapping to labels.

Another example of “zero-shot” transfer learning in connectionist models occurs in the Google neural machine translation model discussed earlier (Wu et al unpublished), in which sentences from previously observed input languages can be mapped to a target language
without any previous translations between those two languages having been performed—the new mapping is based purely on abstract, language-invariant knowledge (or at least, knowledge that does not vary within a language family) derived from other language pairs.

6.7 Biological plausibility

The considerations of simplicity and efficiency that I invoked in the previous section in favor of a connectionist approach to one-shot learning do not depend on the details of that task, and apply quite generally. First, although priors over complex structures are no doubt useful during inference, the inference algorithms they require are nowhere near as fast as a single pass through a deep neural net (they typically employ slow iterative sampling methods like those in the heuristic for program induction mentioned in the previous section).

It’s not that it’s out of the question that such sample-based methods might turn out to be effective and neurally plausible—indeed, the Gibbs sampling method employed in (Hinton 2005), discussed above as a minimal model of multimodal mappings, is of this class. It’s just that, absent very compelling reasons to reject connectionist approaches, the slowness of these methods as applied to “richly structured” models (see Tenenbaum et al 2011, p.1284), together with the lack of a transparent neural implementation, puts them at an explanatory disadvantage.

Though precise implementation details for some aspects of connectionist models are up for grabs, they uniformly stick pretty closely to the neural facts, in marked contrast to Bayesian programs. This is true of learning as well as of representation: a large part of the appeal of the unsupervised learning algorithms I’ve been discussing throughout this dissertation (for example the energy minimization methods) is that they fall naturally out of an attempt to model the intrinsic properties of neurons and their interactions. In these frameworks, equations governing
learning (the “epistemic dimension” of network dynamics) can often be taken as well as
(somewhat loose) descriptions of physical dynamics (see, again, (Hinton and Sejnowski 1983),
for a particularly striking example).

It may be objected that backpropagation is an implausible learning mechanism. Indeed,
I myself have insisted on unsupervised learning wherever possible. But while backpropagation
is psychologically implausible as a model of learning in most domains, there is nothing
biologically implausible about it. The most unrealistic feature of backprop, its requirement that
error messages be passed “backward” along synaptic connections (requiring, in effect, a
backward weight corresponding to every forward one), has fairly recently been shown to be
dispensable (see Lillicrap et al 2016): collections of random feedback weights allow for an
approach to learning (“feedback alignment”) that is nearly as efficient. Further, as discussed
previously, systems that employ supervised learning can be described in the terms of an energy
minimization framework (LeCun et al 2006), and work of Hinton’s\(^{189}\) shows that contrastive
divergence learning (the widely used layer-by-layer unsupervised learning algorithm discussed
in connection with Restricted Boltzmann Machines above) is roughly equivalent to
backpropagation within an autoencoder.\(^{190}\)

The closeness of connectionist representations to biology is, I would argue, intimately
related to the way in which connectionist nodes represent things, namely by taking part in a
simulation whose operation is isomorphic to a data-generating process. Any generative
models-based approach has the virtue of being able to explain in theory, at a psychological
level, how learning and inference are possible (i.e. by inverting the model, bootstrapping, etc).

\(^{189}\) The reference is to a talk called “How to do backpropagation in a brain”. Slides are available freely

\(^{190}\) The proof relies on the idea that a neuron can pass forward both its activation level and an error
derivative (with respect to a local cost) simultaneously, using temporal derivatives to encode the error.
This is intriguing as it suggests a simpler mechanism for hierarchical minimization of prediction error than
the functionally distinct neuronal subpopulations posited in (Friston 2005).
But such approaches only have the additional virtue of explaining *how the model can be implemented* to the extent that they stick closely to the neural facts.

Decisive arguments in this area are difficult, because what is at issue is a tradeoff among various theoretical virtues. Of course, building in strong priors will speed learning. Specifying an inventory of possible graph grammars (Kemp and Tenenbaum 2008) amounts to a very powerful prior, but one may hope to attain at least some of the advantage of this kind of hypothesis while sticking more closely to neuromorphic modeling. I have argued (§4.6) that the “capsules” framework (Sabour et al 2017, Hinton et al 2018), which is at heart a specific proposal about the use of distributed representations in connectionist nets in conjunction with a more powerful inference algorithm, strikes a promising balance.

In connection with these issues, it may be useful to compare the “Hierarchical Deep” (HD) model proposed in (Salakhutdinov and Tenenbaum 2011), a hybrid deep learning-nonparametric Bayesian approach, to a capsules network. In the former model, a hierarchy of discrete category decisions is made at the highest layers of the model based on abstract knowledge, which is used to select only a specified number of top-level nodes of a deep Boltzmann machine. In a capsules system, by contrast, the “one-parent” constraint is enforced at all levels of hierarchy, building a tree structure that extends down to the input layer.

Recent studies of the temporal evolution of neural codes suggest that task-relevant properties of a stimulus are maintained within a widely distributed cortical network, in which unneeded information is discarded after an initial stage at which all stimulus features are represented and processed in parallel (King et al 2016). This is to be expected in a capsules-like architecture, where a fast but temporally extended process of “dynamic routing” chisels tree-structures out of distributed representations at each layer with the help of feedback connections. In an HD architecture, the bottom layers of the model would not be expected to
exhibit such discrete behavior. Of course, the HD network was not advanced as a realistic
process model, but as a way of combining the virtues of deep learning and discrete category
hierarchies in a single framework for inference. And the evidence just cited cuts equally against
traditional feedforward connectionist networks. But the moral is perhaps that a capsules
network has the comparative virtue of being at once a reasonable approach to inference
problems on grounds of the cognitive demands of the task, and far more realistic as a process
model (though still rather crude compared to the real thing).

It may perhaps be objected to connectionism in general that it amounts to a particular,
overly simple style of brain modeling, with one eye increasingly on practical applications
unrelated to the core pursuits of cognitive science and artificial intelligence. Surely, it might be
supposed, whatever is right in contemporary connectionist models with respect to the human
brain will in the end be captured more precisely by models within computational neuroscience,
unencumbered by the demand for results that can be put to immediate use.

But, while practical applications are indeed paramount in the field of machine learning
and sometimes influence the direction of research, achieving empirical success on real,
large-scale tasks involving psychologically realistic input/output mappings is an important and
irreplaceable source of evidence for neurocomputational models, and a reasonable goal of
artificial intelligence research (see Hinton 1981, p.163). This sort of evidence can be gathered
only by building models simple enough to work given current technology, and it seems to me
that at least the best work in the connectionist tradition aims to do just this, while keeping
idealizations as principled as possible with respect to psychological and neural reality.
6.8 The representational power of graphical models

As I’ve mentioned, a running theme in the literature on neo-Classical approaches to cognition is that graphical models like Bayesian networks, even when they are hierarchically structured (thus allowing for at least modest forms of compositionality), are fundamentally too limited in their representational power to explain the flexibility of human cognition (Goodman et al 2015). This criticism has recently been directed at theories that aim to explain cognition in terms of hierarchical Bayesian inference within graphical models (see Williams 2018). In this section I’ll address these worries, and argue that while realistic applications of graph-based representations do run up against limitations due to their finitude, these limitations might be a good fit with those of the human brain.

Though worries of the kind in question are often directed specifically at Bayesian networks in the technical sense defined in (Pearl 1988), this is somewhat incidental. These arguments do not depend on the features that distinguish Bayes nets proper from similar graphical models. And while the directed, acyclic structure of Bayes nets makes inference within them particularly simple, most connectionist systems that are interesting as models of mind (what I call the “Helmholtzian architectures”, roughly those that combine the multimodal encoder-decoder framework discussed in Chapter 5 with unsupervised learning) cannot be understood, at least in their entirety, as Bayes nets. Any model that takes top-down feedback to play a role during perceptual inference, for example, perforce introduces cycles.\footnote{Moreover, top-down feedback connections are ubiquitous in the brain (Friston 2005), so while the Bayesian networks formalism is useful for analyzing idealized systems, it is likely only part of the story.}

So, I will focus in what follows on graphical models more generally. One version of the criticism on the table is that, since a graphical model can represent only propositions and relations among them (captured in its nodes and edges, respectively), rather than the relations
among objects and their properties captured in first-order logic, they lack a rich form of compositionality, and so cannot explain cognition (Williams 2018).

The considerations in §5.1 above have already gone some distance toward answering this worry. One way in which a cognitive system might represent objects, properties and relations is by way of representing the rich variety of states of affairs in which they participate, which it may do in terms of the relations that hold among those states of affairs. There is no a priori requirement that in order to represent objects and properties, a representational system must possess separable syntactic constituents that have them as their semantic values. Indeed, as previously argued, it’s not clear that the idea of just representing a dog (as opposed to a dog’s being somewhere, doing something, etc) is intelligible. The traditional conception of concepts as distinct from propositional representations may be partially reconstructed by viewing them as abstractions over the inferential processes in which they figure\(^\text{192}\) (where the story in §5.2 about word-level concepts as operators on an overall vector space representation is a way of cashing out some of the relevant inferential roles).

I take it that a compositional representation based on proposition-valued nodes would indeed require more primitives to achieve the same productive capacity as a stock of singular terms, predicates, quantifiers and the like. However, it’s important to note that in finite domains at least, graphical models are at no absolute representational disadvantage: graphs can be constructed online to represent any given (finite) evidential situation (Breese 1992)\(^\text{193}\), and more particularly, as argued in (Bacchus 1993), a graph can be constructed that captures the relevant

---

\(^{192}\) This idea is due to David Rosenthal—see slide 34 of “Content and Inference”, a talk given at the Feb. 2018 CUNY-Milan Workshop on Belief.

\(^{193}\) This seems in effect to be exactly the kind of thing proposed in (Kemp and Tenenbaum 2008). But insofar as I find that proposal unmotivated, the concern is with an explicit innate “graph grammar”, not with the more general idea of cognitive systems embodying some procedure for generating graphs to order, which is in effect what the interpretation of connectionist nets in (Hinton 1981), discussed below, amounts to. In any case my point at present is only that graphical models are not fundamentally limited in their expressiveness due to lack of syntactic structure in the nodes.
inferential structure in any collection of first-order logic formulae. The resulting network may in some cases be unwieldy and perhaps highly redundant in its representation, but the issue immediately at hand is that graphical models are not barred from representing some particular class of entity on grounds that they lack a certain type of compositional structure.

Even so, it might still be supposed that graphical models-based approaches to representation have a problem with abstraction. Goodman et al (2015), for example, suggest that the expressive power of Bayesian networks is too limited to model human cognition, and the ease and flexibility with which human beings acquire new concepts and transfer knowledge to new domains. Despite the ability of Bayesian networks to model probabilistic reasoning directly over arbitrary variables, an unadorned Bayesian network seems unable to account for the way in which abstract knowledge functions: abstract reasoning must immediately have implications for indefinitely many concrete cases, which would have to be represented in a graphical model by their own nodes (Breese 1992, §2). There are really two concerns here: first, it is unclear how abstract reasoning with implications for many particular cases should or can be represented in graphical models, and second, it might be argued that true abstraction requires generalizability to potentially infinitely many cases, as is captured, for example, by universal quantification over an infinite domain in predicate logic.

The first worry might be addressed by appeal to an idea in an early discussion of Hinton’s on distributed representations (Hinton 1981). As he notes there, “it is important to realise that the behavior of any network of hardware units can be described either at the level of activities in individual units or at a higher level where particular distributed patterns of activity

---

194 Interestingly, the work in that paper appeals to a knowledge base couched in a probabilistic extension of first-order logic. This does not affect the issue under discussion, since this logic is more expressive than ordinary first-order logic. Importantly, however, the author conceives of probabilities in terms of proportions of assignments that lead to truth within a finite domain, and the semantics of the logic is accordingly limited.
are given particular names” (Hinton 1981, p.162). Graph structure can be read into the network not just at the level of the units and hardware connections, as is usual (cf. §4.5 above), but at a more abstract level, where nodes in the graph correspond to distributed representations and edges represent functional relations among these representations (see Fig. 6.2 in Hinton 1981).

On this interpretation, a layer in a standard artificial neural network model, or a capsule in a capsules architecture (each corresponding to a vector space capable of supporting many distributed representations), would, depending on its state, correspond to many different nodes in a graphical model (such as a typical Bayesian network) representing relations between particular propositions. The graph determined by these states is in a sense “virtual”, and may be much larger and more complex than the neural network itself, since the number of distributed representations supported by a group of hardware units (hence the number of nodes in the virtual network) increases exponentially with the number of units. For example, the “virtual” graph corresponding to a simple feedforward network that maps 64 x 64 pixel images with 8-bit color depth to a distribution over 10 binary class labels would have over a million nodes and over 10 million edges if each possible input image is represented by its own node.

The mapping between the virtual graph thus constructed and the ground-level states of a network is, for standard layer-based architectures, quite transparent. In most cases, the edges in the virtual graph correspond to direct causal connections between the representations, which supervene on the individual synaptic weights that connect their constitutive nodes. In networks with recurrent connections (construed broadly so as to include either direct recurrent connections from \( L \) to \( L \) as in RNNs, or cycles in the graph structure more generally), the relations between different distributed representations \( R_1 \) and \( R_2 \) in the same vector space (i.e.

---

\(^{195}\) A somewhat similar kind of abstraction is prevalent in analyses of recurrent networks, where the causally related states of a single layer over time are represented as “unrolled” in space (see e.g. Figure 6 in §4.1 above).
in the same layer of a network) are simply a special case: $R_1$ and $R_2$ will be represented as different nodes in the virtual graph, linked directly or indirectly. The exception is that alternative distributed representations in the same vector space at a given moment in time are mutually exclusive, a fact that ought to be captured in a graph that keeps track of the dependencies among the distributed representations even though in this case the dependency is not causally mediated in the usual way. This could be encoded via edges between pairs of nodes corresponding to each possible pair of distributed representations $R_i$ and $R_j$ in $L$, and associated probabilities such that $p(R_j|R_i) = 0$, within a given timestep.\textsuperscript{196}

This scheme in principle answers the worry about abstraction posed above: the same weight matrix connecting two distributed representations may encode many distinct but related relations among distinct but related entities, propositions, etc (cf. the interpretation of the transformation matrices in capsules networks as embodying abstract knowledge of part-whole relationships for entity classes, in Hinton et al 2011). The connectionist network then acts, at a somewhat abstract level of description, as a generator for models of particular cases, and new knowledge that modifies this generator may accordingly have consequences for indefinitely many instances. Of course, the type of structure in the vector space representations that supports this form of abstraction—that similar distributed representations have similar causal profiles as well as similar semantic interpretations (cf. §1.3, §4.1, §5.6)—will be only indirectly captured in the structure of the virtual graphical model.

Given the Bayesian interpretation of interactions between individual nodes in models such as that of (Hinton and Sejnowski 1983), it would be nice if some compact analysis could be given of the relation between Bayesian inference at that level and at the level of the kind of “virtual” higher-level graphical model just discussed. This should be possible, since, where $v$ is

\textsuperscript{196} This would of course rule out the interpretation of the resulting virtual graph as a Bayesian network in the strict sense of (Pearl 1988).
an $n$-dimensional vector and $v_1, v_2, \ldots, v_n$ are its components, $p(v) = p(v_1, v_2, \ldots, v_n)$. Thus, inferences between distributed representations might be thought of as equivalent to inferences between sets of premises encoded in the node-level representations. In some models, such as the Helmholtz machine (Hinton et al 1995), the distribution over hidden-layer states is factorial, lending itself to an interpretation in terms of the states of each node in layer $L+1$ being inferred independently from the “premises” in layer $L$. However, the limitation to factorial distributions, while it improves tractability, is in many ways undesirable and may not be a good feature on which to base a general analysis. I leave development of these ideas for future work.

What about the issue of finitude? I do not find this particularly pressing, given that the ways in which universal quantification figures in human reasoning do not hinge on our being able actually to represent an infinite class of entities “all at once”, as it were. A system that can harbor a representation whose content is equivalent to that of $(\forall x)(Fx \supset Gx)$ must be capable of subsuming an open-ended set of particular cases under this representation, which means in practice only that arbitrarily large finite sets of $Fs$ must be classified as $Gs$. Moreover, though this is not a feature of most connectionist models, it’s easy to see how the basic connectionist framework can be extended to treat the number of nodes in a network (or layer) as a free parameter, perhaps one that is optimized along with others during learning. Indeed, proposals of this sort exist—see for example (Yoon et al in review). So there is no reason to be concerned that connectionist nets can only represent a fixed set of entities.\(^\text{197}\)

All this said, one may still worry about combinatorial explosion and capacity constraints, particularly in the case of VSMs in which the top layer of a network is meant to correspond to a single “root node” of a parse tree that can potentially represent any word, sentence or phrase

\(^{197}\) One might also appeal to the infinite density of continuous vector spaces (Pollack 1988) for a more radical reply to this worry. On standard assumptions about the nature of neural computation, it is hard to see how such infinite capacity could be exploited (see below). Those assumptions may be wrong, but that question is beyond the scope of the present study.
(similar remarks apply to the composition matrices that map to this space). We’ve seen (most dramatically in §5.4) that forcing the weights to model a wider range of data actually promotes systematicity, provided the model is big enough. Intuition may suggest, however, that the vector space representations would quickly become overloaded (Baroni 2013, p.518-519), and intuition may be right. As (Fodor and Pylyshyn 1988) note, in discussing a similar issue: “George Miller once estimated that the number of well-formed 20-word sentences of English is of the order of magnitude of the number of seconds in the history of the universe” (p.24).

It’s difficult to assess these intuitions qualitatively. One could of course do the math, though I won’t attempt to here. While the number of sentences that need to be represented (even leaving aside the assumption of arbitrary depth of recursion) is huge, so is the capacity of a large model, and it’s likely that the space would be tailored to the most frequent expressions, with on-the-fly learning (akin to perceptual learning, or adaptation) “making room” in the vector space if some particularly exotic sentence needs to be represented. Putting this point slightly differently, there is no reason that the mapping from propositions to network states must remain fixed, since we needn’t be able to represent everything all at once (one may use what Hinton (1990) calls “time-sharing” of the hardware).

There is a deeper perspective on these issues that I will not be able to explore adequately at present: if we think of words (as well as larger phrases) as operators on a semantic space, we ought really to abandon the idea that what vector space models do is store data points (e.g. the meanings of all possible sentences) at various locations in the space.198 Gibson (unpublished),199 in his remarks on “ordinal stimulation”, claimed, in effect, that we only

198 Indeed, my rejection of VSAs in §2.9 was based in part on the idea that this is not an optimal way of using vector space representations.
199 See “Purple Perils”, 1954-1961, “Ordinal Stimulation and the Possibility of a Global Psychophysics”: “To speak accurately, then, it is not the energy as such which constitutes stimulation for the skin or the retina but typically the changes of energy. The retinal image, according to this conception does not consist of points or spots of light but of transitions.”
perceive differences, i.e. that the smallest units of sensory stimulation that make a psychological impact are ratios between energies, receptor states, etc. Perhaps the same is true for meaning: the content of an expression might be tracked by the alteration it makes to an existing body of belief. Relatedly, according to Donald MacKay, the meaning of a signal, for its receiver, is the “selective function” it performs when it arrives: the modification that the signal makes to the receiver’s “states of conditional readiness”, i.e. dispositions to action (MacKay 1969, p.24-25).

Strictly speaking, in any case, infinitely many vectors can be stored in a continuous vector space (see, again, Pollack 1988), but given finite resources for retrieval, there will be practical limits on such systems well before infinity. Eliasmith (2000a), for example, discusses upper limits on the encoding and retrieval of information in the brain, that point to effectively discrete rather than continuous encodings. And error-correcting mechanisms typically require redundancies that reduce capacity. However, I don’t think there is reason to be skeptical, on the basis of neural capacity considerations alone, about at least most of the particular connectionist models discussed in these pages. As Hinton notes, referencing the network depicted in §5.5, Fig. 9, which recognizes handwritten digits: “The network...has about as many parameters as 0.002 cubic millimeters of mouse cortex. Several hundred networks of this complexity would fit within a single voxel of a high resolution fMRI scan” (Hinton 2005, p.10).

6.9 Serial VS parallel architectures and implementation of the LoT

We’ve seen the issue of Classical constituency that divides connectionist and Classicist systems not to be all that pivotal from a functional perspective. The main remaining difference between these architectures is perhaps not the presence or absence of symbols (in the sense of representations with arbitrary content), but precisely (as should not be surprising) the parallel
nature of connectionist systems, in which processing is locally governed and responsive to a
graded continuum of similarity among representations, rather than relying on a fixed set of
on-off rules applied indifferently to whatever structure is currently being interpreted by a single
processor (as in the standard von Neumann serial computer architecture).

Indeed, this *localism* is the proper source of most of the challenges inherent in
connectionist modeling, and associated objections, rather than the parallelism *per se*. Why
shouldn’t we just build some parallelism into a serial processor, then, as Fodor and Pylyshyn
suggest is perfectly intelligible (Fodor and Pylyshyn 1988 pp.55-56), and have the best of both
worlds? One answer is that in practice, connectionism achieves a workable parallelism only by
employing a legion of *simple* processing units, and the local nature of computation is an all but
necessary result of using such simple processors to perform complex operations: the structure
lacking in the processors must be compensated for in the structures relating them.

We can view the two extremes of logical, rule-based symbol processors and
connectionist nets as poles on a potential continuum from more complexity in the internal
processors and less parallelism to more parallelism and simpler processors.\(^{200}\) Capsules
networks (cf. §4.6) would fall somewhere between these poles. By contrast, a PLoT, as such, is
(if current research is any guide) either an essentially serial system with probabilistic elements, if
cashed out in terms of a probabilistic programming language (Goodman et al 2015, Goodman et
al 2008), or a hybrid system that employs hierarchical structured priors, most likely over
lower-level representations that are themselves well modeled by connectionist nets (see again
Salakhutdinov and Tenenbaum 2011 and Tenenbaum et al 2011).

\(^{200}\) More properly, I suppose, the continuum is a manifold defined by two variables: amount of parallelism
and processor complexity. But I assume that much of this space would be uninhabitable by realistic
models. Connectionist-like systems with very little parallelism are not of much use, and a system with as
many parallel processors as a typical deep learning model, in which each processor had the
representational power of first-order logic, would be prohibitively computationally expensive.
Does the idea floated in the previous section, that the graphical models in terms of which probabilistic inference should be understood are really “virtual” graphs derivable from distributed representations in connectionist networks, amount in any way to a concession to the view that connectionist networks themselves can model cognition only by implementing Classical processing? I think there are many reasons to answer this question in the negative.

First, and most importantly, neither level of description of a connectionist net looks like a Classical system. The description in terms of nodes and synaptic weights obviously doesn’t, but even if the higher-level graphical model includes tree structures, as considered in Chapter 4 above, a graphical model of this sort is not a serial symbol processor. Such graphical models capture the network of relations among states of affairs that the system represents, and these will (in a successful model) capture relevant inferential relations. But whatever other constraints are obeyed by the relations between representations in such a model, they must also be understood as probabilistic (fundamentally associative) relations.

There is, perhaps, room to argue that at least some forms of inference really do involve running a “virtual machine” on parallel hardware—for example this may be the case if a mental model of the kind described by Johnson-Laird (2010a, b) is used for at least paradigm (deliberate, conscious) cases of deductive reasoning (cf. the discussion in §6.4 above). This would amount to imaginatively constructing a model of a situation, akin to the interpretations appealed to in the semantics for first-order logic, and inspecting it to see what is true in the imagined situation. Such an explanation could perhaps be applied even to unconscious cases of deductive reasoning, should there be any.

But this would not amount to the automatic application of rules sensitive to the syntactic structure of input symbols in any case. It would involve using the connectionist hardware to represent a situation (just as it would be used to represent any situation, e.g. in perception), and
then deriving the conclusion from an inspection of this representation. This skips the formal
system and goes straight to the interpretation (in the sense familiar from model-theoretic
semantics). The conclusions that could be drawn in this way should of course comport with
those licensed by logic (idealizing away from performance errors), but they may perhaps
succeed in doing so because, at the intuitive, analog, quasi-perceptual level of representation in
question, at any rate, we can only model consistent situations.\textsuperscript{201}

Finally, it may be argued that if a full explanation of the way connectionist systems
support cognition can only be given by appeal to a level of analysis at one remove from a direct
hardware description, i.e. in terms of a “virtual” graphical model, or a process via which graphs
for particular inferences are “generated” from the underlying network mechanics, connectionist
networks are after all only substrates for the genuinely “cognitive” level of representation, just as
Fodor and Pylyshyn suggested, even if the cognitive-level representations are not Classical in
nature. But this argument would be mistaken, since the nodes in the “virtual” graph under
discussion correspond to the distributed representations in connectionist networks. The virtual
graph is just a way of keeping track of the relations among the network’s representations.

\textbf{Conclusion}

Srivastava and Salakhutdinov (2014) point out that in addition to the various specific
psychological functions that it may help to model, learning cross-modal associations might be
essential to the project of artificial intelligence (and, we may suppose, to modeling human

\textsuperscript{201} One could of course argue that people can think ‘\(p \& \neg \neg p\)’, and in that sense model it. I would argue
that we accept the explicitly articulated logical principles (and develop the logical calculi) that we do,
however, only because at a more fundamental level we cannot simultaneously represent the premises
and the negation of the conclusion of a valid argument. This is perhaps a part at least of why most
people intuitively accept the law of non-contradiction, despite the apparent weakness of explicit
arguments in its favor (see e.g. the discussion of Aristotle’s arguments in Priest 2006).

243
intelligence, insofar as this is a different project) for more general reasons: “Unless we do multimodal learning, it would not be possible to discover a lot of useful information about the world” (p.2950). For example, the captions of photographs and other text associated with images often provides information lacking in the images themselves.

Presumably, this argument runs in the other direction as well: though I’ve been focusing here on the distributional approach to semantics construed narrowly in terms of language-internal distributional facts, a fuller distributional approach would also take into account correlations among linguistic expressions and extralinguistic items like sensory states. As we’ve seen, doing so is at least within the realm of conceivability when all representations are construed as vectors. Thus, Yann LeCun half-jokes that the correct approach to AI is to simply convert everything into a vector and let the vectors predict one another: “world2vec”.

This is, in some ways, an austere theory of mind, as is connectionism generally in many ways. And yet, it seems to work: it supplies the minimal machinery needed to get the job done in a very large number of cases. My goal has been to push connectionism as a philosophical theory, that is, to attempt to take its talk of “representations” seriously as having psychological import, particularly in those areas of mental function associated with “cognition” proper (as opposed to, e.g. perception and sensation), where the applicability of connectionist models has most persistently been questioned. That said, I regard “connectionism”, of course, as a label for an evolving research program, not as a complete theory that has all the answers as it stands. The discussion of recent developments such as capsules networks, I hope, makes this clear. And, while deep learning has certainly made incursions into a wide variety of subfields in AI, the models considered here do not offer explicit accounts of the nature of certain fundamental

---

dimensions of the human psyche, such as consciousness, affect, and volition. Connectionist modeling may be expected to bear at least indirectly on our understanding of these phenomena in the long run, but much of that story remains to be told.

I’ve often heard it argued that the recent achievements of deep learning don’t shed new light on the ways in which neurons give rise to psychological function: they are manifestations of old, already well-digested ideas whose time has finally come, thanks to the availability of faster, cheaper computers. Indeed, some leading practitioners of machine learning have helped popularize this view. I think the wealth of striking new ideas in the models surveyed above belies this assessment, however. For one, many theoretical advances were made in the 1990s, particularly relating to unsupervised learning and generative models, whose large-scale application is only now beginning to be explored. And even if extrinsic technological changes rather than core theoretical advances were the main catalysts of recent events, the process thus set in motion has lead to genuine novelty, often in directions relevant to psychological theory.

To illustrate this, I’ll close with a brief list of some of the core commitments that I take to be shared by many of the particular connectionist approaches I’ve discussed in this dissertation, beginning with more fundamental philosophical claims and shading into more concrete hypotheses about brain function and particular aspects of neural information processing. These features include, but are not limited to, those that I have taken to be definitive of “Helmholtzian” architectures (see §6.8). To the extent that these common commitments remain controversial within cognitive science and cognitive and computational neuroscience in particular, it can be argued that connectionism constitutes a distinct research program of some substance, though one that is to be sure continuous with other traditions, and has gone by other names (e.g. cybernetics, dynamical systems theory, etc).^{203}

^{203} Cf. (Goodfellow et al 2016), §1.2.1.
(1) For many psychologically relevant functions, from low-level sensory processing up to core cognitive capacities, there is no need to posit a complicated, as-yet-unintelligible mapping between the “hardware” (or “wetware”) and “software” levels: while precise neural implementations in most cases remain up for grabs, only slightly abstract and idealized descriptions of neural representations are suitable for the purposes of psychological theory (cf. Eliasmith 2000b).

(2) The mind/brain represents things in the way that statistical models do (cf. Hohwy 2013, Chapter 2), by way of large networks of empirically associated variables, including inferred latent variables, and parameters, encoded in enduring structures, such as long-term potentiation in synapses.

(3) There is a single domain-general learning principle at work locally more or less everywhere in cortex (cf. Hinton 2005), admitting perhaps of small variations for different neuron types.

(4) The core representational language of the mind/brain is the vector space, and the core process defined over such representations is mapping from one such space to another (or to the same space), which constitutes transformation of the representations (cf. Churchland 1986).

(5) There is a broadly Bayesian mechanism at work throughout the cortex, via which top-down predictions are matched against incoming signals, allowing the brain in
principle to bootstrap to a model of the world on the basis of empirical learning alone, given a large enough set of neurons and the right input signals (with the extent of genetically determined priors an open empirical question) (cf. Clark 2013, Hohwy 2013, Friston 2009).

(6) The input and output channels of the brain correspond to (typically hierarchically organized) encoders and decoders that map sensory inputs to a common vector embedding space (or set of such spaces), and back onto the states of motor output channels (see for example §§5.4—5.6 above and references).

(7) One of the core functions of the series of processing steps that re-represent sensory inputs within sensory processing hierarchies is to map the raw inputs to lower-dimensional spaces where representations are more directly (e.g. linearly) related to one another (see e.g. Hinton and Salakhutdinov 2006, Bojanowski et al 2018), constituting a compressed representation of the inputs and potential outputs.

Claims (5) and (6) are closely related to recent large-scale theoretical frameworks in cognitive science such as hierarchical predictive coding (Clark 2013), the prediction error minimization framework (Hohwy 2013), and the free-energy principle (Friston 2009), and have accordingly received widespread attention in contemporary philosophy and psychology. The other claims have been less directly discussed, but (1)-(4), and perhaps to a lesser extent (7), constitute important loci of dispute between pure connectionist approaches and those proposals associated with the more Classically oriented PLoT hypothesis (e.g. Goodman et al 2015, Lake
et al 2015). It is my view that (1)-(7) function most effectively as a package deal, though this list is no doubt incomplete.

My defense of connectionism in these pages has on occasion exhibited a polemical flavor that parallels Fodor and Pylyshyn’s polemics in favor of Classicism, which may seem unnecessary given the de facto dominance of connectionist approaches in contemporary AI. Lest it be thought that I am stirring up controversy where there need be none, however, consider this quote on outstanding challenges for the Hierarchical Bayesian Models approach in (Tenenbaum et al 2011, pp.1284-1285):

"The biggest remaining obstacle is to understand how structured symbolic knowledge can be represented in neural circuits. Connectionist models sidestep these challenges by denying that brains actually encode such rich knowledge, but this runs counter to the strong consensus in cognitive science and artificial intelligence that symbols and structures are essential for thought. Uncovering their neural basis is arguably the greatest computational challenge in cognitive neuroscience more generally—our modern mind-body problem."

Consensus aside, it’s important to keep in view what is really at stake here. For one, we can distinguish between the content or quality of a piece of knowledge and the way in which it’s encoded. “Structured symbolic knowledge” is presumably just knowledge encoded in symbolic structures. Connectionists characteristically deny that brains encode knowledge in specifically language-like symbolic structures (though as we’ve seen, they needn’t deny that), but it is becoming increasingly apparent that connectionist nets can usefully encode rich semantic (and syntactic, where applicable) knowledge in the structure of vector space representations.204

204 It’s been proven, of course, that neural networks as modeled in connectionism, even very structurally simple ones, are universal function approximators (Cybenko 1989, Hornik et al 1989, Siegelmann and Sontag 1991, Nielsen 2015). The question, I take it, is whether this capability can be deployed effectively in real time in the specific sorts of biological neural networks that underlie embodied human intelligence.
References


Jurafsky, D. and Martin, J.H. (2009). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition (2nd


