A COMPUTATIONAL FRAMEWORK FOR LEARNING AND TRANSFORMING TASK REPRESENTATIONS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF PSYCHOLOGY
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Andrew Kyle Lampinen
June 2020

Abstract

Human cognition is fundamentally flexible — we can adapt to novel tasks rapidly. We can sometimes adapt to a novel task without any direct experience on that task, based on its relationship to previous tasks. By contrast, while deep-learning models can achieve superhuman performance on many tasks, they are often unable to adapt to even slight task alterations. This ostensible inflexibility has led to criticism of deep learning models by cognitive scientists. I begin this dissertation by reviewing the literature on cognitive flexibility, and recent advances in building more flexible artificial intelligence systems. I provide a synthesis of these literatures, and outline the challenges that I believe remain. In particular, I focus on the ability to adapt to new tasks zero-shot — that is, without any data — based on their relationship to prior tasks.

To address this challenge, I propose a general computational framework for adaptation to novel tasks based on their relationship to prior tasks. The framework is based on *meta-mappings*, higher-order tasks that transform basic tasks. I propose a parsimonious implementation of this framework in the form of *homoiconic meta-mapping* architectures. I demonstrate this framework across a wide variety of tasks and computational paradigms, ranging from regression to image classification and reinforcement learning. I compare to both human adaptability, and language-based approaches to zero-shot task performance. I show that meta-mapping is quite succesful, often achieveing 80-90% performance on a novel task, even when the new task directly contradicts prior experience. I further show that using this adaptation as a starting point can dramatically accelerate later learning on a task, and reduce the errors made on the way to mastery by nearly an order of magnitude.

Thus, I suggest that meta-mapping can provide a computational basis for adapting to new tasks, and a starting point for efficient learning. This dissertation therefore provides a framework for building better cognitive models and more flexible artificial intelligence systems. In the final chapter, I review the broader contributions of this work to an ongoing discussion about the computational principles necessary for intelligence, and highlight possible future directions ranging from understanding mathematical cognition to neuroscience.

Acknowledgements

I have been very fortunate both while writing this dissertation, and along the longer road to the PhD. I am grateful to many people who played many roles in this journey.

First, I'd like to thank the other graduate students and researchers in my lab, who have helped shape my thinking and clarify my ideas, and collaborated on several fascinating projects. That includes Kevin Mickey, Steven Hansen, Arianna Yuan, Katherine Hermann, Andrew Nam, and Effie Li, as well as Irán Román, Amarjot Singh, and Rav Suri. I'd also like to thank the students in the cognitive area more broadly, especially Erin Bennett, MH Tessler and Robert Hawkins, who have really shaped my thinking as a cognitive scientist.

I have been greatly inspired by conversations with many researchers outside my area as well. In particular, Dan Birman, Tyler Bonnen, Akshay Jagadeesh, Dawn Finzi, Nathan Kong, Ari Beller, Corey Fernandez, Ian Eisenberg, Em Reit, and Dan Bear. My apologies to the many important people I have likely forgotten to mention. I would also like to extend a special thanks to Fred Bertsch and Doug Eck, with whom I interned at Magenta (Google Brain) in 2017, and Adam Santoro and Felix Hill, with whom I interned at DeepMind in 2019. The latter have been particularly influential on my thinking about some of the issues, such as compositionality and systematicity, that I discuss in this dissertation.

I've been inspired by many conversations and questions from the incredible Stanford faculty. I'd particularly like to thank my reading committee members Surya Ganguli and Noah Goodman. Surya led the way on one of my most fascinating projects during the PhD, exploring how learning dynamics could explain generalization, and offered crucial comments at many points along the way. Noah has asked me good questions and offered useful suggestions since before I was admitted, and many of the ideas in this dissertation were inspired by those conversations.

Most importantly, Jay has been the most wonderful advisor that I could imagine. You have always known the right question to ask and the right experiment to run. I have learned so much from you about designing models, simplifying problems, designing experiments for humans and machines, asking good questions, and teaching and writing clearly. I am continually inspired by the breadth of your expertise and knowledge. Thank you for everything.

I'd also like to thank my mentors and inspirations from my past academic lives, especially Yury

Kolomensky & Bernd Sturmfels from my time at Berkeley; and Javier Armendariz & Amanda Galante from my time at Johns Hopkins; as well as all the wonderful people I learned from at both institutions.

On a less academic note, I would like to express my gratitude to my family. My father provided me with very early inspiration about science and research and asking good questions. My mother was a programmer as well as homeschooling my sister and I, and taught me a great deal about computers and everything else. My sister Linnea has always been an inspiration, and I'm very proud that she is completing her bachelor's degree in psychology and biology. Finally, my grandfather Hugh, who taught me about physics and math from a very young age, and helped put me on the trajectory that led me here.

The Stanford climbing community was one of the best parts of my time at Stanford. In particular, I am grateful to Dan Birman for teaching me so much about climbing; to Robert Hawkins, Allison Ong, Mona Rosenke, Corey Fernandez, Andrei Kamalov, Adrienne Mueller, Erin Bennett, Em Reit, and Daniel O'Leary for many adventures in Yosemite, Pinnacles, Castle Rock, and beyond; and to Andrew de Torres and Sonia Rackelman for allowing me to work as a routesetter at the Stanford Climbing Wall.

I'd also like to thank my friends from outside of Stanford, in particular Kyle Cameron, Hurshal Patel, Ben Sklaroff, Andrew Gearhart, Anita Satish, and Aditi Narayanan, for bringing me some much-needed outside perspective during my immersion here.

The other students in my cohort, who have journeyed through this program with me, have been an invaluable source of support and laughter along the way. Jesse Reynolds, Lester Tong, and Pam Wang have been great friends. Special thanks Erin Bennett, Mona Rosenke, and Yochai Shavit. Erin, you have been a great friend, climbing partner, and provided amazing feedback on ideas, papers, and talks. Mona, you have been an amazing friend, climbing and adventure partner. Yochai, you have been a wonderful friend and discussion partner for research, statistics, or just philosophy, sometimes to the annoyance of the rest of the cohort. Thank you all.

Finally, I'd like to thank Julie, who I'm so grateful to have met toward the end of my time at Stanford. You bring so much joy to my life; our adventures and laughter have kept me sane through a pandemic and writing this dissertation. Thank you.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

"The great evolutionary advantage of the human species is adaptability."

Dedre Gentner, *Why We're So Smart*

"The most elementary single difference between the human mind and that of brutes lies in this deficiency on the brute's part to associate ideas by similarity."

William James, *Principles of Psychology*

Deep learning models have achieved incredible success recently, reaching human-level (or super-human) performance in domains ranging from vision (Szegedy et al., 2015) to board games (Silver et al., 2016) and video games (Vinyals et al., 2019). However, these models still lack some important human abilities (e.g. Lake et al., 2017). These models are often data-hungry, while humans can frequently learn from relatively few examples. Furthermore, even if deep-learning models are given a large quantity of training data, they may not be able to generalize well outside the data distribution they experience during training. Finally, once knowledge is learned in the weights of a deep learning model, it is difficult to flexibly reuse that knowledge. From the perspectives of Gentner & James (quoted above), most deep learning models are unlike humans. Deep learning models don't understand the relationship between new situations and their prior experiences, and so do not know how to flexibly adapt to changes in their environment or goals.

By contrast, humans can reuse knowledge flexibly. We can learn from few examples. We can introspect about our learned behavior to explain or change it. We are often able to accurately behave according to linguistic instructions, without requiring examples of the desired behavior. We can even remap our behavior in a way completely inconsistent with our prior behavior, such as trying to lose a game we have previously been trying to win (Lake et al., 2017). All of these types of flexibility can be quite difficult for deep learning systems.

This apparent contrast between humans and neural networks has caused researchers to question the validity of neural networks as a cognitive model for many years (e.g. Fodor and Pylyshyn, 1988). The recent successes of deep learning have only increased the frequency of these critiques (e.g. Lake et al., 2015, 2017; Lake and Baroni, 2018; Marcus, 2018). It is important to address these perspectives, and to understand how deep learning systems can be improved to serve as better models of human flexibility. There have been a number of attempts to defend from, or integrate, these critiques, from a variety of perspectives (e.g. McClelland and Plaut, 1999; McClelland et al., 2010; Hill et al., 2020). For example, in past work my colleagues and I have argued that these critiques overlook the benefits of transfer (Lampinen et al., 2017) and recent progress in meta-learning (Hansen et al., 2017). Recent deep learning research has made important progress on improving learning speed and generalization. I will review some of that progress below.

Despite this progress, however, some challenges remain. In this dissertation I will focus on the ability of humans to adapt to a new task *zero-shot*, that is, without any data at all. For example, imagine trying to lose a game that you have previously been trying to win. Humans can perform reasonably on their first try at losing, despite having no data on the losing version of the task, and despite their previous goals being directly in opposition to their new one. I will suggest that we can do so by using our knowledge of what it means to lose more generally transform our representation of the present game. I will therefore provide a general computational framework for adaptating by transforming task representations, and will evaluate it across a wide range of task domains. This framework endows deep-learning models with flexibility more like that of humans.

To situate my work within the broader field, in this chapter I will review some of the cognitive issues and the progress to date. I will try to provide a unifying perspective on how various types of transfer contribute to human learning and flexibility, by reviewing my prior work as well as that of others. In the remainder of this dissertation, I will tackle the problem of building deep learning models that can adapt to task alterations zero-shot.

1.1 Cognitive flexibility & generalization

What kind of flexibility do humans have? We are often able to learn rapidly. For example, we can achieve some competence in a novel video game within a few minutes (Lake et al., 2017). We can learn new concepts from seeing only a relatively small number of examples (e.g. Bourne, 1970). We can often learn even faster if we can actively participate in the learning process by selecting examples rather than passively receiving them, especially if the concepts are simple enough that we can generate a good hypothesis space (Markant and Gureckis, 2014a).

We can then apply what we have learned to new situations. The studies of Bourne (1970) show that, not only can people generalize to new examples of a concept they have learned, they can also transfer that knowedge to learn structurally-similar new concepts more rapidly. Indeed, even without explicit awareness of the relationship between two tasks, humans can sometimes benefit from transfer effects (e.g. Day and Goldstone, 2011). In general, human analogical transfer abilities have been suggested to be a critical component of “ what makes us smart” (Gentner, 2003).

Yet human flexibility is apparent beyond transfer between isomorphic tasks. We can often competently change our learned behavior in response to instruction or other goals, such as trying to lose a game we were previously trying to win, or trying to achieve some orthogonal task (Lake et al., 2017). Indeed, it has been known for almost a century that even other animals exhibit flexible knowledge use — they engage in “latent learning” of environmental features that are irrelevant at present, but may be useful when solving future tasks (Blodgett, 1929). Both humans and other animals are capable of flexibly applying our knowledge in many situations.

However, our flexibility is not universal. Sometimes it is quite difficult for us to integrate new knowledge. For example, even undergraduate students with substantial mathematical background often struggle with understanding new mathematical concepts.[1] They may mistakenly assume the converse of a theorem, or get caught up in concrete ways of thinking about abstract concepts (Hazzan, 1999). Socrates' dialog about doubling the area of a square captures the misunderstandings that modern subjects make, just as it did for those over 2000 years ago, yet it does not help them to deeply understand the principle (Goldin et al., 2011). Even after engaging with the dialog, nearly 50% of modern subjects failed at the simplest generalization of the principle: to a square of different size. Similarly, even students who complete a course in geometry in high school may not achieve formal deductive understanding of the concepts taught unless (or until) they become undergraduate mathematics majors (Burger and Shaughnessy, 1986). Furthermore, superficial details of how a concept is presented can have profound impacts on how easy it is to reason about, even if the underlying concept is exactly the same (e.g. Kotovsky et al., 1985; Kaminski et al., 2008); we have shown that these details can therefore change how easy it is to learn future concepts in the domain (Lampinen and McClelland, 2018). There is a wealth of research showing that our learning is far from universally flexible.

Humans also often fail to flexibly use the knowledge we have. For example, even mathematics students who can correctly state a rule or theorem are not necessarily able to apply it to create a proof (Weber, 2001). Similarly, even if experimental subjects can learn a basic concept rapidly, it may be difficult for them to apply it in more abstract situations or to extract more formal understanding from it (e.g. Lampinen and McClelland, 2018). Likewise, rapid analogical transfer is often only possible when superficial details match closely, or when subjects are explicitly told to transfer (e.g.

[1]I will focus on mathematical cognition in some places within this dissertation. Although the phenomena I discuss are much more general, and I will include a number of other examples, mathematical cognition offers a unique microcosm of human abilities for transfer, abstraction and flexibility.

Gick and Holyoak, 1980). Because of findings like these, Detterman (1993) has argued that inducing transfer requires manipulations "with the subtlety of a baseball bat," and so we should conclude that "significant transfer is probably rare and accounts for very little human behavior." This quote is a particularly tendentious presentation of the issues, but it captures the important broader insight that humans are not always rapid learners or flexible reasoners.

How can we reconcile the demonstrations of rapid learning and flexibility with the evidence that some concepts are learned slowly and some knowledge is inflexible? How can we reconcile arguments that transfer is key to "what makes us smart" (Gentner, 2003), with arguments that it "accounts for very little human behavior" (Detterman, 1993)? There are a variety of factors that affect whether transfer will occur (Barnett and Ceci, 2002; Lampinen et al., 2017). First, we need high-quality representations of the concepts we are learning in order to reason flexibly with them. These generalizable representations are generally created through making connections between different pieces of our knowledge (Wilensky, 1991; Schwartz and Goldstone, 2015), and perhaps through re-representation processes (Karmiloff-Smith, 1992, see below). Second, we often need strategic meta-knowledge about where and how to apply our knowledge in new situations, which also must be learned (Weber, 2001). Both these factors mean that transfer may happen more easily over longer periods of time, as I have argued in my prior work (Lampinen et al., 2017). The quality of the representations we have, and the way those representations relate to the new tasks we are presented with, both affect our ability to learn rapidly, reason flexibly, and generalize.

1.1.1 Flexibility as transfer

I argue that all these types of flexibility (and inflexibility) can be seen as transfer, defined broadly as the way that "knowledge acquired in one situation applies (or fails to apply) in other situations." (Singley and Anderson, 1989). From this definition it is clear that applying learned features and structures to accelerate learning in a new situation, as in Bourne (1970), is a type of transfer. I argue that in fact all rapid learning must rely on transfer of prior knowledge in order to constrain the hypothesis space under consideration.[2][3] Even other types of flexibility, such as adapting to instructions, can be seen as transferring several different types of knowledge (prior knowledge about a task, other related taks, and language) to a new situation. For example, if we are asked to try to lose at chess, we are essentially presented a new task to which we need to transfer both our prior knowledge of chess and our prior knowledge of what "trying to lose" means. Thus flexibility and transfer are essentially two perspectives on the same broad phenomena. I will therefore use these

[2]This conclusion is a consequence of the "No Free Lunch" Theorem (Wolpert, 1996), which shows that a learning algorithm cannot be *a priori* better at generalizing than any other, including seemingly adversarial ones like choosing the model with the *worst* validation error. The advantages of a learning algorithm can only be due to the match between its inductive biases and the structure of the task(s) it must learn.

[3]While I sometimes use terms like "hypothesis space" for their useful intuitions, this does not necessarily imply that I believe that reasoning is occuring explicitly in an explicit hypothesis space — a particular set of parameters in a neural network can be seen as a particular hypothesis about an input-output mapping, and thus the parameter space of a network can be seen as a hypothesis space.

words to refer to distinct aspects of these phenomena. Specifically. I will use "flexibility" to refer to the behavioral phenomena, and "transfer" to refer to the computational principles underlying these phenomena.

Why do humans sometimes fail to adapt flexibly? There are several reasons this can occur. First, we may not have the prior knowledge necessary to be flexible. We can't transfer what we don't know. Second, our representations may lack the quality necessary to transfer them. We often can't transfer what we don't know well (c.f. Karmiloff-Smith, 1992; Hazzan, 1999; Weber, 2001). Third, we may not recognize that we have applicable prior knowledge to transfer (Detterman, 1993). Finally, our prior beliefs may be inaccurate in the present situation, and so transferring them could actually interfere with our ability to learn. This effect is generally called "negative transfer" (Singley and Anderson, 1989).

For transfer to be beneficial overall, we must generally encounter settings where our prior knowledge is applicable, and furthermore we must have good ways of integrating that prior knowledge with new experiences. In the next section I will discuss some of the features that allow us to do so.

1.2 What factors contribute to our flexibility?

We need to address the computational question of how and when we can use transfer to behave flexibly. In this section I will give a brief overview of the contributing factors.

Complementary learning systems: First, it has been proposed that we have complementary learning systems (McClelland et al., 1995; Kumaran et al., 2016). These complementary systems allow us to learn rapidly from new knowledge while avoiding catastrophic interference (McCloskey and Cohen, 1989) with the statistical knowledge we have accumulated over longer timescales. The key idea is that we have a slow (parametric) learning system which sets up good representations, while a fast (nonparametric) learning system stores new knowledge by using these representations. Throughout this dissertation, we will return to this theme of mutually-supporting fast and slow learning. I will therefore divide the rest of this section into considerations of the "slow" and "fast" systems that contribute to transfer.

However, "slow" vs. "fast" learning systems is not a strict dichotomy. This broad distinction is useful, but in reality learning systems fall on a continuum, and the time-course of learning in a given system is often dramatically affected by what knowledge is already present (McClelland, 2013; McClelland et al., 2020). Thus this dichotomy between learning systems should not be interpreted as a strict division. Instead, it is a useful way of highlighting the cooperation between distinct systems that operate across distinct, yet sometimes overlapping, timescales.

1.2.1 Slow

In this section I will discuss the slow learning systems that contribute to transfer and flexibility.

Culture & education: One critical type of transfer is cultural knowledge transmission. Our cultures have accumulated knowledge over extremely long time scales, and building upon this knowledge allows us to advance much more rapidly now (Tomasello et al., 1993; Bengio, 2012). For example, mathematical concepts have been constructed by humans (Hersh, 1997; Mac Lane, 1986), and it has taken us millenia to build up to domains like calculus. Yet today many students gain fluency with these ideas before they graduate high school. Because culture has set up useful representations for these concepts, we are able to acquire them much more rapidly (e.g. McClelland et al., 2016). Furthermore, culture has constructed systems of education that are structured precisely to help us learn rapidly and generalize effectively.

By contrast, if our culture does not represent or highlight certain concepts, we may struggle to reason about them. For example, one Amazonian tribe that lacks words for exact numbers shows substantially impaired ability to do basic tasks involving cardinality, which are interpreted as a fundamental deficit in the ability to represent exact cardinality at all (Gordon, 2004). Similarly, although the number line as a spatial representation of number appears relatively universal, cross-cultural and historical studies reveal that it is constructed rather than innate (Núñez, 2011). Culture helps set up powerful representations and metaphors for us to learn from. Without these representations, our learning would be substantially slower.

Even beyond culture, various aspects of our developmental experience may construct natural curricula (perhaps discovered by evolution). For example, the structure of visual (and visuo-motor) experience shows a natural progression as children develop, with increasing complexity and diversity at older ages (Fausey et al., 2016). These natural curricula may play a critical role in our development (Smith and Slone, 2017).

Transfer between tasks: Even if culture has not explicitly highlighted (or engineered) structural relationships between tasks, we can benefit from structural similarity. For example, after learning an artificial grammar, subjects can generalize their knowledge to novel sequences from the same grammar applied to novel symbols (e.g. Tunney and Altmann, 2001). From learning about simple harmonic oscillators in the context of springs, participants can transfer to a superficially unrelated problem about controlling the population of a city (e.g. Day and Goldstone, 2011). Because many tasks we perform share deep underlying structures, we can take advantage of transfer to learn faster on new tasks.

Grounding, embodiment, and representation quality: One particular type of transfer that seems to be especially useful is grounding (Barsalou, 2007). In particular, conceptual representations often tend to be tied into more basic perceptual-motor systems, e.g. arithmetic in the Approximate Number System (Park and Brannon, 2013), or mathematical (and other) reasoning in gestures (Goldin-Meadow et al., 1993; Goldin-Meadow, 1999). Indeed, the way we talk about many important abstract concepts, such as time, seems to fundamentally rely on metaphors linking these abstract concepts to more concrete ones, such as space (Lakoff and Johnson, 1980). This sort of grounding can be very beneficial to understanding (Nathan, 2008; Schwartz and Goldstone, 2015; Wakefield et al., 2018). Because our perceptual-motor system have exceptionally good representations that are trained over long developmental time-scales, we may benefit from leveraging these representations to transfer our understanding to analogous conceptual domains.

However, it is also worth considering that grounding and embodiment can hold us back. Even our understanding of symbolic expressions seems to be influenced by "meaningless" perceptual details like their spacing (Landy and Goldtone, 2007). It has also been argued that too concrete of examples can limit generalization (Kaminski et al., 2008), although the details of that particular demonstration have been debated (De Bock et al., 2011; Lampinen and McClelland, 2018). There is probably some negative transfer from grounding in some cases, but overall it is probably outweighed by the positive.

There are also a number of conceptual issues that come along with grounding, such as how to define abstraction (Dove, 2016), and whether the grounding must be in the real world, or can more generally be in any concepts that are better understood (Wilensky, 1991), regardless of whether those concepts are basic perceptuo-motor knowledge. The line between grounding and other types of transfer can be blurry, because, like many psychological ideas, its definitions are multifarious.

The overlapping field of embodied cognition has raised the opposite issue, arguing that cognition cannot be considered at all outside of the physical, physiological, and social situations in which it is grounded (Anderson, 2003). Others have argued that even the computation metaphor in cognition is fundamentally flawed, because it neglects the fact that intelligence evolved in systems interacting with the world (Cisek, 1999), through a process of hierarchically constructed control systems (Cisek, 2019). The fact that many of our basic concepts seemed to be formed at the optimal level for action (Rosch et al., 1976) has been cited as evidence for the importance of grounding (Lakoff and Johnson, 1999). This perspective is challenge for purely computational theories of mind (Fodor, 2001b). It is important not to neglect the world in which our brains reside in favor of a disembodied, computational mind.

Rerepresentation and different kinds of knowledge: The work of Karmiloff-Smith (e.g. Karmiloff-Smith, 1986, 1992; Clark and Karmiloff-Smith, 1993) focused on the idea that we repeatedly redescribe our internal knowledge, reorganizing it in order to better understand the world. To support this theory, she examined evidence of U-shaped developmental curves, where children would actually get worse at a task before they reached ceiling, often because they at first over-generalized a rule. She argues that these results show a pattern of systematic progression of knowledge from implicit representations to various stages of explicit representation which allow progressively more flexibility. In Karmiloff-Smith (1986) she describes some particularly interesting evidence: children fail to balance an oddly shaped block when asked to do so, but successfully balance it when asked to build a house, because their procedural knowledge is more sophisticated than their explicit knowledge.

This pattern of procedural knowledge preceding more explicit or object-like knowledge is supported by a much broader literature, from the gesture results of Goldin-Meadow et al. (1993) that I referenced above, to work suggesting that we progress from understanding mathematical concepts as processses to understanding them as objects (Dubinsky, 1991; Hazzan, 1999). If we do not already have good representations for a concept, we must create them by slow, procedural learning and reorganization, before we can begin to reason flexibly with that concept. However, it's worth noting that the process is bidirectional — procedural knowledge supports conceptual understanding, but conceptual knowledge also helps improve procedural performance (Rittle-Johnson et al., 2001). Furthermore, procedural learning can occasionally be misleading. Getting too caught up in procedures that only work in simple cases can actually inhibit conceptual understanding (McNeil and Alibali, 2005). There are sometimes trade-offs to transfer (see section 1.2.3).

Is a separate rerepresentation process necessary to explain these phenomena? It is difficult to determine. Often rapid transitions and rule-like behavior can be fully captured within neural networks or statistical learning models more generally (e.g. McClelland and Plaut, 1999; McClelland and Patterson, 2002; Schapiro and Mcclelland, 2009; Aslin and Newport, 2012). While Karmiloff-Smith (1992) argues that these mechanisms can not explain U-shaped developmental trajectories, since error-correcting learning will not apply where there are no errors, she ignores the potential effects of all the other learning that children are simultaneously doing on other related tasks. Furthermore, some compression emerges naturally from the learning dynamics of gradient descent (Tishby and Zaslavsky, 2015; Shwartz-Ziv and Tishby, 2017; Achille et al., 2017), although there has been some debate about the generality of these observations (e.g. Saxe et al., 2018). It is difficult to rule out the possibility that complex developmental trajectories children exhibit can be explained by the combination of compression and the effects of error-driven learning on other related tasks.

Summary: We accumulate knowledge throughout the course of our lives. Some of this knowledge is implicit in the statistics of the world around us, while some is culturally constructed and transmitted. The quality of our knowledge representations can be improved by connecting to other knowledge, or by grounding in perceptual-motor understanding. Once we have acquired sufficiently high-quality knowledge representations, we are often able to transfer when we encounter a new task, and thereby learn more effectively than we could from *tabula rasa*. This improvement in learning manifests as both more efficient learning and better generalization.

1.2.2 Fast

In this section I will discuss the systems that contribute to fast learning and transfer. It's important to note that these are systems that can be applied rapidly, but are not necessarily learned rapidly. Indeed, many of these reasoning systems are themselves culturally constructed, and must thus be learned over development.

Hippocampal: From the complementary learning systems perspective, the hippocampus serves as a fast learning system which can store an essentially unlimited number of distinct experiences while minimizing interference, i.e. as a nonparametric learning sytem (Kumaran et al., 2016). This feature makes the hippocampus an excellent sub-system for learning from a small amount of data, because it can store a few experiences and allow them to be retrieved at a later time. The hippocampus can also reduce catastrophic interference (McCloskey and Cohen, 1989) with knowledge gleaned from prior experiences, by allowing interleaving of these prior experiences in learning to integrate new knowledge with old (McClelland et al., 1995). This can potentially occur in various usefully-biased ways (Kumaran et al., 2016; McClelland et al., 2020). There may even be interesting computations performed within the hippocampus to support certain types of rapid generalization (Kumaran and McClelland, 2012).

Interactive learning & hypothesis testing: Humans are able to use our stored experiences to rapidly learn. For example, we often behave as though we are formulating and testing hypotheses, even from a very young age (Sobel et al., 2004; Gopnik and Sobel, 2014). We can even take advantage of these hypotheses in order to actively acquire information from the world that is most useful for us (e.g. Markant and Gureckis, 2014a). By using our prior knowledge of the world to help interpret new experiences, we are able to make extremely fast inferences about how to understand a new situation.

Education & learned flexibility: However, our ability to reason rapidly is not solely learned on our own. Indeed, an explicit focus of education is preparing us for future learning (Bransford and Schwartz, 1999), and flexibility (e.g. Richland et al., 2012). That is, we are taught to learn increasingly rapidly as education goes on. Elementary school children require months or years of rote practice to learn arithmetic, but college mathematics students are expected to hear a theorem once and then immediately apply it. As noted above, we may not be perfect at these faster learning tasks (e.g. Hazzan, 1999). However, adults are much better at them than children, and mathematics graduate students are much better than undergraduates (Weber, 2001). The ability to learn and apply new knowledge rapidly develops as we do. Furthermore, as we grow up we also grow better at explaining our actions, and adapting to instructions (e.g. Doebel and Zelazo, 2015). Over the course of development, we practice many types of flexibility.

Explanations and demonstrations: Both in education and outside of it, explanations are a key way we learn about the world, because they can succinctly convey rich structure (e.g. Keil, 2006; Lombrozo, 2006). Importantly, explanations can be succinct because they exploit our prior knowledge in order to convey only the information that needs to be clarified (i.e. they follow the pragmatic principle of not being overinformative, expressed by Grice (1975) for communication more generally). We not only learn from hearing explanations, but also from producing them (Chi et al., 1989, 1994), and so it is a common pedagogical principle to ask students to generate explanations. Explanations form a key tool for learning.

Similarly, we can also learn quite rapidly from demonstrations, which provide another succinct means for conveying important structure in the world. This learning requires applying our prior knowledge to infer what should and should not be generalized, but fortunately we can do so even from a young age (e.g. Van Damme et al., 2002). Both explanations and demonstrations provide powerful tools for learning rapidly. We develop the ability to exploit these tools early in life (Carpenter et al., 2005).

Analogical and relational reasoning, and abstraction: Some researchers have argued that analogical transfer and abstraction form crucial components of our learning (e.g. Gentner, 2003; Lakoff and Johnson, 1980; Gentner and Asmuth, 2017). These accounts often focus on fast, explicit transfer of the form explored by Gick and Holyoak (1980), for example. The structure-mapping algorithm (Falkenhainer et al., 1989) proposed for analogical reasoning is based on explicitly searching over possible isomorphisms, which tends to be infeasible in practice. However, there may be ways to implement it efficiently enough that it could be considered for complex cognitive models (Forbus et al., 2017), and in past work I suggested that implicit learning could provide heuristics to dramatically speed up this search (Lampinen et al., 2017). It's also reasonable to expect this ability to be related to education and other individual factors, since its been observed that features such as fluid intelligence may interact with the type of scaffolding provided to affect explicit analogical transfer (Kubricht et al., 2017). Indeed, older students are often better at transferring knowledge than younger ones (e.g. Chen and Klahr, 1999). These observations are suggestive of this type of transfer being a learned skill, rather than a cognitive primitive.

Much of the work on analogical and relational reasoning also focuses on the benefits of comparing multiple examples, which can lead to more reliable induction of abstractions or schemas and better transfer (Gick and Holyoak, 1980; Gentner and Asmuth, 2017). It has even been suggested that this comparison is a key way to understand the benefits of grounding (Jamrozik et al., 2016). However, in some past work, I found that seeing two different presentations of a mathematical concept lead to better learning overall than seeing either individually, but did not lead to significantly better abstraction of formal principles (Lampinen and McClelland, 2018). Thus it remains important to ask when reasoning about multiple examples leads to abstraction, and when it does not. One feature that is often important is explicitly considering the relationship between examples (Gentner and Asmuth, 2017), but it's likely that the quality of understanding of the examples matters as well.

Compositionality: Many researchers have argued that cognition must rely on strictly compositional representations in order to exhibit the systematic and productive generalization that humans are capable of (e.g. Fodor, 2001a, 2008; Lake and Baroni, 2018). However, an alternative approach has suggested that structured generalization can emerge from the structure of learning experience, without needing to be built into the model (McClelland et al., 2010; Hansen et al., 2017). This debate is complicated, and we will not try to resolve it. However, we will note that it can be difficult to define "compositional representations" in a way that offers an actual constraint on the model class (c.f. Zadrozny, 1992). Furthermore, there have been exciting observations that altering aspects of the training regime, rather than the architecture itself, may allow more compositional generalization (Hill et al., 2020). We will generally take the perspective that compositionality may be an emergent feature of learning, rather than a cognitive primitive. We will return to this discussion in greater detail in Section 1.4.1 below, as well as in Chapter 6.

Consciousness and explicit reasoning: Consciousness is a slippery topic, but unfortunately must be discussed, as it underlies many of the fast-learning systems above. Most of these rely on our ability to explicitly reason about concepts once we have sufficiently high-quality representations of them. The global workspace theory of consciousness (Baars, 2005; Dehaene et al., 2017) is closely aligned to this perspective. It states that conscious knowledge is precisely that which is globally accessible, and therefore with which we are most flexible. (C.f. also the higher-order-thought theory of consciousness (Rosenthal, 1990)). This perspective concords to some degree with the perspectives of Karmiloff-Smith (1986), who argued that once we re-represent knowledge to be explicit, we can use it more flexibly.

However, Karmiloff-Smith (1986) also argued that there are different levels of explicit representation of knowledge, and indeed some consciousness researchers have proposed more graded transitions from implicit to explicit (e.g. Cleeremans and Jiménez, 2002). Computational models of graded transistions have been proposed, where explicit knowledge is essentially learned by a separate system which reasons over implicitly learned representations (Cleeremans, 2014). This fits more with the work reviewed above showing a graded transition to explicit knowledge, built upon implicit understanding grounded in percetual-motor features (e.g. Goldin-Meadow et al., 1993), procedures (e.g. Hazzan, 1999), or more basic concepts (Wilensky, 1991; Patel and Varma, 2018). It is also aligned with the perspective that structure should emerge, as above. Thus, when thinking about the relationship between implicit and explicit knowledge is unavoidable, we will take the perspective that explicit reasoning is built upon implicitly learned representations.

Another place where explicit reasoning plays a role is in helping us to understand and learn from our own mistakes. For example, *deliberate practice* (targeting specific aspects of performance for improvement) has been claimed to be key to developing expertise (Ericsson et al., 1993; Ericsson, 2017). Yet to engage in deliberate practice requires meta-knowledge about what needs to be improved, and indeed this abstract understanding is an important component of expert knowledge (Feltovich et al., 2012).

Summary: We possess fast learning systems that are engineered by evolution, such as the hippocampus, and ones that are culturally transmitted, such as our ability to follow instructions in our native language(s). Together, they allow us to infer a great deal from little information, by leveraging our slowly-accumulated prior knowledge. They also allow us to flexibly adapt our behavior through practiced algorithms like following instructions.

1.2.3 Interactions between fast and slow learning systems

I claim that most transfer arises as a synergy between different kinds of learning across different timescales. In particular, the knowledge we have accumulated over our lifetimes (part of which has been accumulated by our cultures over millenia) allows us to constrain the hypothesis space for new learning, so that we can make accurate inferences from a few examples in a new situation.

From my perspective, this slowly-learned knowledge of the world can take multiple forms. It can occur in the mapping from inputs to our awareness, for example in visual cortex neurons which adapt to the regularities encountered over development (Barlow, 1975). However, it can also occur in the systems that implement higher-level and more rapid computations.[4] For example, the results on transfer in artificial grammar learning described above (Tunney and Altmann, 2001) or the results on transfer of simple harmonic oscillator strategies (Day and Goldstone, 2011) show that humans are able to transfer knowledge at the level of structures or algorithms. The limits of this transfer are as yet unclear, as are the time-scales over which different kinds of transfer can occur.

Limitations & tradeoffs: Of course, there are trade-offs to relying on transfer and prior knowledge. When new tasks are not well aligned with our prior knowledge, relying on prior knowledge can actually interfere with learning. For example, this observation is one piece of the argument that we made (Lampinen and McClelland, 2018) to explain the results of Kaminski et al. (2008). This is an illustration of the broader phenomenon of negative transfer — interference effects produced by transferring between non-isomorphic domains.[5] A wide variety of studies have observed phenomena like this (e.g. Luchins, 1942; Landrum, 2005). Our prior knowledge can be detrimental in situations that are very different from those we have encountered previously.

This brings us back to the broader point raised by Detterman (1993) above, that humans are often unable to efficiently or flexibly transfer knowledge to new situations. Instead, this flexibility must be a goal of education (Bransford and Schwartz, 1999), and learning what is transferable may require developmental time (Lampinen et al., 2017) if the quality of the representations is insufficient to support faster transfer. We can be efficient and flexible when our prior learning has set us up to be. We are not always. We can be mislead by mismatches between the past and the present, or we can simply fail to find the correct analogy.

1.3 Steps towards flexibility in deep learning

A great deal of recent work in machine learning can be interpreted as attempts to engineer machine-learning systems with greater flexibility. In particular, much of this work has tried to allow them to

[4]Clearly certain kinds of knowledge will lend themselves more easily to being learned in development and others will lend themselves more easily to being culturally transmitted or even learned by evolution. For the most part, I will generally assume that this knowledge emerges from experience or is culturally conveyed rather than being built in (Hansen et al., 2017), but my conclusions will mostly be agnostic to the origins of any particular piece of knowledge.

[5]In fact the tasks in Kaminski et al. (2008) *are* isomorphic at an abstract level — part of our argument is that they are not isomorphic at the level at which participants actually engage with them.

learn from fewer examples, or generalize better to data that weren't seen during training. This work typically follows one of two approaches. The first, multi-task learning, focuses on learning multiple tasks with a single model, in the hope that the additional constraints on the model's representations will cause it to learn more rapidly or generalize better. The second approach, meta-learning, focuses on learning how to learn tasks, in the hopes of learning much faster and generalizing better from extremely small samples. I suggest that multi-task learning therefore relates to the "slow" transfer processes I discussed above, and meta-learning relates to the "fast" ones. In this section, I will give a brief overview of both these literatures, after some comments on generalization.

1.3.1 On generalization in deep learning

Before diving into the recent developments in flexibility, it is worth reviewing what we know about generalization in deep learning. Many problems of flexibility can be seen from another perspective as a problem of generalization. Ultimately the problem of cognition can be seen as a control loop from multimodal stimuli to responses (Cisek, 1999, 2019). If we think of our lives as an unending sequence of inputs, influenced partly by our actions, behaving appropriately in any new situation we encounter can be seen as a generalization problem — the question is simply whether we can recognize the relationships between a new situation and prior ones, and use those relationships to generalize appropriately.

As a clarifying example, a sufficiently good language prediction model should be able to do translation without ever having been explicitly trained on it, simply because it is a reasonable use of language. For example, if the model is conditioned on pairs of English and French sentences which are translations of each other, and is then provided with new English sentences as input, it should be likely to output appropriate French translations. Indeed, recent work demonstrates this striking generalization in state-of-the-art language models trained only to predict missing words (Radford et al., 2019) — see below for further discussion. Thus, what from one perspective can seem to be an altogether different task requiring adaptation, can from another perspective seem like basic generalization. From this perspective, different notions of flexibility are united under one general notion of generalization.

Moreover, many critiques of the inflexibility of deep learning highlight failures of generalization (e.g. Marcus, 2018). To answer fundamental questions about the appropriateness of deep learning as a cognitive model, and to know whether we can trust AI systems to handle novel situations, it would be very beneficial to characterize the generalization capabilities of deep learning.

Unfortunately, we understand relatively little about generalization. We know that a deep model which can memorize arbitrary labels for every ImageNet image nevertheless generalizes well on the real dataset (Zhang et al., 2016). How can this be true? We know that a relatively small network (by modern standards) can be Turing-complete (Siegelman and Sontag, 1992) — that is, it can represent any computable function — so how can it be that after training so many of these models compute close approximations of the right function? Classic machine learning generalization bounds relied on restrictions on how much a model could memorize (e.g. Vapnik and Chervonenkis, 1971), so they cannot explain these results. In fact, these bounds are often anti-correlated with generalization performance, because more overparameterized deep models often generalize better!

There has been some recent progress on understanding these results, though that understanding is far from complete. With collaborators, we showed that (in a simpler deep linear model) the structure of the data, together with gradient descent and optimal stopping, effectively impose an inductive bias on the function computed (Lampinen and Ganguli, 2019). In particular, the most important dimensions of structure in the data are least contaminated by noise, and are learned earliest. Thus learning will capture most of the important structure in the data before corrupted and noisy signals are learned, ensuring that optimal stopping will yield good results.

Other lines of work have shown that aspects of the architecture may bias deep networks toward simpler regions of function space, which may therefore bias the networks towards solutions which generalize better (Pérez et al., 2019). Still other work has shown that computing generalization based on weight norms or model compressibility can yield bounds that behave more appropriately with over-parameterization (Neyshabur et al., 2018; Arora et al., 2018b). Thus, there are growing suggestions about how various features of our model architectures, algorithms, and datasets may contribute to generalization. However, there is not enough understanding yet to make strong claims about how to build models that generalize better.

Thus, most progress in building more flexible deep learning models has been empirically driven. In the next sections, I will review some of that work.

1.3.2 Multi-task learning

Multi-task learning is generally related to the “slow” learning systems I described in humans. The general idea is that relevant auxiliary tasks will serve as a useful inductive bias for the target task (Caruana, 1997). Typically, parameters are partially shared between the tasks, and these shared parameters are learned over long time-scales. This learning can be done either sequentially (where you use one task to pre-train the network for another), or simultaneously (where you learn multiple tasks at the same time or on alternating gradient steps). Auxiliary tasks need not be of the same type as the main task, for example reinforcement learning tasks can be supplemented with auxiliary supervised or unsupervised tasks like temporal autoencoding (e.g. Hermann et al., 2017), and unsupervised tasks can be used to pre-train for supervised ones (e.g. Wu et al., 2018).

Pre-training: One example of sequential multi-task learning is the extremely common practice of pre-training a network on some canonical task in order to use the representations of one of its hidden layers as a feature for learning some other related task. For example, pre-training on ImageNet (Deng et al., 2009) is often seen as a useful way of constructing a general feature extractor for vision tasks (Huh et al., 2016), even for quite different transfer domains (e.g. Marmanis et al., 2016). Even unsupervised pre-training can be helpful, since it helps identify the relevant axes of variation in the data, at least some of which are likely task relevant (Erhan et al., 2010). There is still active research on how to optimally pre-train features for various goals (e.g. Wu et al., 2018).

Pre-training is used on more than just vision tasks. In natural language processing (NLP) applications, the representations of words is often pre-trained on co-occurrence prediction tasks (e.g. Pennington et al., 2014). The more general language modeling task (predicting words conditioned on past — and possibly future — words) can serve as unsupervised pretraining for many tasks (e.g. Radford et al., 2019). Using larger sets of supervised language tasks as pre-training can be equally good, if not better (Raffel et al., 2019). In AlphaGo (Silver et al., 2016), the networks were pre-trained to predict expert go-players' moves, and then were tuned from that starting point using reinforcement learning. For other RL tasks, various pre-training approaches can be useful, such as training the agent to be able to reach diverse states (Gregor et al., 2016), or trying to learn adversarially discriminable skills (Eysenbach et al., 2019). The principle that pre-training a network can allow it to generalize better from smaller training sets appears to be quite general, although there is still some debate and ongoing research (He et al., 2018, e.g). When it is possible to collect enough data on the target task, pre-training may no longer be necessary (e.g. Silver et al., 2017).

There are also many remaining questions about how other features of training interact with transfer. Some researchers have argued that *disentangled* representations — loosely those that represent distinct features in distinct subspaces — will be helpful for future tasks (Higgins et al., 2018), while others have argued that disentangled representations are hard to define, and that the evidence that they are useful is weak (Locatello et al., 2019). Some recent work suggests that some forms of regularization may actually harm transfer (Kornblith et al., 2019). One possible explanation for this result is that regularization is too compressive in the transfer setting — it may compress away some real features that are useful for the transfer task, but not for the source task. However, the results might be different if the tasks were trained simultaneously (see below), rather than in a pre-training paradigm. Understanding when this interference occurs, and how to balance the generalization benefits of regularization with transfer benefits, will be an important area for future work.

Curriculum learning: A more general kind of pre-training is curriculum learning (Bengio et al., 2009) — the idea that training models on a structured progression of tasks can improve generalization. Pre-training on a single task is essentially a very simple curriculum. The idea of curricula began with Elman (1993), but was debated (e.g. Rohde and Plaut, 1997). However, subsequent work has demonstrated the importance of curricula both in toy settings (Gülçehre and Bengio, 2013) and in much more complicated tasks and models (e.g. Zaremba and Sutskever, 2014; Graves et al., 2016).

Of course, having to devise a curriculum for each task you want to perform requires substantial human effort, which has led to work on automated ways of deriving curricula (Graves et al., 2017).

Deriving curricula automatically is a difficult challenge — while approaches like tracking progress on many tasks and training on the ones where performance is changing the most (Baranes and Oudeyer, 2013), adversarially generating tasks of intermediate difficulty (Florensa et al., 2018), or learning curricula via meta-gradients (Such et al., 2019) can work in simple domains, in more complex task settings the problem remains open. When the task consists of a two-player game, play between agents as they learn can form a natural curriculum — the difficulty of the task increases precisely as the agent learns (Silver et al., 2017; Jaderberg et al., 2019). However, in more general non-competitive tasks, it is not clear how such an approach could help.

In recent work, my collaborators and I explored a new approach to automated curriculum generation for goal-conditioned reinforcement learning (Racaniere et al., 2020). We combined several ideas from the previous work in novel ways, in order to scale these approaches up to more complex tasks, and tackled novel challenges, such as automated curriculum generation in environments where the possible tasks vary between episodes. We also highlighted a challenge to naive approaches — as the complexity of the task space grows, it becomes very inefficient to explore task space at random, or by uniformly increasing difficulty. Most tasks in a complex environment will not be useful for the ultimate goal; for example, teaching a self-driving car to do a flip might be difficult, but it would not help with most tasks we want the car to perform. Human curricula are designed to efficiently lead learners to the desired competency, and we drew inspiration from this to propose a novel technique for pulling curricula towards a desired task distribution. These results represent a substantial step towards automated curriculum derivation in environments closer to the complexity of the real world. However, as we highlight in the paper, in more complex tasks it can be difficult to generate a curriculum without auxiliary information about the environmental structure. In these settings, the cultural knowledge behind human curricula, and the systems of mutually supporting tasks we have developed, will likely be necessary for achieving human-level intelligence.

Continual learning & avoiding interference: Curriculum learning raises another issue — what if you want your model to perform well at several tasks? If you switch to training on a new task, it may catastrophically interfere with your ability to perform a previous task (McCloskey and Cohen, 1989). The complementary learning systems perspective (McClelland et al., 1995; Kumaran et al., 2016) was intended in part to address this issue. Some of our new work in this area has shown that (at least under some circumstances) replay can be quite efficient — old items need only be replayed proportionally to how related they are to the new, interfering items (McClelland et al., 2020).

A number of other approaches have also been proposed recently, based on ideas like preserving parameters from updates proportionally to how important they are for old tasks (Kirkpatrick et al., 2016; Zenke et al., 2017), or learning tasks separately but distilling the knowledge into a single network later (Rusu et al., 2015), or by trying to use unsupervised learning to find good representations and allocate new tasks so that they don't interfere (Achille et al., 2018; Rao et al., 2019). Using

HyperNetworks (Ha et al., 2016) to specify task specific parameters may be helpful, as was observed contemporaneously in the preprint versions of Lampinen and McClelland (2019) and Oswald et al. (2020). There are also some meta-learning and memory-based approaches discussed in section 1.3.3 that can ameliorate this problem. Thus there are a variety of approaches that can address catastrophic interference, while still maintaining the benefits of curriculum learning. See Parisi et al. (2019) for a thorough recent review.

There is an additional challenge — much work on continual learning assumes that the boundaries between tasks are known, and that task identities are known. However, relaxing one or both of these assumptions might be more realistic (Ven and Tolias, 2018). Some recent work has attempted to address this issue by developing algorithms that infer the tasks and task transitions from a continuous stream of data (Nagabandi et al., 2019).

However, beyond simply avoiding interference, a major hope is that prior knowledge will help with learning of a new task. Ideally, prior knowledge would be useful even in superficially dissimilar domains, if the underlying structure is similar. While most curriculum learning work at present still samples the curriculum from a very narrow set of tasks, such as navigation goals of varying difficulty (Florensa et al., 2018), humans seem to be able to leverage analogies from much more disparate domains, such as using the flow of water to understand the flow of heat (see Falkenhainer et al., 1989). It remains a challenge for machine-learning systems to learn different types of knowledge from the variety of tasks that humans experience, in such a way that prior learning supports new learning rather than interfering with it (Mitchell et al., 2018). I will provide some of my perspectives on the potential for positive transfer in Chapter 5.

Simultaneous multi-task learning: Many multi-task learning approaches train on the tasks simultaneously rather than sequentially. For example, simultaneously training a natural language translation system to do image captioning in the target language improves its translation performance (Luong et al., 2016). Even training it to translate between multiple language pairs is beneficial (Dong et al., 2015). This approach can even lead to zero-shot generalization to translation between language pairs never seen together in training (Johnson et al., 2016; Platanios et al., 2017). Training a shared model on many natural language tasks can substantially improve its generalizability (Raffel et al., 2019). For this reason, recent natural language benchmarks have focused on broad sets of challenging tasks rather than single-task evaluation (Wang et al., 2019b,a).

The idea of multi-task learning has proven even more critical in Reinforcement Learning (RL), where auxiliary tasks have been suggested as a key approach to overcoming the problem of reward sparsity (e.g. Le Cun, 2016). Auxiliary tasks have been used in a variety of RL settings, for example in grounded language learning (Hermann et al., 2017) or game playing (OpenAI et al., 2019; Vinyals et al., 2019).

The broad observation that deep networks will learn representations that represent shared structure in the tasks they perform, and can exploit this shared structure to generalize better, is not new.

It was observed at least as long ago as Hinton (1986), and has continued to intrigue researchers since (e.g. Lampinen et al., 2017). In particular, it is a key feature that separates deep networks from simpler statistical learning architectures, and allows them to uncover and exploit deep structure in the world (Rogers and McClelland, 2008). This structure extraction may support some of the most interesting kinds of transfer that deep networks demonstrate, and may have provided substantial benefits in the various projects reviewed above.

Learning which parameters to share: Most the work above simply fixes architectures with pre-specified shared and unshared weights. However, there are other approaches that attempt to learn which weights to share. The work of Achille et al. (2018), referenced above, is one example of learning what to share. Other authors have considered using evolutionary algorithms to decide which subsets of modules should be shared (Fernando et al., 2017). Some other approaches can also be seen from this perspective, for example choosing a sparse subset of modules for each forward pass (Shazeer et al., 2017) or using a HyperNetwork to generate the weights for other networks (Ha et al., 2016). This latter work has led to a great deal of productive research in domains ranging from language (Platanios et al., 2017) to meta-learning of visual classifiers (Garnelo et al., 2018; Li et al., 2019). Both HyperNetworks and using subsets of modules can be seen as a way of learning which weights should be shared or separate between different "sub-tasks" of a task. Some of the work discussed in section 1.3.3 can also be viewed as learning what parameters should be shared and which should be separate. While in principle a fully-shared network could learn which parameters to share just by gradient descent, in practice a more structured approach to this problem can be useful.

Tasks as an input feature: The most flexible approach to multi-task learning involves simply providing task representations as an input to the model and letting everything be learned, rather than pre-specififying anything about how computations should be shared and separated. This approach has the benefit that the system can potentially learn to generalize to novel tasks. Research on general value functions in reinforcement learning (Sutton et al., 2011) — value functions which take a task specification as an input — provide one inspiration for this approach. Task-conditioned RL approaches have exhibited success in certain cases, for example generalizing to unseen natural-language task specifications (Hermann et al., 2017). With increasing dataset size, this approach is becoming more feasible in complex natural langauge processing tasks (Raffel et al., 2019).

Reinforcement learning: Reinforcement learning (RL) has a long history within neuroscience & cognitive science (O'Doherty et al., 2003; Niv, 2009; Dabney et al., 2020), and artificial intelligence (Sutton and Barto, 2017). RL tackles the fundamental problem of learning in tasks where an agents actions affect the environment it is in. This interaction means that it is impossible for the agent to observe and learn from all counterfactual possibilities, unlike in supervised learning where a model may update its predictions for every alternative label. It also means that supervision can be harder to provide, because the supervisory signal must take into account the actions of the agent. Both of these factors make RL settings more realistic for real world tasks. They may therefore provide an important bridge to more flexible models.

Indeed, some of the major recent successes of deep learning have been driven (at least in part) by reinforcement learning. For example, using RL to play Atari videogames (Mnih et al., 2015) was a powerful demonstration of the ability of deep learning to perform complex tasks. More recently, RL-based models have achieved superhuman performance in complex games such as Go (Silver et al., 2016, 2017), and expert human performance in extremely complicated video games (OpenAI et al., 2019; Vinyals et al., 2019).

However, there remain difficult challenges to applying RL to complex problems, and it remains an area of active research. Applying genetic algorithms might increase data-efficiency in some cases (Petroski et al., 2018), and representing value as a distribution (instead of a point estimate) may improve performance (Bellemare and Dabney, 2017). Clever replay schemes, such as relabeling data, may allow for more effective learning from a small amount of experience (Andrychowicz et al., 2017). Using memory lookups as a cue may help solve the long-term credit assignment problem (Hung et al., 2019). Still, RL algorithms are often unstable, and seemingly inconsequential changes like rescaling rewards may substantially alter results (Henderson et al., 2018). Making RL reliable on complex tasks remains a challenging goal.

Summary: To summarize, curriculum and multi-task learning can help set up representations that allow deep learning models to generalize better from less data, or to learn tasks that would not otherwise be learnable. They do so because the additional constraints imposed on the network's representations through auxiliary tasks can help the network to uncover the true underlying structure in the environment. While multi-task learning and curriculum approaches do not support all the kinds of flexibility that humans demonstrate, they are a key piece of the puzzle of how deep networks can learn faster and generalize better, and may be one important feature that separates humans from our best current models.

1.3.3 Meta-learning & related approaches

The fundamental insight of meta-learning is that there is a continuum between data and tasks. We can interpret each (input, target) tuple as a simple task, and we can interpret subsets of a dataset as sub-tasks, for example all the dog images contained within ImageNet form a semantically distinguishable sub-task of the overall task. Analogously, we can often interpret a single task as being just a point in a larger space of tasks. Most meta-learning models exploit these insights by having the architecture adapt to a given task within its activations instead of its weights, just as a CNN would adapt to the fact that its current input was a dog image rather than a tree image within its activations. Learning to adapt to new tasks in this way has been shown to allow for much more efficient learning on new tasks, and may be key to modeling human intelligence (Hansen et al., 2017).

Basic meta-learning: The basic meta-learning approach (representing a task within activations) has been applied to many settings. For example, it has been used to learn to classify new images based on single positive exemplars of each class (Vinyals et al., 2016; Ravi and Larochelle, 2017). It has also been used to teach recurrent networks to solve simple reinforcement learning problems (Duan et al., 2016; Wang et al., 2016; Stadie et al., 2018).

There have been many variations on this approach. Some meta-learning work has exploited both slow and fast weights with some success (e.g. Munkhdalai and Yu, 2017), taking one perspective on the dichotomy I proposed above. Many approaches have exploited other tricks that are broadly useful in machine learning. For example, some work has leveraged unlabelled examples along with a few labelled ones to yield better meta-learning results (e.g. Ren et al., 2018). Other work has shown that attention-based models are useful (Reed et al., 2017), as in many other machine-learning applications (e.g. Vaswani et al., 2017; Gregor et al., 2015). Many recent approaches have constructed task representations, and used these to classify new data points (Li et al., 2019; Ravichandran et al., 2019). Finally, some work has shown that Bayesian inference may be a useful tool in these settings (Burgess et al., 2016).

Optimization-based methods: Another productive line of meta-learning work uses an inner optimization loop to tune the model to each task. This work largely follows from the work of Finn et al. (2017a) on Model-Agnostic Meta Learning (MAML), an approach based on optimizing the model so that it could adapt well to a new task in a few gradient steps. MAML has led to many follow-ups (e.g. Finn et al., 2017b, 2018; Nichol et al., 2018). At least in some settings, it may only be necessary to adapt a few task-specific parameters rather than the whole model (Zintgraf et al., 2018) — this approach may be thought of as a form of task representation.

Memory-based: There are a variety of meta-learning approaches that are based on a non-parametric memory, and generally some form of key-value attentional lookup over it. A general extension of the basic meta-learning approach to the case with memory is given in Santoro et al. (2016). Other approaches are based on using memory only at testing, e.g. by tuning the network rapidly to perform well on similar examples, as proposed by Sprechmann et al. (2018).

More flexible variants have also been proposed. For example, the differentiable neural computer (DNC), proposed by Graves et al. (2016) is able to receive a graph structure as inputs, learns to store it in memory, and then learns to use that stored information to solve problems like computing shortest paths. That result can essentially be seen as meta-learning — the architecture has the flexibility to learn and reason from new knowledge rapidly. However, this flexibility remains fundamentally within the computations done for a task which is slowly learned, and does not by itself allow the more general flexibility that humans exhibit.

Language & zero-shot performance: There is some work that has explored related problems from a language-based perspective. For example, Larochelle et al. (2008) considered the general problem of behaving in accordance with language instructions as simply asking a model to adapt its response when conditioned on different "instruction" inputs. Since this approach does not require data on the new task, it is a form of zero-shot learning. Later work explored zero-shot classification based on only a natural language description of the target class (Socher et al., 2013; Romera-Paredes and Torr, 2015; Xian et al., 2018). Many of these used very simple language, e.g. a single word (the target class label), and used tricks such as combining prior classifiers based on their labels' word-vector-similarity to the target (Norouzi et al., 2014; Changpinyo et al., 2016, e.g.). More recently, there has been a productive line of research in using language to compose network modules for question answering (Andreas et al., 2016b, 2017b), at least in toy domains.

There has also been some work on reinforcement learning systems that learn to follow natural language instructions in simple environments (Hermann et al., 2017; Oh et al., 2017). In past work with collaborators, we showed that more realistic environments improved the generalization exhibited by these instruction-following systems (Hill et al., 2020). Other work has shown that language can form a useful intermediate representation for a simple form of hierarchical reinforcement learning (Jiang et al., 2019).

More recently, Radford et al. (2019); Brown et al. (2020) showed that a language model trained on an extremely large corpus of curated websites actually acquires some meta-like abilities, for example the ability to translate languages "zero-shot" (i.e. when conditioned on examples of translation pairs and then given a new sentence, it produces a translation). It is also able to summarize, answer questions, etc. It presumably is able to accomplish these tasks because the corpus contains some translation pairs in context or summaries on a page, and so those tasks essentially compose a small part of the whole language modeling problem. It's worth noting that the performance on these tasks is rudimentary, and far from models that are trained for these tasks in a supervised way. Even in follow-up work (Brown et al., 2020), a much larger model still shows only moderate flexibility, and the authors note that grounding and richer training are likely necessary to achieve full language competence. Despite this, the success of these models is an impressive demonstration of the power of prediction to extract deep latent structure in the world, and the power of broad training data distributions to teach flexibility.

Demonstrations: The issue of learning from demonstrations has also been considered in the machine learning literature for some time, because of its potential applications in problems where we know what behavior we want, but not how to encode it. There the problem has generally been referred to as "inverse reinforcement learning" (Ng and Russell, 2000), i.e. the problem of inferring a value function from observed behavior. A large number of approaches to this problem have been proposed (e.g. Ng and Russell, 2000; Abbeel and Ng, 2004). Recently, approaches combining meta-learning and deep learning have achieved some success. For example, Finn et al. (2016) present an algorithm which can learn to infer a reasonable approximation of the objective function from a single demonstration. This work shows that neural networks can meta-learn to learn from demonstrations.

Relational and analogical reasoning: There are a number of other approaches that attempt to explicitly build relational inductive biases into deep-learning architectures. Relation networks (Santoro et al., 2017) build in this prior by having the architecture explicitly relate different parts of the input, and achieve better performance on answering relational questions about visual scenes. Architectures that directly allow for true relational binding can be beneficial for a variety of applications, especially natural language processing or symbolic reasoning (e.g. Smolensky, 1990; Smolensky and Goldrick, 2014; Huang et al., 2017). Graph-structured architectures form a very natural way of representing few-shot learning problems (Garcia and Bruna, 2018), and more generally graph-structured or other relational inductive biases have been suggested as a promising direction in deep learning (Battaglia et al., 2018). How these inductive biases benefit (or limit) learning is an important direction for future research.

However, relational reasoning is constrained by training as well as the architecture. For example, choosing the negative examples that a network learns from to explicitly contrast relational hypotheses can help to yield more relational reasoning (Hill et al., 2019). Exploring how architecture and training interact to produce relational reasoning will be an important future direction.

Abstraction: There has also been some work on explicitly building abstraction capabilities into machine learning systems. For example, in reinforcement learning the idea of options (Sutton et al., 1999) and hierarchical reinforcement learning more broadly (e.g. Botvinick et al., 2009) are essentially encapsulations of temporal abstraction, where a sequence of actions can be represented as a single higher-level action. For example, we can think of going to the office as a single action, rather than a sequence of many steps. Similar attempts have been made to allow deep learning models to share knowledge across tasks, with some success (e.g. Tessler et al., 2016). Other approaches have tried to infer hierarchical task representations during meta-learning, for better generalization (Yao et al., 2019).

There have also been a variety of attempts to combine deep learning methods with approaches based on programming. For example, Neural Programmer-Interpreters (Reed and de Freitas, 2015) essentially endow a recurrent network with the ability to call sub-routines, and a stack of memory for these sub-routines to use. Applying these ideas to meta-learning problems has been reasonably successful, especially with carefully chosen algorithms for integrating knowledge across tasks (Devlin et al., 2017, e.g.). Similar techniques have been applied to many domains, such as learning from demonstrations (e.g. Xu et al., 2017). Combining the old ideas of cognition as executing symbolic programs (Newell and Simon, 1961) with the techniques of deep learning can yield improvements in flexibility.

However, most of these methods are still not as universally flexible as humans. The number of abstractions is usually fixed, often abstractions cannot be composed from other abstractions, and abstractions are inflexible to other demands. For example, a system that has learned an option for walking to a goal will not necessarily be able to change to running to the goal without learning this option from scratch. Thus there are still limitations to these approaches at present.

Model-based reinforcement-learning: Model-based RL methods provide a useful factorization of the RL task, that can allow the same model to be used for a new task with a different reward function (e.g. Laroche and Barlier, 2017). More flexible hybrid model-based methods, such as letting an agent learn to plan (Tamar et al., 2017), show potential promise. However, many of these suffer from stability issues, as prediction errors compound over rollouts (Talvitie, 2014). However, treating these rollouts as a potentially flawed imagination, and letting the model learn to interpret them can help (Racanière et al., 2017), as can rolling out in latent space rather than in observations (Gregor et al., 2019). It seems likely that one component of flexibility will be learning models that can be reused for new purposes. However, it is as yet unclear what those models will look like, and whether they will need to have planning as an inductive bias at all. At least in some circumstances, planning-like behavior can emerge in a model-free architecture with an appropriate recurrent structure Guez et al. (2019).

There has also been a substantial amount of work on the succesor representation, which is a hybrid between model-based and model-free methods that caches state transitions and values. The successor representation may serve as a useful compromise between the model-free and model-based methods in some cases, and there is some evidence that humans create successor-like representations on some simple tasks (Momennejad et al., 2017). However, the successor representation cannot adapt well if state transition probabilities change drastically, and other approaches such as task clustering must be adopted to accomodate these challenges while maintaing flexibility (Madarasz and Behrens, 2019).

Furthermore, both model-based and successor-representation-based methods only handle replanning if given a new reward or value function. They thus offload a substantial part of the problem of adaptation to another system. Combining these methods with the methods I propose in later chapters might allow for a more complete solution to the problem of adaptation.

Other work: There is a variety of other work that has exploited different perspectives on meta-learning. Some of this work could be useful for thinking about flexibility. For example, Xu et al. (2018) proposed using meta-learning to adapt hyperparameters of reinforcement learning algorithms across tasks. Other work has attempted to meta-learn auxiliary tasks for transfer, based on improvement on target tasks (Liu et al., 2019). Some work has even attempted to combine these, using meta-gradients to choose auxiliary tasks (Veeriah et al., 2019). Other research has shown that reframing continual learning as a meta-learning problem (of learning to learn without interference) can be effective (Velez and Clune, 2017), at least in simple settings. All of these approaches allow for better adaptation to new tasks or environments.

In addition, there has been some work showing that basic neural network models can adapt rapidly to new data that is consistent with prior knowledge, simply by optimizing weights specific to that data while freezing the remaining weights of the network (Rumelhart and Todd, 1993). This approach only works if there are weights that are specific to the new item(s), so it is not a general kind of flexibility. Nevertheless, this approach has been applied to understand human semantic cognition (Rogers and McClelland, 2004). More recently, I applied it to one-shot and few-shot learning of words in a language-modeling task (Lampinen and McClelland, 2017). Thus under some circumstances adapting item-specific parameters can yield a certain kind of flexibility, even at the scale of large machine learning tasks. This observation provides further evidence for the general point that information which is consistent with prior knowledge can be rapidly integrated (McClelland, 2013; McClelland et al., 2020).

Unintentional "flexibility": There has also been some interesting work on flexibility that emerges accidentally under standard training of deep learning systems. In particular, adversarial examples (Szegedy et al., 2014) are cases when adding a very small perturbation to an input can radically alter the network's output. This drastically altered behavior is a kind of flexibility, but it is not the desired kind. Instead, these appear to be evidence for the fact that deep networks are inherently chaotic systems, which can respond in surprisingly sensitive ways to their inputs. However, it's worth noting that humans can be susceptible to more extreme adversarial perturbations derived on deep networks (Elsayed et al., 2018b), and that many perturbations are human-interpretable even if we would not make the same mistake (Zhou and Firestone, 2019).[6] Furthermore, adversarial examples can be exploited, e.g. for better training (Goodfellow et al., 2015).

Elsayed et al. (2018a) demonstrated an even more interesting type of unintentional flexibility: deep networks can be "reprogrammed" by an input to solve a different task.[7] This reprogrammability is a step closer to the flexibility that humans have, but the "reprogramming" inputs have to be derived via an optimization process, and tend to be uninterpretable. Thus there is no systematicity in this flexibility. However, my interpretation of these results is that they are encouraging evidence that these models have the capacity to be extremely flexible under appropriate conditions. All that is needed is to train the models to be flexible systematically.

Meta-learning AI itself: Clune (2019) argues that we should meta-learn all aspects of the AI engineering process. In particular, he suggests that we should meta-learn the architectures, algorithms, and the tasks that we use to train our AI systems. This approach is indeed an exciting direction for future work, and may ultimately prove fruitful, but in these settings it is more complicated to determine what the over-arching reward or loss should be, and how to represent the features of learning themselves. Recent work has become to hint at potential solutions to these problems, but it will be some time before we can validate these techniques on tasks of the scale at which more mature architectures and algorithms are evaluated.

Summary: To summarize, meta-learning has made progress on several fronts. It is starting to solve the small data problem, by allowing networks to learn efficiently from a small number of

[6] Although this latter claim has been debated (Dujmovic et al., 2020).

[7] Although not one completely unrelated to the main task the network was trained on — the network was trained on a vision task, and their reprogrammed tasks were just other vision tasks. It would be interesting to explore the limits of this "reprogrammability."

examples (e.g. Wang et al., 2016), although they require a large number of training tasks in order to do so. At the other extreme of very large data, deep networks can generalize to some extent to tasks which are only hinted at by the trained task distribution (e.g. Radford et al., 2019). It has been suggested that meta-learning the AI architectures, algorithms, and tasks themselves may be the approach to creating artificial general intelligence in the futuer. However, there are a number of key features of human flexibility that remain unexplained by current approaches. In the next section, I will relate transfer and flexibility in humans and deep networks, and discuss the features that are still missing.

1.4 Relating flexibility in humans and neural networks

The encouraging progress on multi-task and meta-learning in recent years suggests that cognitive models exploiting these techniques may help explain human flexibility and transfer (Hansen et al., 2017). In this section, I will attempt to relate the aspects of human and network flexibility that I outlined in the previous sections.

First, it generally seems that the division between fast and slow transfer is applicable both to the machine learning literature (meta-learning vs. multi-task) and to human transfer, as we highlighted above (and in prior work, namely Lampinen et al., 2017). Following complementary learning systems, I suggest that broadly our slow learning of structure in the world happens over the course of developmental time in a multi-task fashion. Algorithmically, I suggest that this occurs because networks in our brain come to represent and exploit structural similarities across the many tasks we experience. We also explicitly practice using these slowly-learned representations to support rapid transfer & learning, in educational settings as well as in everyday experience more broadly. Thus I think that cognitive models should employ both slowly-learned shared representations, and a system that allows for rapid and flexible reasoning over them. I argue that incorporating both aspects will be key to modeling the full range of human flexibility — one of my main goals in this dissertation will be to construct a model which exhibits these characteristics.

It is also important to note the differences between the systematic, structured training that humans encounter in our systematic development and culturally-constructed educational systems, and the unstructured, IID training that deep learning models canonically receive (Smith and Slone, 2017). While curriculum learning addresses some of the sequential learning in development and education, and meta-learning addresses part of the learning-to-learn aspect, the full training on flexibility is generally missing. For example, humans learn a great deal from explaining as well as simply doing, yet we rarely train our machine learning models to explain their actions. Given the sensitity to training data that both humans and neural networks display, we cannot expect deep learning models to capture human behavior completely under drastically different learning regimes. It is important to develop richer, more structured educational paradigms for neural networks, both in order to use them as models of human intelligence, and to develop more human-like artificial intelligence systems.

In summary, I believe that deep learning systems remain promising cognitive models. They have successfully modeled a wide range of phenomena ranging from low-level neural activity to cognition. Furthermore, they are compelling because they are some of the only systems to successfully achieve human performance at difficult tasks like visual object recognition or playing go. They even have some inherent (if often unstructured and unintentional) flexibility, as indicated by adversarial examples and reprogramming, and more recent methods like meta-learning have given them more systematic flexibility.

I suggest that humans are similarly flexible. I suggest that the key to our flexibility is that we learn over the course of development and education to exploit our flexibility in systematic ways, in order to be adaptable in new tasks and situations — just as a meta-learning system learns over many tasks how to learn rapidly on a new one. This flexibility does not necessarily need to be an explicit target of the learning procedure, e.g. the results of Radford et al. (2019), discussed above, show that training a language model on a large enough text distribution gives some generalization to related tasks like translation. However, that network required far more training than could be assumed for humans, and still lacked some of the flexibility that humans have. I suggest that its weaknesses are due to its lack of multiple tasks to constrain the representations, the lack of an architecture explicitly designed to allow synergies between fast and slow learning, and the lack of the systematic, structured training at the scale that humans experience. However, there are alternative perspectives. We will consider one of the most frequent ones in the next section.

1.4.1 On compositional symbol manipulation

One of the most frequent criticisms of deep learning is that it lacks the compositional symbol manipulation ability that humans possess (Fodor and Pylyshyn, 1988; Lake et al., 2017; Lake and Baroni, 2018; Marcus, 2018, see also above). This argument has inspired a variety of works that incorporate symbol-like processes into the representations of deep models (e.g. Andreas et al., 2017a; Mao et al., 2019). These works explicitly constrain the representations of the models to use language-based representations, or symbolic (perhaps probabilistic) programs.

Before we discuss the merits of these approaches, it's worth considering the motivation. The notion of compositional cognitive representations was introduced by Fodor and Pylyshyn (1988), motivated by the importance of compositionality as an axiom in linguistics. However, within linguistics there is a growing recognition that it may be necessary to discard strict compositionality to properly understand semantics (Goldberg, 2015; Potts, 2019); ironically, this change is due in part to the success of deep learning models in natural language processing domains. In fact, theoretical work shows that the strict notion of compositionality does not even constrain semantics — any semantics can be rewritten to be compositional (Zadrozny, 1992). Furthermore, even Fodor (2001a) argued that language is not compositional (although he maintained that thought is).[8] Given that the compositionality of language motivated the arguments for compositionality in cognition, what should we think if language is not strictly compositional? To elaborate on this point, consider the argument of Potts (2019):

> "The usual story is that compositionality is crucial to our ability to produce and understand creative new combinations of linguistic units, because it offers guarantees about the systematicity and predictability of new units. However, these observations alone do not imply compositionality. The interpretation of a given phrase could be systematic, predictable, and also determined in part by global properties of the utterance, the speaker, the discourse situation, and so forth. And, indeed, it seems to me that our everyday experiences with language are in keeping with this."

It seems to me that similar reasoning applies to the compositionality of representations in cognitive models.

Despite these arguments, the debate over compositionality persists, both within linguistics and cognitive science. I think that part of the reason is that compositionality can be challenging to define. The arguments above apply to a standard linguistic definition: "the meaning of a complex expression is determined by its structure and the meanings of its constituents" (Szabó, 2017) — that is, that the meaning of a complex expression cannot be influenced from context outside the expression. Part of the conceptual challenge is that researchers sometimes equivocate between this strong notion of compositionality and weaker notions, e.g. that "expressions have internal structure." However, to the extent that it is argued that compositionality is a feature deep learning models lack, the definition used cannot be a weaker definition, since the representations of these models acquire internal structure insofar as it is afforded by the training data (e.g. Mikolov et al., 2013; Vankov et al., 2019).

Indeed, the state-of-the-art systems on complex natural language tasks do not use any symbolic or compositional inductive biases (e.g. Radford et al., 2019; Raffel et al., 2019). By contrast, the works that incorporate language or programs into the model representations are typically demonstrated on small toy experiments where the world decomposes nicely into simple elements (Andreas et al., 2017a; Mao et al., 2019). Even within these carefully constructed environments, when symbolic inductive biases are compared to more end-to-end approaches, the latter often prove superior. For example, using language as a latent bottleneck on a toy visual meta-learning task improves task performance compared to not using language at all (Andreas et al., 2017a), but using language as just an auxiliary signal performs even better (Mu et al., 2019). At least in that setting, language is a useful learning signal, but a harmful constraint. It is often the case that building in inductive

[8] It is not even clear if syntax is safe as a computational principle of language processing, as recent work has shown that language-selective brain regions respond mostly to local word transition probabilities, rather than syntax *per se* (Mollica et al., 2020).

biases and domain specific knowledge has proven detrimental in the long run — Sutton (2019) calls this observation the "bitter lesson that can be read from 70 years of AI research."

As another example, relatively unstructured deep networks sometimes outperform carefully engineered symbolic methods even in strictly symbolic domains like mathematics. Lample and Charton (2019) recently showed that a deep transformer model outperforms Mathematica (and Maple and Matlab) at symbolic integration and solving differential equations (Lample and Charton, 2019). Given that Mathematica is an extremely sophisticated symbolic reasoning system, and mathematical reasoning is probably the most symbolic human task, this result is quite surprising. However, Davis (2019) pointed out that the comparison is not truly balanced. That response raises some valid points about differences in the set of possibilities and solutions that the two approaches are considering. It is clear that we could not yet replace Mathematica with a deep network. However, it does make it more challenging to assert that deep networks cannot do compositional symbolic reasoning.

One objection raised by Davis (2019) is that, while the deep network is 98.4% accurate (at 1 beam) in the domain of problems given, compared to 84% from Mathematica, Mathematica will never produce an incorrect answer. That is, Mathematica is either correct or it times out, whereas the output of the model is not deterministically informative about whether the answer is correct. This objection is valid, up to a point, and is one of the reasons that deep models will not fully replace Mathematica anytime soon. However, when it comes to human reasoning, this point becomes more challenging. When humans make a compositionally-valid interpretation 80% or 95% of the time, it is presented as evidence of their compositional symbolic skill (Lake et al., 2019), yet when a deep model achieves 98.4% accuracy on much more difficult problems, it is depicted as a failure of the model class to exhibit compositional symbolic reasoning. Because it is difficult to know how to attribute mistakes in either humans or models, and it is difficult to match the scale and experience of humans in our contemporary models, it is not clear how to produce a fair evaluation of both that would decide the issue of whether symbol manipulation is a necessary ingredient of cognition.

It seems intuitive that symbol-like systems and particular compositions of representations would be helpful for particular types of reasoning. However, maintaining a strictly symbolic or compositional representation requires that the decomposition be imposed a priori. If there are many possible ways to decompose our knowledge, it might be more useful to flexibly decompose and represent knowledge according to the task at hand. This trade-off might be one reason that compositional inductive biases could be helpful in toy cases, but harmful on complex real-world data.

Instead, one might hope for a model that can construct appropriate decompositions, and could therefore allow for any inference in an appropriate context. To do so, the model would need to avoid considering the space of all possible decompositions, because one of the main issues with using strictly compositional representations (and symbolic reasoning more generally) is the exponential number of possible inferences in complex tasks. However, one key success of deep learning models is to find extremely good direct approximations that reduce (or entirely avoid) searching. Thus, one might hope that compositional structure might emerge in the representations and behavior of a deep learning model, insofar as it is relevant and useful.

1.4.2 On scale & emergence

Thus, an alternative possibility is that compositionality and symbolic-rule-like reasoning may be an emergent feature of learning, rather than a learning mechanism (McClelland and Plaut, 1999; McClelland and Patterson, 2002; McClelland et al., 2010; McClelland, 2010). Indeed, facility with symbolic mathematical reasoning emerges only after a great deal of experience in the domain (Burger and Shaughnessy, 1986; McClelland et al., 2016). Likewise, cross-cultural comparisons have shown that speakers of a language without number words are unable to express or remember exact quantities (Frank et al., 2008). Similarly, Gleitman and Gleitman (1970) showed that only graduate students (and not undergraduates or high school graduates) exhibited a strong ability to understand three-word compound nouns, despite these being a relatively basic construction that should be easily parsed according to the syntactic rules of the language. It is important to consider the possibility that our ability to generalize in symbol-like ways is an emergent property of our lifetime of experience in cultural systems that emphasize formal reasoning. It is difficult for researchers to think outside of the cultural framework in which we have been educated, and to consider how differently we might reason if our education had been different.

This obvservation leads to the possibility that our current deep learning training paradigms are just too unstructured, our tasks too simple, and our models too small for human-like flexibility to emerge. If we could immerse a deep network with as rich and recurrent an architecture (and as many parameters) as the brain in a rich lifetime of experience and education, would it be able to reason like a human? Changes in scale can often result in qualitatively different *emergent* behavior — "more is different" (Anderson, 1972). It has been suggested that this emergence might underlie many important aspects of human intelligence (McClelland, 2010), from semantic cognition (Rogers and McClelland, 2008; Saxe et al., 2019) to consciousness (Chalmers, 2006). It is important not to underestimate the difference in scale between deep learning models and the human brain. While extremely large models may have billions of parameters (e.g. Radford et al., 2019), the human brain has hundreds of trillions of synapses (Drachman, 2005), each of which is much more complex than a single weight from an artificial network. Furthermore, while modern meta-learning approaches may expose a machine learning system to many closely related tasks, they do not approach the years of experience in disparate domains that humans experience (c.f. Mitchell et al., 2018). There is room left for emergence.

Indeed, changes of scale have driven many of the recent successes of deep learning. The rise of deep convolutional neural networks in computer vision was driven in large part by increasing dataset size (Deng et al., 2009), combined with increasing computational power and efficient implementations of neural networks on GPUs (Krizhevsky and Hinton, 2012). It is both intuitive, and supported by long-standing theoretical results (Bartlett and Mendelson, 2002), that generalization improves with increasing dataset size. As noted above, recent work in machine-learning theory has shown the less intuitive result that qualitatively different results may occur when optimizing deep neural networks with many parameters than shallower or smaller ones — overparameterization can actually be beneficial (Dauphin et al., 2014; Arora et al., 2018a). Perhaps these factors suffice to explain the gap between human flexibility and that of deep models.

In support of this hope, the results of Radford et al. (2019), discussed above, offer a powerful example of the emergence of flexibility in machine learning. While natural language translation seems like a difficult machine-learning problem on its own (Wu et al., 2016), the results of Radford et al. (2019) show that a passable translation ability can emerge simply from training a large enough word-prediction model on a large enough corpus of webpages, and conditioning it on a few translation pairs. That is, the model is able to translate simply because translation is a systematic use of language, and so is implied by learning to predict language well enough. This result is surprising and promising.

Relatedly, my collaborators and I showed that increasing various aspects of environmental richness and realism improved compositional language generalization in RL agents (Hill et al., 2020). We found that making the agent more embodied (comparing an egocentric frame of reference to an allocentric frame in a 2D task) improved language generalization. We also found that switching from making a single-frame decision to behaving over time in an RL setting resulted in compositional generalization increasing from around 75% to 100% on the same abstract language generalization task. These results support the idea that more systematic generalization may emerge from more realistic and varied training regimes, unlike the extremely simplified settings that are sometimes used to critique deep learning models (see e.g. Lake and Baroni, 2018). Indeed, building language models with more embodiment, more realistic environments, and more integration and interaction between different systems may be key to more human-like language understanding (McClelland et al., 2019).

Furthermore, as I noted above, planning-like behavior can emerge in a model-free architecture, if it is is allowed to perform recurrent computations between actions Guez et al. (2019). That is, abstract behaviors like planning do not necessarily need to be built into the system — they can emerge from appropriate experience. This possibility is especially important when considering the rich cultural tools humans have built for transmitting knowledge and structuring the thinking of future generations (see above). How flexible and systematic could an unstructured deep neural network be after years of structured schooling?

Unfortunately, the answer is still unclear — these demonstrations of emergent flexibility have not yet reached human-level generality. While the model of Radford et al. exceeds non-trivial translation baselines, it is far from reaching the performance of a state-of-the-art model trained specifically for translation. These models in turn are not yet as sophisticated as humans at translation, perhaps in part for reasons outlined by McClelland et al. (2019).

Thus, we simply do not know enough yet to determine whether the emergent effects of scale, curricula, and more realistic tasks are the only difference between the cognitive flexibility of humans and the comparatively inflexible intelligence displayed by neural networks, or whether we will need to incorporate symbol manipulation or another paradigm to capture the flexibility and generality of human intelligence. While the dominance of symbol-free deep networks in domains like natural language translation is promising, the ultimate answers will only come with further research and scale. In the next section, I discuss some abilities that are missing, at least from the scale of models that we currently have. In the remainder of the dissertation I will propose mechanisms that solve these problems at a feasible scale, and without requiring compositional symbol manipulation as an architectural axiom.

1.4.3 What's still missing from current models?

Deep learning systems still lack some of the flexibility that humans have. Although deep learning systems can often learn a new task from few examples, most demonstrations of this rapid learning have involved sampling tasks that are within a dense region of the training task distribution.[9] Furthermore, humans have a great deal more flexibility than simply learning rapidly. For example, we can follow instructions to accomplish a novel task.

We can also adapt to a novel task without any data at all, if we know how it relates to prior tasks. For example, if we are told to try to lose at poker when we have previously been trying to win, it is easy for us to adapt, despite the fact that the new goal contradicts all our prior experience. With poker by contrast, it would be quite difficult for any contemporary reinforcement learning model to invert its value function. We can also achieve goals that are orthogonal to the original value function, for example trying to follow the motions of a meaningless background sprite. We can do these tasks reasonably well on our first try. That is, we can flexibly adapt our task representations in order to perform a new task zero-shot, based on its relationship to prior tasks. The meta-learning systems I have reviewed do not yet possess this flexibility. Even the emergent flexibility demonstrated by Radford et al. (2019) required them to condition the system on examples of translation — it was not zero-shot.

Some of the zero-shot work I reviewed above does show the ability to perform new tasks without data, but if those models use the systematic relationships between the new tasks and the old, they do so only implicitly. That is, a system like that used in Hill et al. (2020) may generalize to putting a red vase on the bed because the representations that emerge in its language system capture something about the color red and vases, and that allows it to generalize. However, it does not actively use its representations of prior tasks involving vases or red objects. It instead generates a representation for the new task completely from scratch. This approach seems fundamentally

[9]See Chollet (2019) for a detailed discussion of this limitation, and other related points.

different from the approach humans would take when trying to switch to losing at poker – rather than building our understanding of "losing poker" from scratch, using only implicit knowledge of poker, humans actively use our knowledge of poker, but use it in a systematically transformed way. I think this ability will be important for building more flexible deep-learning models.

Thus, my main goal in this dissertation will be to propose a computational model that adapts to a new task by transforming a prior task representation, and to explore whether such a model provides a better model for adaptation than constructing a task representation from language alone. The model involves a system which learns to represent both data and tasks themselves in a shared latent space. It then learns to infer transformations of this space, which can be used both for performing basic tasks, but more critically for transforming task representations themselves. Transforming a prior task representation can allow the model to adapt to novel tasks zero-shot. By learning basic and more abstract transformations in a shared space, the model parsimoniously explains the human ability to adapt task representations, without needing to posit new systems for each new type of transformation.[10]

I will describe this model in detail in Chapter 2. The model does not require building in task-specific knowledge or symbols, and is therefore extremely general, so I will demonstrate its effectiveness across a broad range of domains in the subsequent chapters. I will compare it to human zero-shot adaptation in simple card games in Chapter 3. I will also compare to the alternative approach of constructing a representation for the new task from language alone, without explicitly transforming prior tasks representations. I will then extend my approach to more complex tasks in Chapter 4, including recognition of visual concepts and reinforcement learning. In Chapter 5, I will illustrate one reason why zero-shot adaptation is important – it allows us to learn better once we begin the task, and make many fewer errors along the way to mastery. Finally, in Chapter 6, I will return to the cognitive issues raised in this chapter, and discuss the broader implications of this dissertation.

[10]This observation is related to the general point that humans have the ability to flexibly reason across levels of abstraction. We can relate between examples of a concept and what those examples imply about the overall concept, e.g. as counter-examples to universal properties. With training, we can even reason flexibly across complex hierarchies of abstraction, as when thinking about the mathematical concepts of numbers, sets, functions, and categories. We are able to recursively build abstractions on top of abstractions (although this is often a slow process). By contrast, deep learning models typically represent different levels of abstraction separately, e.g. at separate layers of a feed-forward architecture. This separation limits the flexibility of reasoning between levels of abstraction (the only available mapping is the canonical transformation given by the weights), and because the abstraction is built in to the architecture, it cannot be applied recursively (its depth is fixed). Furthermore, humans can reason both about data and the computations we perform over data, whereas most deep learning architectures restrict their knowledge of computations to weights to which they have no explicit access. This limits the flexibility and representational capacity of these networks. Addressing some of these limitations may be important for achieving more human-like intelligence from deep learning (c.f. Chollet, 2019).

Chapter 2

Meta-mapping

"In the human, internal representations become objects of cognitive manipulation."

Annette Karmiloff-Smith,
Beyond Modularity

As noted in the introduction, humans have the ability to transform our behavior on a task, in accordance with a change in goals. For example, if we are told to try to lose at poker, we can perform quite well on our first try, even if we have always tried to win previously. If we are shown an object, and are told to find the same object in a new color or texture, we can do so. How can we adapt our behavior so drastically, without any data on the new task, even when our new goal contradicts all our prior experience? I suggest that we can do so by exploiting the relationship between the adapted version of the task and the original. In this chapter, I propose a computational model of this adaptation, and demonstrate its success across a variety of domains. The model is both useful for understanding the flexibility of human cognition, and for designing artificial intelligence systems with more human-like flexibility.

The model incorporates several key insights into human cognition. First, when performing a task (such as playing poker), humans are aware of what we are doing and why. I propose that this awareness is mediated by an internal representation of the task. The model I propose therefore performs tasks from a task representation. I take inspiration from various approaches from the machine learning and cognitive science literature, and construct task representations from examples via meta-learning (e.g. Vinyals et al., 2016; Santoro et al., 2016; Finn et al., 2017a, 2018; Stadie et al., 2018; Botvinick et al., 2019), or from a natural language instruction (Larochelle et al., 2008;

Some of the material in this chapter originally appeared as a workshop paper in the Learning Transferable Skills Workshop at NeurIPS 2019.

Hermann et al., 2017; Hill et al., 2020). The model then uses this task representation to respond in a task-appropriate way to its inputs. (It also uses the representations for other purposes, such as identifying features of the tasks.)

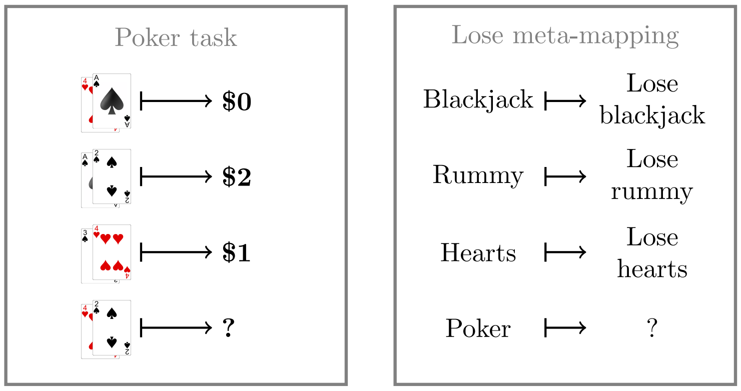
To implement the task computations, we allow a great deal of input processing (perception) to be shared across different tasks. If a human is playing cards, much of the perception of the cards will be identical whether the game is poker or blackjack or bridge, and the task-specific computations will be performed over abstract features such as suit and rank relationships. We thus allow the system to learn a general basis of perceptual features over all tasks with a domain. The system then uses these features in a task-specific way in order to perform task-appropriate behavior. Specifically, the model uses its representation of the current task to parameterize this computation over the perceptual features, and then decodes the result through an output system which is also shared across the tasks.

We also allow the model to transform its representations of tasks, to accomodate task alterations. We refer to these transformations of tasks as **meta-mappings**. Meta-mappings allow the model to adapt to a new task *zero-shot* (i.e. without requiring any data from that new task), based on the relationship between the new task and prior tasks. Meta-mappings can be cued either by examples of the meta-mapping applied to other tasks, or by an instruction, just as basic tasks can be inferred from examples or instructions.

Concretely, our model is able to lose at poker on its first try. To do so, it constructs a representation of poker from experience with trying to win the game. It then infers a "try-to-lose" meta-mapping, either from language, or examples of winning and losing at other games, such as blackjack. It then applies this meta-mapping to transform its representation of poker, thereby yielding a representation for losing at poker. This adapted task representation can then be used to perform the task of trying to lose at poker zero-shot.

The class of models we propose uses the same architectural components both to perform basic tasks and meta-mappings. This is in keeping with the idea that we, as humans, have a single mind that implements computations of all types. The approach is also inspired by the computational notion of *homoiconicity*. A homoiconic programming language is one in which programs can be manipulated in the language just as data can. Our task representations are like programs that perform tasks, and our implementation is therefore homoiconic in the sense that it can operate both on tasks and data. We refer to architectures with the proposed characteristics as homoiconic meta-mapping (HoMM) architectures.

The HoMM approach is parsimonious, in that it does not require adding new networks for each new type of computation. Furthermore, in many cases, functions have some common structure with the entities they act over. For example, the set of linear maps over a vector-space is itself a (higher-dimensional) vector space. If the different levels of abstraction share structural features, sharing computation can improve generalization. Homoiconic models could also support the ability to build



(a) A basic task. (b) A meta-mapping.

Figure 2.1: Basic tasks and meta-mappings. (a) Basic tasks can be seen as mappings from inputs to outputs, for example, from poker hands to bets. (b) Meta-mappings are higher-order tasks, which take a basic task as input, and output a transformed version of that task, for example, switching from winning to losing a game.

abstractions recursively on top of prior abstractions (Wilensky, 1991; Hazzan, 1999; Lampinen and McClelland, 2018), without requiring new computational machinery at each level. We argue that such an ability is an important characteristic of human intelligence, and one we should strive to capture if we hope to build truly intelligent machines. We propose that these architectures will help to bring the adaptability of artificial intelligent systems closer, at least in some ways, to that exhibited by humans.

The main contributions of this chapter are to propose meta-mapping as a computational framework for understanding zero-shot adaptation to new tasks, and to propose a parsimonious implementation of this framework in the form of homoiconic meta-mapping. In this chapter we will demonstrate the success of our approach in a simple proof of concept domain, and explore some features of its performance. In subsequent chapters, we will compare to human adaptation and show the success of our approach across a variety of tasks, ranging from visual classification to reinforcement learning. See below for a discussion of related work. To my knowledge, this dissertation is the first work that proposes transforming a task representation in order to adapt to task alterations zero-shot.

2.1 Task transformation via meta-mappings

Basic tasks are input-output mappings (Fig. 2.1a): We take as a starting point the construal of basic tasks as mappings from inputs to outputs. For example, poker can be seen as a mapping from hands to bets, chess as a mapping of board positions to moves, and object recognition as a mapping from images to labels. This perspective is common across machine learning approaches, which generally try to infer a task mapping from many examples, or meta-learn how to infer it from fewer examples. In our work, we infer these mappings either from examples, from natural language instructions, or by transforming a prior task mapping.

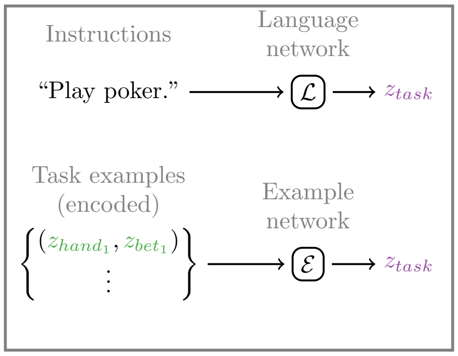
Tasks can be transformed via meta-mappings (Fig. 2.1b): We propose meta-mappings as a computational approach to the problem of transforming a basic task mapping. A meta-mapping is a higher-order task, which takes a task representation as input, and outputs a representation of the transformed task. For example, we might have a "lose" meta-mapping. If given poker as an input, the lose meta-mapping would output a losing variation of poker. If given a representation for blackjack, it would output a losing variation of blackjack. If we have such a meta-mapping, we can use it to transform a task representation in order to perform the transformed version of the task. This transformation allows a model to adapt to a transformed task without having any data on it, just as humans are able to easily switch to trying to lose at a game they have only tried to win in the past.

How can a meta-mapping be performed? There is an analogy between meta-mappings and basic task mappings – they are both simply functions from inputs to outputs. Thus to perform a meta-mapping we use approaches analogous to those we use to perform basic tasks. In particular, we infer a meta-mapping from examples (e.g. winning and losing at a set of example games), or natural language (e.g. "try to switch to losing"). We can then apply this meta-mapping to other basic tasks, in order to infer losing variations of those tasks. Importantly, the system is able to generalize to new meta-mappings, just as it can generalize to new basic tasks. For example, if it experienced meta-mappings which altered the rank of some cards (e.g. "ace is high rather than low"), it could potentially generalize to switching the rank of other cards, either from examples or an instruction.

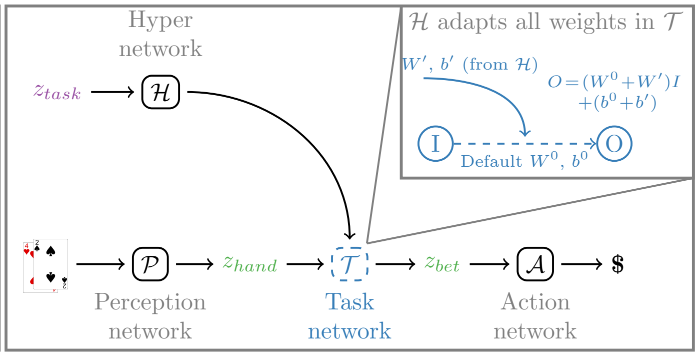
2.2 Homoiconic meta-mapping architectures

We propose homiconic meta-mapping (HoMM) architectures as an implementation of a system that can perform tasks, and adapt to task alterations via meta-mappings. In this section, we describe the basic details of these architectures and their training. See Appendix A for full architecture specifications and hyperparameters (Supp. Table A.1), and training details, including a depiction of inference and gradient flow through the model (Supp. Fig. A.1)

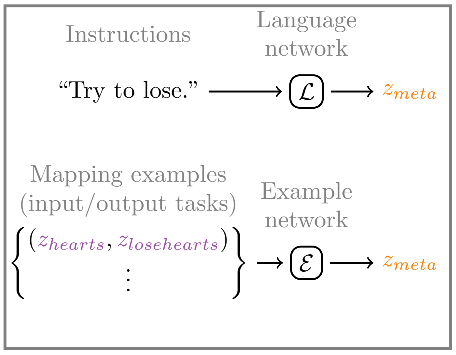
Constructing a task representation (Fig. 2.2a): When humans perform a task, we need to know what the task is. In our model, we specify the task using a task representation. Just like humans, our model supports several different ways of cueing a task, such as instructions (natural language strings), examples of appropriate behavior (e.g. (input, target) tuples, or (state, action, reward) tuples for RL), or by transforming a representation for a known prior task (meta-mapping). To construct a task representation from language, we process the language through a deep recurrent network (LSTM). This approach is similar to techniques used in other work (Hermann et al., 2017;



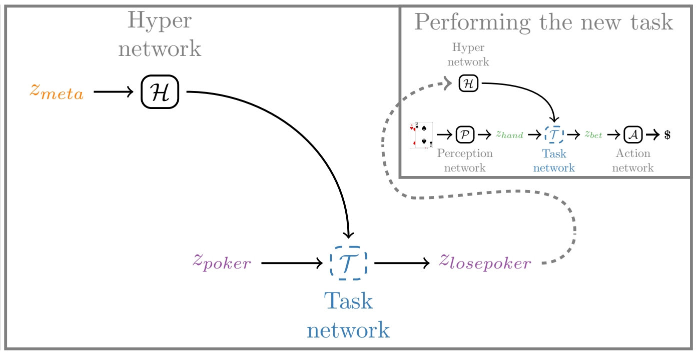
(a) Constructing a basic-task representation.



(b) Performing a basic task from its representation.



(c) Constructing a meta-mapping representation.



(d) Transforming a task via a meta-mapping.

Figure 2.2: Performing and transforming tasks with the HoMM architecture. (a,c) The HoMM architecture performs basic tasks and meta-mappings from a task representation, which can be constructed from appropriate language inputs or examples. (b) The task representation is used to alter the parameters of a task network (see detail) which executes the appropriate task mapping. (d) The meta-mapping representation is used to parameterize the task network to transform a task representation. The transformed representation can then be used to perform the new task zero-shot (see detail). The HoMM architecture exploits a deep analogy between basic tasks and meta-mappings — both can be seen as mappings of inputs to outputs, although they have different types of inputs and outputs. Thus, the architecture uses type-specific models to embed all basic inputs, as well as tasks and meta-mappings, in a shared representational space. Then all tasks and meta-mappings can be seen as transformations applied to entities in this space, which can be executed by shared systems. The parallels between the basic tasks and meta-mappings are reflected in the parallels between the top and bottom rows of the figure.

Oh et al., 2017; Hill et al., 2020, e.g.). To construct a task representation from examples, we process the examples individually to construct appropriate representations for each example, and then aggregate across them by taking an element-wise max. (The element-wise max provides a nonlinear and dataset-order-invariant way of combining examples (c.f. Zaheer et al., 2017) — we found other methods, such as averaging, performed similarly.) This aggregated representation then receives further processing to produce the task representation. This approach shares some common elements with other approaches used for meta-learning (Garnelo et al., 2018).

Performing a task from its representation (Fig. 2.2b): Once we have a task representation, we need to use it to perform the task. We allow a large part of the input processing (perception) and output processing (action) to be shared across the tasks, so that the task-specific computations can be relatively simple and abstract. This idea is related to the long-standing notion that deep networks (both artificial and biological) will construct more disentangled representations of the task relevant features in deeper layers (Dicarlo and Cox, 2007; Erhan et al., 2010), and is used in a number of meta-learning approaches (Vinyals et al., 2016, e.g.). We use a HyperNetwork (Ha et al., 2016) which takes as input the task representation, and adapts the parameters of a task network. Specifically, the HyperNetwork adapts the values of learned "default" connection weights, to make the network task-sensitive. The adapted network then transforms the perceptual features into task-appropriate output features, which can then be decoded to outputs via the shared output processing network. The whole model (including the construction of the task representations) can be trained end-to-end, just as a standard meta-learning system would be. (There are other possible architectures; we show that our approach outperforms the simple alternative of concatenating a task representation to an input embedding before passing it through a fixed network in Supplemental Figures B.2 and D.2. See also Supplemental Figure C.4, for a similar comparison for our language generalization baseline, described below.)

Transforming task representations via meta-mappings (Fig. 2.2c-d): Above, we defined a meta-mapping to be a higher-order task, which takes as input a task representation, and outputs a transformed task representation. Because our model constructs task representations to perform the tasks, all that we need to implement is a way of transforming these representations to perform a meta-mapping. To do so, we exploit the functional analogy between basic-tasks and meta-mappings, noted above. We can infer a representation for a meta-mapping from examples of the meta-mapping, or from a language description, just like we infer a basic task representation from examples or language. We then use this meta-mapping representation to adapt the parameters of the task network to the meta-mapping, and we can then use the network to transform other task representations. This approach is analogous to how we used a basic-task representation to adapt the network to that task, and then used that network to perform the task. (In Appendix B.2.1, we prove that a simpler approach of using vector analogies for meta-mapping is inadequate.)

We exploit this analogy by using exactly the same networks for inferring a meta-mapping representation from examples or language as we do for inferring a task representation. We use exactly the same hyper network to parameterize a meta-mapping-specific network as we use to parameterize the task-specific network. That is, we use precisely the same architecture (and default weights that are modulated by the same hyper network) for both basic task computations and meta-mappings. To make this possible, the shared perceptual processing embeds individual data points into a representation space that is shared with that used for task representations and meta-mapping representations. This approach allows all task- or meta-mapping-specific computations to be seen as operations on objects in the same space, and to be inferred identically, regardless of the objects being transformed. This is parsimonious, homoiconic, and reflects aspects of the Global Workspace Theory of consciousness (Baars, 2005).

Classifying task representations: We can also train the HoMM model to classify task representations. We refer to these task-classifications as *meta-classifications*. Performing meta-classifications could encourage the model to represent the relevant features of the tasks. However, in practice meta-classification does not seem to be particularly important to the performance of the architecture, see Supp. Figs. B.3 and C.5.

Training & evaluating the model: To train the system to perform the basic tasks, we can compute a task-appropriate loss at the output of the action network, and then minimize this loss with respect to the parameters in all networks. This includes the networks used to construct the task representation, and even the representations of the examples or language that they receive as input. That is, we train the sytem end-to-end to perform the basic tasks. When constructing a task representation from examples, we do not allow the example network to see every item, in order to force the system to generalize, in a standard meta-learning fashion. For example, if the basic task is poker, the system will have to construct a task representation from some hands that will be useful for playing other hands. This approach ensures that the task representations capture the structure of the task, rather than just memorizing the provided examples.

To train the system to perform meta-mappings, we try to match the output task representations to those constructed when actually performing those tasks. Specifically, we minimize an ℓ_2 loss on the difference between the output embedding and the embedding constructed when performing the target task. For example, if the system has been trained to play winning and losing variations of blackjack, we would take the task representation for winning blackjack as input, and try to match the output to the task representation for losing blackjack. Again, when we construct a meta-mapping representation from examples of the mapping, we force it to generalize to other examples. Regardless of how the meta-mapping representation is constructed, we can then test this generalization by passing in the representation for a task like poker, that has never been used for any training on this meta-mapping (either as an example or for generalization). We take the output task representation produced by the meta-mapping, and actually perform the task of losing poker with it. This is how we perform all evaluation of the meta-mapping approach in this paper.

In meta-mapping, generalization is possible at different levels of abstraction. The paragraph above refers to basic generalization — applying a meta-mapping seen during training to a basic task that meta-mapping has not been applied to during training, in order to perform a held-out basic task zero-shot. However, if the system has experienced sufficiently many meta-mappings during training, we can also test its ability to generalize to held-out meta-mappings. For example, if the system has been trained to switch various pairs of colors in a classification task (red for blue, green for yellow, etc.), it should be able to generalize to switching held-out pairs (red for yellow, green for blue, etc.) from an appropriate cue (examples or instructions). We view this generalization as an important part of intelligent adaptability — the system should not only able to adapt tasks via meta-mappings that it understands well, but also to infer and use new meta-mappings based on specific instructions or examples. We will demonstrate this ability in the subset of our experimental domains where we can instantiate sufficiently many meta-mappings.

Comparing to language-based generalization: Natural language instructions are an important part of how humans are able to generalize to a new task, and prior work on zero-shot performance has often assumed a description of the task as input (Larochelle et al., 2008, e.g.). For example, a system that has learned to behave in accordance with instructions like "win at poker," "win at blackjack," and "lose at blackjack," should be able to generalize to "lose at blackjack," given sufficiently many training tasks. This approach also does not require data on the novel task. However, transforming the task representation via a meta-mapping may be a more useful inductive bias that allows the system to transform the prior task representation in a targeted way. We thus compare our meta-mapping approach to an approach that simply constructs task representations from language. We show in subsequent chapters that meta-mapping results in better performance on the new tasks, especially when the held-out tasks are very different from trained ones (e.g. directly contradicting). That is, in our experiments, meta-mapping generally has better sample-complexity in terms of the number of prior tasks it must have experienced to perform well, especially when the space of tasks is sparsely sampled, or generalization is challenging. This efficiency is crucial, because humans have not seen 95% of the possible task space when they need to generalize to a new setting.

2.3 Experiments

Meta-mapping is an extremely general framework. Because the assumptions are simply that the basic tasks are mappings from inputs to outputs, and that meta-mappings transform basic tasks, the approach can be applied to most paradigms of machine-learning with only minor modifications. We demonstrate the success of meta-mapping in four settings over the next few chapters, ranging from regression to classification to reinforcement learning. We summarize the contributions of the different experiments in Table 2.1. In this chapter, we explore the performance of meta-mapping in

Chapter	**Experiment**	**Motivation**	**Held-out MMs**	**Lang. Comp.**	**Paradigm**	**Input**
2	Polynomials	Proof of concept	✓		Regression	Vector ($\mathbb{R}^4$)
3	Card games	Comparison to humans		✓	Regression	Several-hot vector
4.1	Reinforcement learning	Cognitive and AI relevance		✓	RL	91×91 RGB image
4.2	Visual concepts	Cognitive and AI relevance	✓	✓	Classification	50×50 RGB image

Table 2.1: The contributions of our four sets of meta-mapping experiments. Our results span various computational paradigms and various degrees of input complexity, and are motivated by both cognitive and AI relevance.

detail in a proof-of-concept polynomial domain. We also describe some interesting behavior of the model that may intrigue researchers in cognitive control (but is not a focus of the remainder of the dissertation).

2.3.1 Polynomial regression

As a proof of concept, we first demonstrate the success of our approach in polynomial regression (see Fig. 2.3). Specifically, we construct basic tasks that consist of regressing polynomial functions (of degree ≤ 2) in four variables (i.e. from $\mathbb{R}^4 \to \mathbb{R}$). We sampled these polynomials by first uniformly sampling a subset of variables to be included, and then sampling coefficients from $\mathcal{N}(0, 2.5)$ for the possible monomials. These basic tasks can be inferred from (input, output) tuples, where the input is a point in $\mathbb{R}^4$ and the output is the evaluation of that polynomial at that point. (We actually restrict the input range to $[-1, 1]$ to avoid testing extreme outlier points.) This is a simple meta-learning regression problem, which the system performs well (Supp. Fig. B.1).

These tasks/polynomials can then be transformed by various meta-mappings — we considered squaring a polynomial, permuting its variables, or adding or multiplying by a constant. We trained the model on 100 basic polynomials, and we train mapped versions of 60 of these for each meta-mapping. We can evaluate the performance of that meta-mapping on the remaining 40 target tasks (corresponding to the 40 other basic polynomials) that the model has never experienced before. We also held out some of these meta-mappings to evaluate the ability of our method to generalize at the meta-mapping level (see above). For example, we can train the model to adapt to a subset of the input variable permutations, and then evaluate its ability to adapt to a held-out permutation

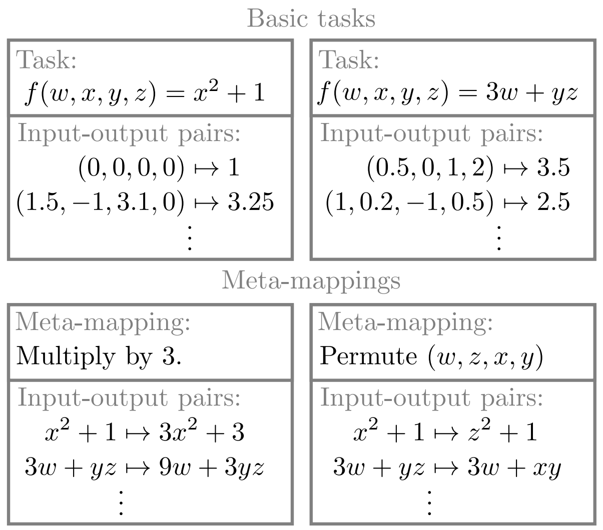


Figure 2.3: The polynomial task domain. A basic polynomial task consists of regressing a single polynomial, i.e. the inputs are points in $\mathbb{R}^4$ and the outputs are the value of the polynomial at that point. These basic regression tasks can be transformed by various meta-mappings, such as multiplying by a constant, or permuting their variables.

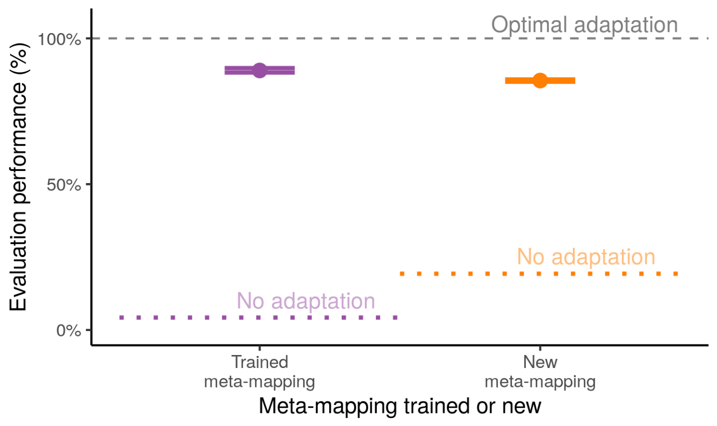


Figure 2.4: Meta-mapping results in the polynomials domain. We plot zero-shot performance (normalized, see text) on new tasks via meta-mappings. The system not only-generalizes trained meta-mappings to examples it has never seen before (purple), but also generalizes to held-out meta-mappings from examples (orange), and does both substantially better than a baseline model which does not adapt (dotted lines). Thus our approach is able to flexibly adapt to a new polynomial without any data from that polynomial, based on that polynomial's relationship to polynomials it has experienced.

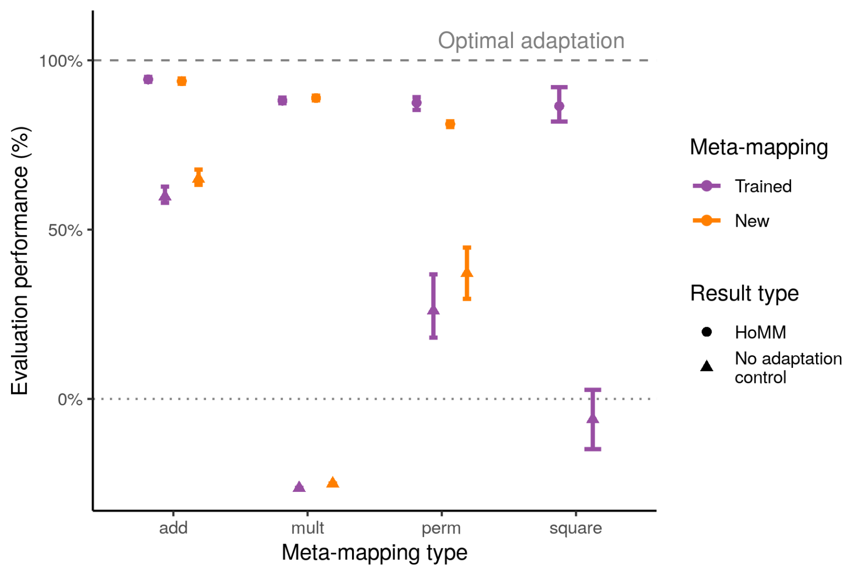


Figure 2.5: Meta-mapping results in the polynomials domain, broken down by meta-mapping type. We plot a normalized performance measure, as in Fig. 2.4. The system is performing well across all meta-mapping types, although there is some variability. Triangles show performance of a baseline model that does not adapt — note that this baseline performs decently on some meta-mappings, while in other cases such a model results in worse performance than outputting all zeros.

based on examples of that permutation. In total, we trained on 20 meta-mappings, and held-out 16, corresponding to 2260 ($= 100 + 60 \times 36$) trained basic tasks, and 1440 held-out for evaluation.

Training: We trained the system in epochs, during which it received one training step on each trained basic task and one training step on each trained meta-mapping, interleaved in a random order. For one step of training on a basic task, we used 1024 evaluations of the polynomial — we present the model with 50 example evaluations from which to generate a task representation, and make one gradient update that improves the predictions on the remaining evaluations (as well as the example ones). This approach encourages the model to generate an accurate representation of the polynomial from seeing a (relatively) small set of evaluations.

For one step of meta-mapping training, we take task representations for each of the 60 basic tasks (and corresponding target tasks) on which the meta-mapping is trained, where each basic task representation is computed from 50 examples as above. We randomly chose half of these (input task, output task) pairs to provide as examples of the meta-mapping from which to generate a representation, and train the system to match the output embeddings from the meta-mapping to the targets for all 60 examples. This approach encourages the model to generate a representation of the meta-mapping from half the available examples that will generalize to the other half.

To evaluate the system on a trained meta-mapping, we parameterize the mapping using all 60 input-output function embedding pairs that were used to train the meta-mapping, and evaluate the performance resulting from applying that mapping to the other 40 basic tasks to perform their 40 corresponding held-out target tasks. The system never experienced those 40 target tasks during training. Similarly, to evaluate on a held-out meta-mapping, we use 60 (input task, output task) pairs where both the input and target basic tasks were trained, and evaluate on 40 trained input tasks for which the corresponding 40 targets have not been trained. However, in the case of a held-out meta-mapping, the meta-mapping itself is never encountered during training. This allows us to evaluate whether the system is able to infer a new meta-mapping based on basic tasks that it has experienced, mapped in a way it has not experienced.

Furthermore, when evaluating a meta-mapping (either trained or held-out), we do not simply evaluate how closely the output embeddings match the targets. Instead, we use each of those output embeddings to perform the appropriate task — i.e. we use a dataset of 1024 polynomial evaluations to compute the MSE between the predictions produced by the model with the mapped embedding, and the true target polynomial. See the Supplemental Information for futher details on training and evaluation.

Results

In Fig. 2.4, we show the success of our meta-mapping approach in this setting. We plot a normalized performance measure, which is 0% if the system outputs all zeros for every polynomial, and 100% if the system performs perfectly. Specifically, we measure performance as $1 - \text{loss}/c$, where c is the loss for a baseline model that always outputs zero.[1] We also show performance of a baseline model which just performs the original tasks without adaptation (dotted lines). Our HoMM approach is able to achieve 89.0% performance (bootstrap 95%-CI across runs [88.3, 89.8]) on a polynomial it has never experienced during training, based on a trained meta-mapping, and 85.5% performance (bootstrap 95% CI [85.1, 85.9]) based on a held-out meta-mapping. By comparison, not adapting would yield only 4.3% and 19.3% performance, respectively. That is, our system is able to achieve good performance on a new task without any data, based only on its relationship to prior tasks. It is able to do so much better than a baseline model which does not adapt to the new tasks.

To further explore this performance, in Fig. 2.5 we plot the results for each of the different meta-mappings we considered: adding a constant, multiplying by a constant, permuting the variables, and squaring the polynomial. Some of these mappings are more challenging than others, as can be seen from the performance of a baseline model that does not adapt (dotted lines). For example, adding a constant to a polynomial does not alter it too drastically, so the non-adapting baseline performs well there. By contrast, multiplying by a constant sometimes changes the sign of the polynomial, so the non-adapting baseline performs extremely poorly there. The HoMM approach results in good performance across all the types of meta-mappings we considered, although unsurpsingly performance

[1]This measure is closely related to the variance explained, except that the square meta-mapping skews the mean of the output polynomials slightly away from zero.

is slightly better on the easier tasks, and generalization to the held-out permutation meta-mappings is more challenging than generalization to e.g. the held-out addition ones.

Representation analysis: In order to explore the performance of the model further, we performed principal components analysis on the task and meta-mapping representations in the HoMM model after training (Fig. 2.6). This analysis reveals strikingly similar organization of the representation space across different training runs, with constant polynomials pushed to the outside in a semi-circle, and more complex polynomials stretching toward the center, where meta-mappings and meta-classifications are located. This organization may be due to the learning dynamics — the distance of the task representations from the center appears to be roughly inversely proportional to the complexity of the task, which might imply that the constant polynomials have the largest-magnitude representations because they are easiest to learn.

To analyze this further, in Fig. 2.7 we plot the representations for only the constant polynomials, colored by their value (square-root compressed for clarity). This analysis shows that the constant polynomials are consistently arrayed angularly from lowest to highest value.

Finally, we examined the meta-mapping representations more closely (Fig. 2.8). This analysis shows that the mappings have a consistent organization across runs, with permutations and addition grouping tightly, but multiplication and squaring, which more drastically alter the polynomials, more dispersed. In particular, multiplying by negative numbers and squaring, which can change polynomials signs and therefore cause a more drastic adaptation, are more separated from the remaining meta-mappings. It is also interesting to note that the addition meta-mappings appear to be organized more by absolute value than sign in at least some runs. However, in higher principal components (not shown), the mappings appear to be organized more linearly by value.

A non-homoiconic comparison: In Fig. 2.9, we compare HoMM to a nonhomoiconic architecture – i.e. one in which there are separate example networks ($\mathcal{E}_{base}, \mathcal{E}_{meta}$) and hyper networks ($\mathcal{H}_{base}, \mathcal{H}_{meta}$) for the base tasks and meta-mappings. The nonhomoiconic approach performs substantially worse. Specifically, on trained meta-mappings the HoMM model is achieving a normalized performance of 88.99% (bootstrap 95%-CI [88.20, 89.98]), while the non-homoiconic achieving a normalized performance of 83.2% (bootstrap 95%-CI [81.9, 84.9]). On new meta-mappings the HoMM model is achieving a normalized performance of 85.54% (bootstrap 95%-CI [85.14, 85.94]), while the non-homoiconic model is achieving a normalized performance of 81.3% (bootstrap 95%-CI [80.3, 82.2]). These results highlight the value that homoiconic architectures can have when meta-mappings share structure with the basic tasks. (See also Supp. Fig. C.6 for this architectural comparison in the cards domain — the direction of the effect is the same, but the difference there is not significant.)

2.3.2 Aside: HoMM & cognitive control

Although it is not the primary focus of this project, the HoMM architecture could be of interest to researchers in cognitive control, even beyond the idea of meta-mapping as adaptation. The

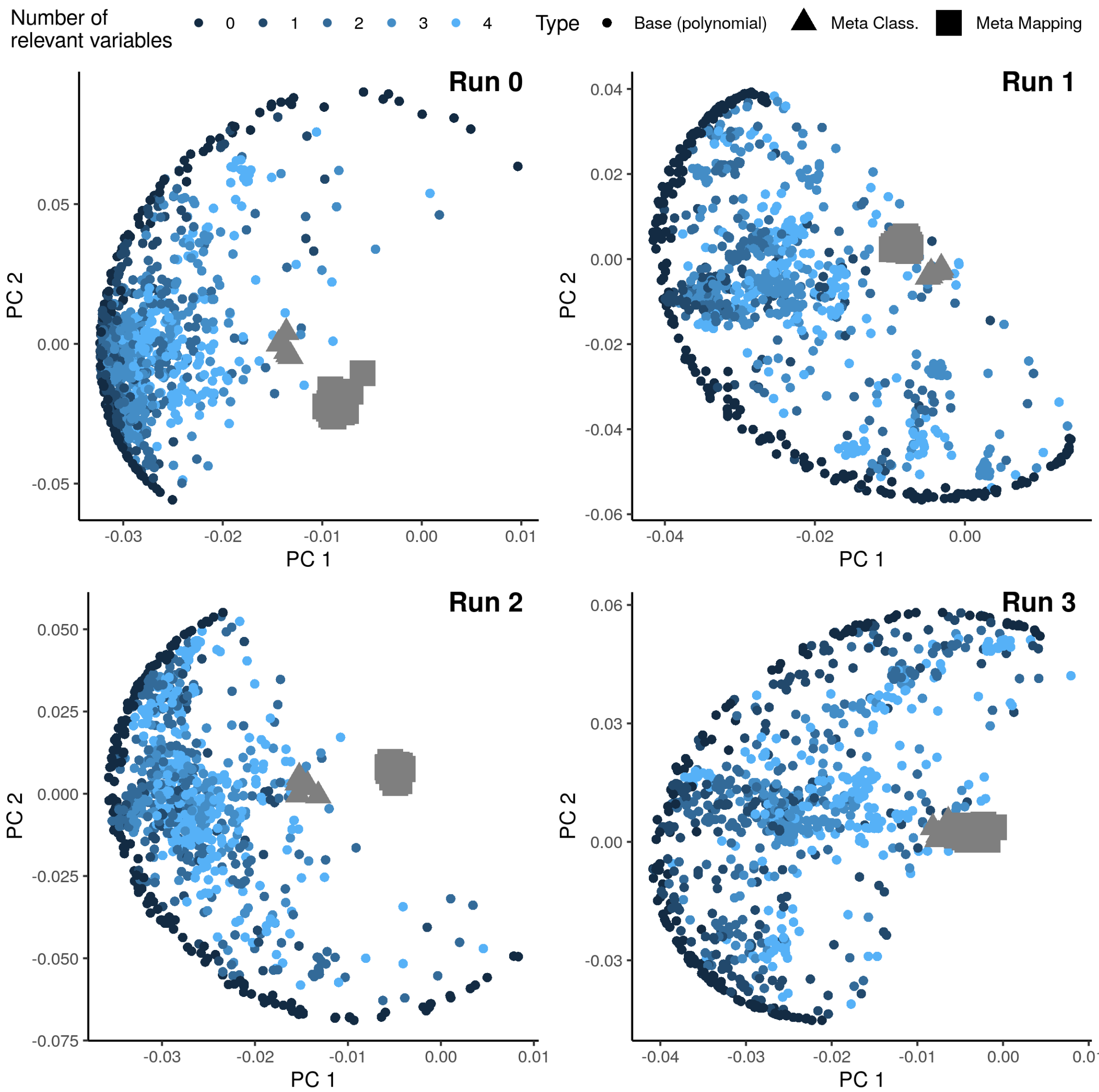


Figure 2.6: Principal components of task and meta-mapping representations of HoMM after training on the polynomials domain. The representation space is organized relatively consistently across runs, with constant polynomials pushed to the outside, and meta-mappings and meta-classifications more centrally located.

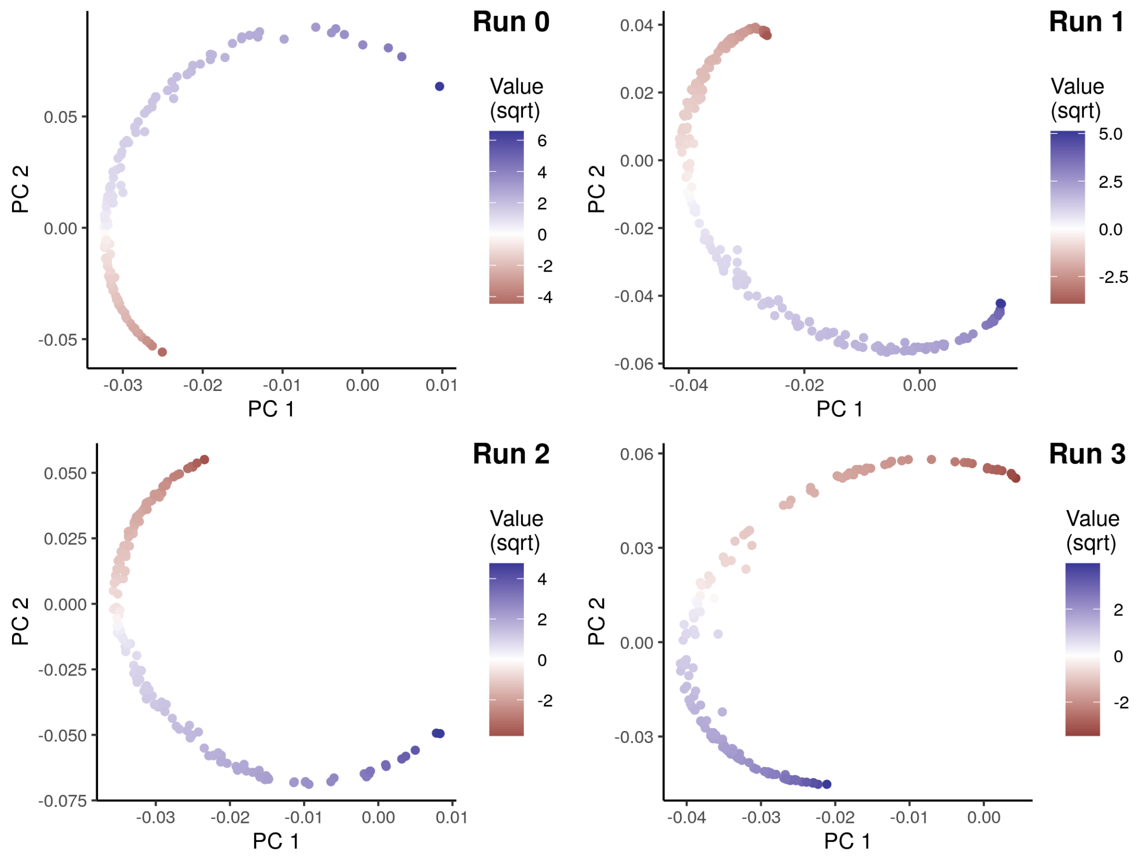


Figure 2.7: Principal components of constant polynomial representations, showing systematic organization by value. Intriguingingly, this relationship appears to be systematically non-linear across runs. (PCs computed across all task representations, color scale of values is compressed with a square-root transformation.)

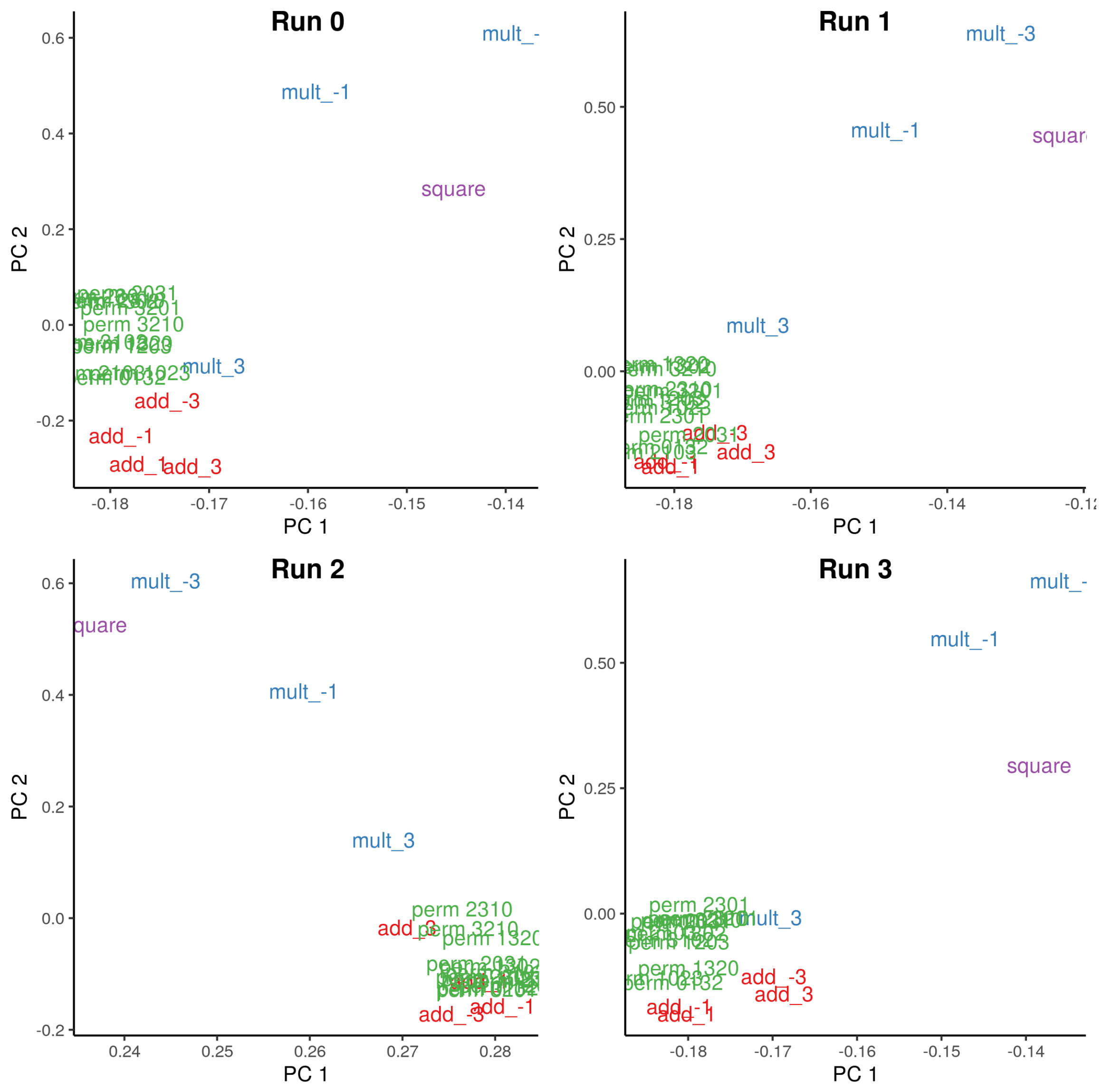


Figure 2.8: Principal components of meta-mapping representations in the polynomial domain, showing systematic organization by type. Permutation mappings cluster tightly, as do addition, while multiplication and squaring are more dispersed. The addition and multiplication mappings are partially organized by absolute value.

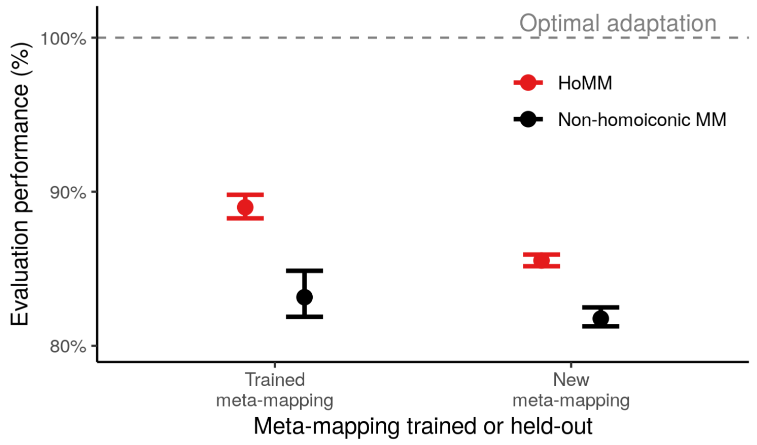


Figure 2.9: HoMM outperforms a non-homoiconic baseline in the polynomials domain. This figure compares the meta-mapping performance of HoMM with a nonhomoiconic model that instantiates separate copies of the example network ($\mathcal{E}_{base}, \mathcal{E}_{meta}$) and hyper network ($\mathcal{H}_{base}, \mathcal{H}_{meta}$) for the basic tasks and the meta-mappings. HoMM significantly outperforms the non-homoiconic approach. These results suggest that there is sufficient shared structure between the basic tasks and the meta-mappings for the homoiconic approach to improve generalization, and supports our use of homoiconic architectures.

architecture offers an instantiation of a model which can perform different tasks based on task examples or language inputs, which is fundamentally the same problems human face when we must adapt our behavior. There are a number of features of the model that offer the opportunity for intriguing investigations based on this idea. For example, the task network in our architecture has a default set of bias weights that are modulated by the HyperNetwork. These can be thought of as the "automatic" or "default" processing habits of the system, whereas the weight alterations the HyperNetwork imposes can be thought of as the exertion of cognitive control to modulate behavior.

To explore this idea, we trained our architecture on a very simple stroop task taken from Cohen et al. (1990). The model receives two sets of two inputs, that can be thought of as corresponding to "word" and "color" domains. One input in each domain is turned on, representing a color word written in a color. The model's task is to report either the color or the word, depending on context.

The context we give the model is in the form of examples of the task as (input, output) pairs as usual. These are used to construct a task representation, which is then used to modulate the parameters in the task network, via a HyperNetwork.[2] We trained the model repeatedly with different proportions of training on the word task vs. the color task, in order to investigate the default vs. controlled behavior in different training regimes. Specifically, we compared training

[2]For this experiment, we used similar hyperparameters to the polynomials experiments, except we used a much smaller model — a single-layer task network, a Z-dimensionality of 8, and $\mathcal{H}, \mathcal{M}$ had 64 hidden units per layer. We optimized the model via stochastic gradient descent with a learning rate of 0.01 to follow more closely the approach taken by Cohen et al., although results are similar with other optimizers.

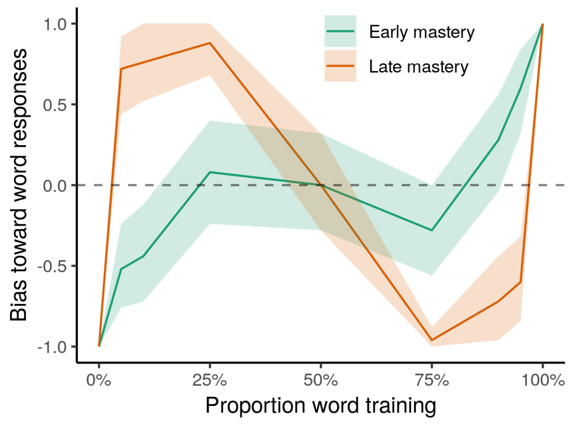


Figure 2.10: Measuring the default behavior of the HoMM architecture on a Stroop-like task. We plot the bias of the model towards word or color responses, when given an all-zeros task representation, at different proportions of training on words or colors, and different stages of training. When the model has just mastered the less frequent task, it exhibits a default bias towards the more frequent task. However, later in training, when it has mastered both tasks, it exhibits a paradoxical bias towards the *less* frequent task.

the model to the point that it barely mastered the less frequent task (when it first achieves 100% performance and cross-entropy loss < 0.3 on both tasks) to the point that it had mastered both tasks (100% performance and cross-entropy loss < 0.01 on both). We then tested the model's default behavior by giving it an all-zeros task representation, and seeing whether its performance was more aligned with the "word" or "color" task.

In Fig. 2.10, we show the results. We plot the bias as $2 \times$ (word accuracy $-$ color accuracy), which is -1 if the model is responding only to color, 1 if the model is responding perfectly to word, and 0 if it is responding equally to each (or otherwise responding randomly). When the model has just barely mastered the less-frequent task, it exhibits a default bias towards the more frequent task. However, once we train it to full master of both tasks, it exhibits a surprising paradoxical bias towards the task that was mastered more recently. This effect may relate to observations that switching from a less-practiced task back to a more practiced one is difficult (Monsell, 2003), possibly because performing the less-practiced task requires strong suppression of the default behavior. It's possible that in the course of achieving full mastery on the less-practiced task, the more practiced task must be so suppressed that it fades away from being the default. These phenomena provide possible inspiration for future investigations in cognitive control.

2.4 Discussion

I have proposed meta-mappings as a computational account of the human ability to perform a novel task zero-shot (without any data), based on the relationship between the novel task and prior tasks. I have shown that a homiconic meta-mapping architecture performs well in a proof-of-concept polynomial regression setting. Here, I will briefly discuss some alternative architectures that could be explored within the meta-mapping framework, as well as some of the work in machine learning and cognitive science that inspired this project. I will discuss the broader implications of our work and future directions in more detail in Chapter 6.

2.4.1 Architectural and algorithmic choices

There are a number of architectural aspects of the approach that could potentially be altered. Exploring these in full seemed beyond the scope of this project. I outline a few that provide promising future directions here; see Chapter 6 for a cognitively-focused discussion.

Firs, although we used HyperNetworks to parameterize our task network, it would also be reasonable to have a fixed task network which simply receives the task representation as an additional input. As noted above, we evaluated this simpler approach in the polynomial and RL domains, and found it did not perform as well at meta-mappings (although it performed similarly at the basic tasks). We also found that the language model generalized similarly with either architecture in the cards domain. However, the simpler architecture might be a useful approach in some settings.

We also noted in the visual categories domain that linear task networks seemed to improve meta-mapping, while nonlinear ones seemed to result in better basic task performance — thus it might be reasonable to consider a deep, nonlinear task network, but with a linear skip-connection from beginning to end. An identity (ResNet-like) inductive bias on this linear connection might be helpful as well, so that the network would only have to learn what to change about the task representation.

Furthermore, although we found that homoiconic architectures were useful, it might be that in some task domains a shared representational space with shared networks across different types of tasks is detrimental. In general, whether sharing an architecture across different tasks is beneficial depends on the data regime — shared architectures can be a useful regularizer with small datasets, but correspondingly harmful with sufficiently large ones. Thus, the answer to this question will likely depend on the depth and breadth of the training data. One cognitively-motivated intermediate option might be to have a domain general shared-system like ours, but with additional learning of more domain-specific mappings directly from inputs to outputs, which could potentially allow for the benefits of both approaches.

Finally, in principle any meta-learning approach which uses task representations or similar abstractions (e.g. Rusu et al., 2019; Zintgraf et al., 2018) could be applied instead of the example-network and hyper-network based approach we used. It would be interesting to explore whether alternative approaches offer benefits, such as greater computational or memory efficiency, or greater hyper-parameter robustness. It would also be interesting to explore whether different approaches would lead to differences in the structure of the model's representations.

2.4.2 Related work in machine learning

As mentioned above, there is a large body of prior work on zero-shot learning based on natural-language descriptions of tasks. Larochelle et al. (2008) considered the general problem of behaving in accordance with language instructions as simply asking a model to adapt its response when conditioned on different "instruction" inputs. Later work explored zero-shot classification based on only a natural language description of the target class (Socher et al., 2013; Romera-Paredes and Torr, 2015; Xian et al., 2018), or of a novel task in a language-conditioned RL system (Hermann et al., 2017; Hill et al., 2020). Some of this work has even exploited relationships between tasks as a learning signal (Oh et al., 2017). Other work has considered how similarity between tasks can be useful for generating representations for a new task (Pal and Balasubramanian, 2019), but without transforming task representations to do so. Furthermore, similarity is less specific than an input-output mapping, since it does not specify *along which dimensions* two tasks are similar. To my knowledge none of the prior work has proposed using meta-mapping-like approaches to adapt to new tasks by transforming task representations, nor has the prior work proposed a parsimonious homoiconic model which can perform these mappings.

My work is also related to the rapidly-growing literature on meta-learning (e.g. Vinyals et al., 2016; Santoro et al., 2016; Finn et al., 2017a, 2018; Stadie et al., 2018; Botvinick et al., 2019; Ravichandran et al., 2019). Our architecture builds directly off of prior work on HyperNetworks (Ha et al., 2016) and other recent applications thereof (e.g. Brock et al., 2018; Zhang et al., 2019; Li et al., 2019; Rusu et al., 2019). In particular, recent work in natural language processing has shown that having contextually generated parameters can allow for zero-shot task performance, assuming that a good representation for the novel task is given (Platanios et al., 2017) – in their work this representation was evident from the factorial structure of translating between many pairs of languages. Our work is also related to the longer history of work on different time-scales of weight adaptation (Hinton and Plaut, 1982; Kumaran et al., 2016) that has more recently been applied to meta-learning contexts (e.g. Ba et al., 2016; Munkhdalai and Yu, 2017; Garnelo et al., 2018) and continual learning (Hu et al., 2019, e.g.). It is more abstractly related to work on learning to propose architectures (e.g. Zoph and Le, 2016; Cao et al., 2019), and to models that learn to select and compose skills to apply to new tasks (e.g. Andreas et al., 2016b,a; Tessler et al., 2016; Reed and de Freitas, 2015; Chang et al., 2019). In particular, some of the work in domains like visual question answering has explicitly explored the idea of building a classifier conditioned on a question (Andreas et al., 2016b, 2017b), which is related to our approach in the visual categorization tasks (Chapter 4.2).

Work in model-based reinforcement learning has also partly addressed how to transfer knowledge between different reward functions (e.g. Laroche and Barlier, 2017); the HoMM approach is more general. For example, rather than needing a new reward function to be given, meta-mapping provides a principled way to infer a new reward estimator by transforming a prior one. Meta-mapping could also be used to transform a transition function used in the planning model in response to environmental changes. Our insights could therefore complement model-based approaches, which provides an exciting direction for future work.

There has also been other recent interest in task (or function) embeddings. Achille et al. (Achille et al., 2019) recently proposed computing embeddings for visual tasks from the Fisher information of the parameters in a model partly tuned on the task. They show that this captures some interesting properties of the tasks, including some types of semantic relationships, and can help identify models that can perform well on a task. Rusu and colleagues recently suggested a similar meta-learning framework where latent codes are computed for a task which can be decoded to a distribution over parameters (Rusu et al., 2019). Other recent work has tried to learn representations for skills (e.g. Eysenbach et al., 2019) or tasks (Hsu et al., 2019, e.g.) for exploration and representation learning, but has not explored transforming these representations to achieve zero-shot performance on a novel task.

In summary, our perspective builds on several lines of prior work in machine learning. While there has been substantial prior work on meta-learning, task representation, and there have even been other approaches to zero-shot task performance, to the best of our knowledge none of the prior work has explored zero-shot performance of a task via meta-mappings. In the following chapters, I will show experimentally that this approach yields better performance than alternative approaches across a variety of domains. I therefore suggest that meta-mapping may complement other approaches to adaptability, such as model-based RL.

2.4.3 Related work in cognitive science

The HoMM model is inspired by several streams of research in cognitive science as well. I will briefly review some of these here, in order to provide some grounding for the rest of the dissertation. However, I will discuss some of these issues in greater detail in Chapter 6, when I reflect on the dissertation as a whole.

A first inspiration is a long line of research has suggested that analogical transfer between structurally isomorphic domains may be a key component of “what makes us smart” (Gentner, 2003). Analogical transfer has been demonstrated across various cognitive domains (e.g. Bourne, 1970; Day and Goldstone, 2011). Yet there has been relatively little exploration of adaptation without any examples of the new task at all.

This dissertation also touches on the issues of compositionality and systematicity. Some researchers have advocated that cognition must rely on strictly compositional representations in order to exhibit systematic and productive generalization (e.g. Fodor, 2001a; Lake and Baroni, 2018). We avoided explicitly enforcing compositional task representations in our model, instead allowing those representations to emerge. This approach has several advantages. First, it requires much less hand-engineering for each application domain (e.g. our model did not need the notion of variables or permutation built into it to generalize to held-out permutation meta-mappings), and second, it may allow for novel decompositions at test time. (See Chapter 6 for further discussion.)

Some aspects of the HoMM architecture may also seem reminiscent of the modularity of mind for which Fodor advocated (Fodor, 1983), particularly the fact that we divided the model into feed-forward input and output systems, with the flexible, task-specific computations in the middle shared across many domains. In fact, I believe that perception and task-processing are mutually constraining and reinforcing, but for simplicity our model does not incorporate all aspects of this. Cognitive modeling always requires some simplification in order to provide a useful description of the system. (Again, see Chapter 6 for further discussion.)

Finally, as discussed above, the HoMM architecture and approach may relate to areas like cognitive control. Similarly, the shared workspace for data points, tasks, and meta-mappings relates to ideas like the Global Workspace Theory of consciousness. Exploring these connections would be an exciting direction for future work.

In summary, meta-mapping and the HoMM architecture draw inspiration from many areas of research in cognitive science. I hope that this work will reciprocally provide inspiration to many researchers in this field. In support of this, the subsequent chapters explore our model on a variety of tasks more relevant to human cognition, and make direct comparisons to the adaptation abiilites of humans and language-conditioned models.

Chapter 3

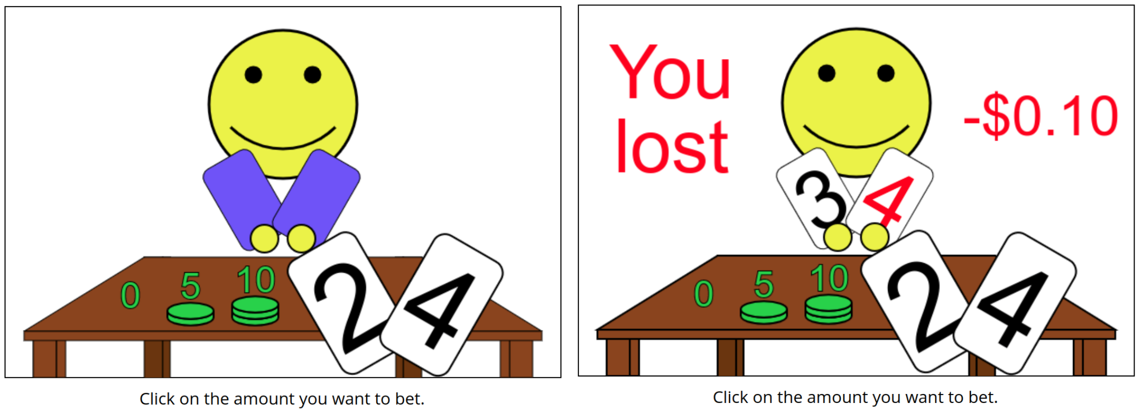
Comparing to human adaptation

In the previous chapter I proposed a framework for modeling human adaptation to new tasks, based on relationships between tasks. To evaluate the quality of the framework, it is necessary to compare its adaptation to human adaptation.

It is worth stopping for a moment to consider how flexible humans actually are. We certainly experience short-term interference from switching tasks or goals (Rogers and Monsell, 1995), or when we try to override a habitual response (Stroop, 1935; MacLeod, 1991). Over longer periods of time, our adaptation might be a response to learning in the new situation, rather than the type of zero-shot flexbiility that HoMM is intended to model. How flexibly are humans able to adapt to task changes in a short amount of time?

Unfortunately a complete evaluation of human flexibility is a large research program. In this chapter we present an evaluation of human adaptation in one setting, inspired by a motivating example used in the previous chapters. We taught participants a simplified poker-like card game, and evaluated how well they were able to switch to losing the game, after practicing trying to win. This is a difficult form of adaptation, which requires completely reversing a value function, and has been highlighted as a challenge for deep learning (Lake et al., 2017).

We compared human performance on the game to both a HoMM model and a task-description-based language-generalization model. We trained both models on variations of 5 card-game tasks, including losing variations of 4 of these, but crucially holding out all losing variations of the task that the human subjects played. We then evaluated the ability of the models to adapt to the losing variation zero-shot, just as we had evaluated the humans. In the HoMM model, this adaptation was based on applying a "switch-to-losing" meta-mapping to the learned variation of the game, whereas in the language-generalization model it was based on a novel instruction to lose the game that was systematically related to the training instructions for losing other games.


0
5
10
2
4
Click on the amount you want to bet.
You lost
-$0.10
3
4
0
5
10
2
4
Click on the amount you want to bet.

(a) Before betting. (b) Feedback.

Figure 3.1: The card game experiment trials, as seen by participants.

3.1 Experimental design

The human experiment was conducted on Amazon Mechanical Turk. We tried to design the game that participants played to make it easy for them to learn, without relying on their prior knowledge of card games. The game was a simplified variation of poker. The participants were dealt hands which consisted of two cards, each with a number (rank) between 1 and 4, and a color (suit) of red or black. The participants played against a computer opponent that was dealt a similar hand. The hands were ranked such that straight flushes (adjacent cards in the same suit) beat adjacent cards in a different suit, which beat non-adjacent cards (including pairs). Ties were broken by the highest card, or by suit if both cards were tied.

On each trial, participants were dealt a hand and asked to make a bet of 0, 5, or 10 cents (see Fig. 3.1). If their hand beat the opponent's hand, they won the bet amount. If their hand lost, they lost it. If the hands were tied, they neither won nor lost money.

The experiment had several phases. First, participants were instructed in the rules and payment scheme for the experiment. Next, they were instructed on the rules of the game. After this, they were tested with four hand-comparison trials intended to probe their understanding of each of the rules of the game. If they failed more than one of these trials, they were not allowed to continue with the experiment.

Following this understanding check, participants played a block of 32 hands (sampled to have a diversity of expected values), where they saw the results of their play (as in Fig. 3.1b). After this block, they played a similar block of 24 trials where they did not see the results of their play. The results were replaced with a brief grayed-out screen, and participants were payed the net expected value of their actions over the block (rounded to the nearest 10). This provides an evaluation phase with relatively less potential for learning.

Finally, participants were told that we wanted them to try to lose for the remaining trials, and that "for the remainder of the experiment, if you bet and lose, you'll gain the amount you bet, and if you bet and win, you'll lose the amount you bet." They were then given an attention check to evaluate whether they had understood this instruction. Subjects who failed this attention check were excluded from the analysis. They then played another block of 24 trials where they were rewarded for losing instead of winning (i.e. the expected returns were reversed). As in the previous block, they did not see the results of their actions. They were finally asked a few demographic questions. See Appendix C for detailed instructions & methods.

Our main target comparison was performance in the two blocks without feedback – were participants able to switch their behavior to lose at the game as well as they won at it?

3.1.1 HoMM model

To compare to the human participants, we wanted to evaluate the HoMM model's ability to switch to losing based on a "try-to-lose" meta-mapping. To do so, we needed other games (with winning and losing variations) to use as training examples of the meta-mapping. We therefore created 4 other card games, based on simplifications of other existing games, like blackjack or matching cards. We created variations of these games that switched whether suit or rank was the most important attribute, and which suit was most valuable. We created losing variations of each of these. In total, there are 5 card games $\times$ 3 binary attributes = 40 basic tasks. See Appendix C.2 for full descriptions of all the games and variations.

We trained the model on 36 of these games, but held out the losing variations of all versions of the game that the human subjects played. We trained the model on three meta-mappings, corresponding to toggling the three binary game attributes. We also trained the model to classify the basic game types (one vs. all) and each of the attributes. We then evaluated the ability of the model to adapt to the losing variation zero-shot, just as the humans did. This adaptation was based on instantiating the "switch-to-losing" meta-mapping with the 32 available examples, and applying it to the 4 other training tasks. We used the three other held-out variations of the game as a validation set to pick an optimal-stopping point for evaluating the model on the remaining hold-out (the game that the humans played).

In order to perform the basic tasks with the HoMM model, we had to make one minor alteration. Because in this task the model (or participant) only receives feedback on the action taken, this is not a simple regression problem with inputs and targets. It is more similar to a reinforcement learning setting, where the model takes actions and may or may not receive rewards in response (although there is no temporal component in our simplified tasks). We thus altered the way a task representation is constructed from examples in the model. Instead of using a dataset of (input, target) tuples, we used a dataset of (input, (action, reward)) tuples, where the action and reward were processed together to form a single embedding.[1] The model was then trained with a masked loss, such that it only updated its predictions for actions it actually took, rather than other possibilities. (See Appendix C.2 for further details of the training.)

3.1.2 Language model

We also compared to a language-generalization baseline. As a reminder, this is an alternative approach to zero-shot task performance, where the model simply receives a natural language description of the task. The task descriptions were sequences of the form: `[‘‘game’’, <game_type>, ‘‘losers’’, <losers-value>, [other attributes]]`, encoded by a 2-layer LSTM network. Similar approaches have yielded good zero-shot generalization in some domains (e.g. Hermann et al., 2017). The language model was trained on the same set of tasks as the HoMM model, and was optimally stopped by using the same validation set. Aside from the construction of the task representation, the remainder of the architecture was identical to HoMM. We also compared to a simpler language architecture, more similar to those used in prior work, with similar results, see Supp. Fig. C.4. (Again, see Appendix C.2 for further architectural details.)

3.2 Human performance

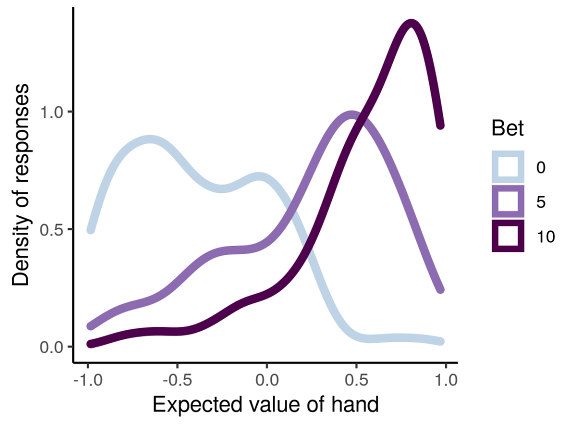
First, how well were participants able to learn the game? Participants' performance is reasonable (mean performance 64% of optimal, bootstrap 95%-CI $[0.57, 0.70]$). However, they are far from optimal, and there is substantial individual variability (Fig. 3.2). In particular, participants are sometimes making intermediate bets, which an optimal agent would never do. Furthermore, the thresholds where each subject crosses a betting probability of 0.5 are variable, some subjects are substantially over- or under-conservative. However, performance is actually more optimal than some basic statistics might suggest, see Appendix C.3.1.

After being asked to lose, participants also performed above chance, but far from optimally (Fig. 3.3). So how well were they able to adapt?

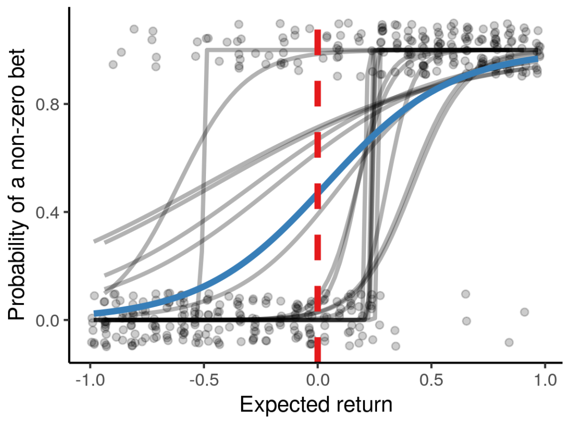
3.3 Adaptation in humans and HoMM

Human adaptation was quite good, in the sense that, on average, performance was almost identically preserved (losing phase mean performance 64%, bootstrap 95%-CI $[0.55, 0.72]$), see Fig. 3.4. However, this average masks substantial individual variability. Because the figure plots *expected* earnings, and hand values were closely matched, almost all the change from one phase to the next is due to either stochasticity in the participants policies, or non-standard adaptation. (We cannot call

[1]Note that this requires some slight abuse of the notion that the basic tasks and the meta-mappings are precisely analogous, see Section 4.1.2 for some discussion.

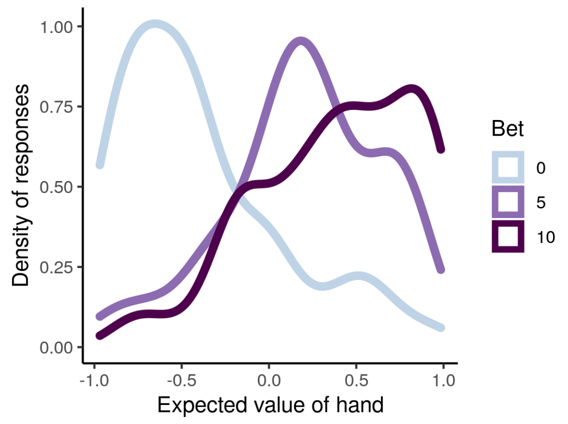


(a) Bet density by expected value.

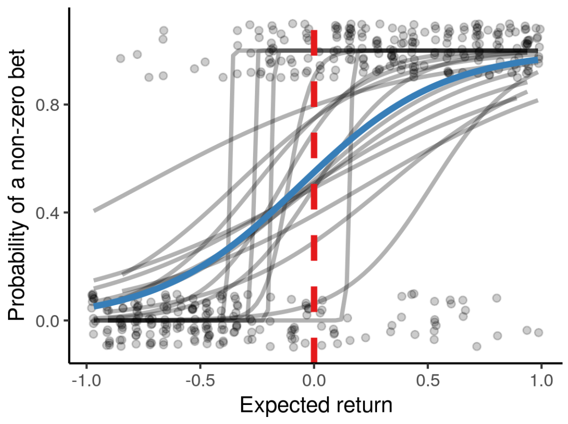


(b) Probability of non-zero bet by expected value. The red dashed line is the optimal threshold, the grey curves are the individual subject fits.

Figure 3.2: Human performance on the card game task, basic game evaluation block. While participants are performing well above chance, they are far from optimal. They make intermediate value bets, and do not switch optimally between betting and not betting. There is also substantial inter-subject variability.



(a) Bet density by expected value.



(b) Probability of non-zero bet by expected value. The red dashed line is the optimal threshold, the grey curves are the individual subject fits.

Figure 3.3: Human performance on the card game task, losing evaluation block. There is again substantial inter-subject variability.

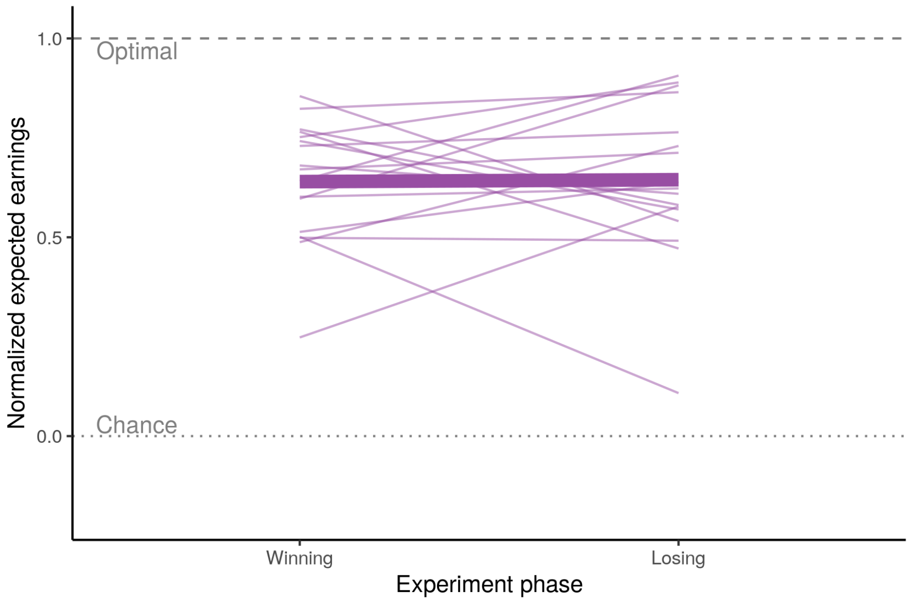


Figure 3.4: Human adaptation in the cards experiment: the change in performance from the winning evaluation block to the losing evaluation block. Larger lines are the average, smaller are individual subjects. While there is substantial individual variability, average performance is preserved.

this non-standard adaptation sub-optimal, since it sometimes results in *improved* performance on the adapted task.) It would be interesting to explore in detail what factors underlie this variability, but here we focus on the comparison to the HoMM model and language-generalization model.

In Fig. 3.5 we plot the adaptation of the models against the human results. Both models are performing near-optimally on the training tasks (see Supp. Fig. C.3 for evidence that HoMM is generalizing well at the basic meta-learning level), but of course they have much more experience with these tasks than the humans do. However, the performance of the models on the novel tasks is substantially different. Both models are worse at the losing versions of the tasks than the trained versions, but the HoMM model is still performing quite well (mean 85%, 95%-CI $[79, 90]$), while the language-based model is degrading to near chance performance (mean performance on losing variation 2%, bootstrap 95%-CI $[-12, 16]$).[2]

In order to make a more fair comparison between the human subjects and the models, in Fig. 3.6 we plot performance on the losing task, as a percentage of performance on the winning tasks. Note, however, that this metric is biased against the models, because of the ceiling effect – their scores on the adapted tasks cannot be higher than their optimal scores on the original tasks, whereas the humans are only preserving their performance on average because many of them are performing substantially better on the adapted tasks (for unclear reasons). While HoMM is adapting slightly

[2]As noted above, results are similar for the language model with a simpler task-concatenated architecture (Supp. Fig. C.4).

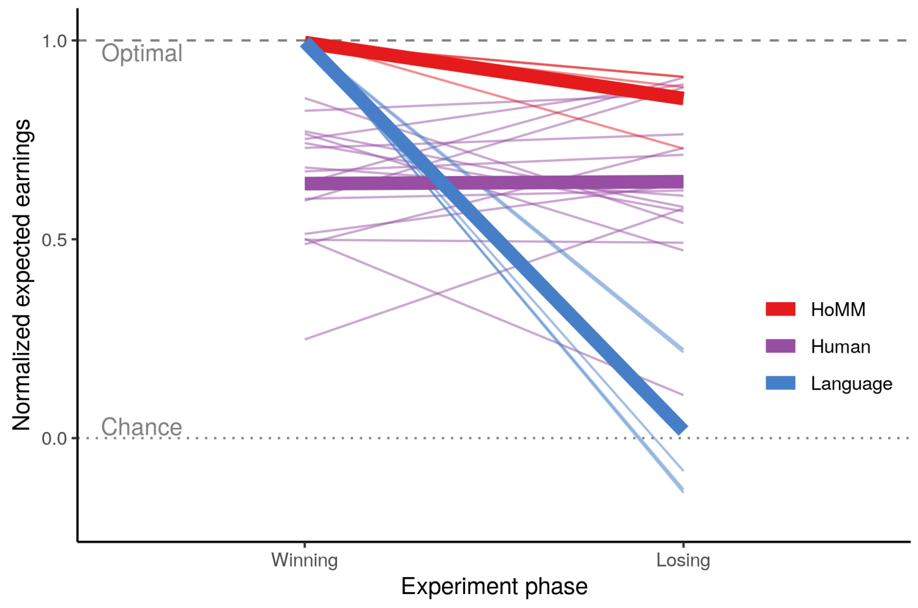


Figure 3.5: Comparing human adaptation to the HoMM and language models.

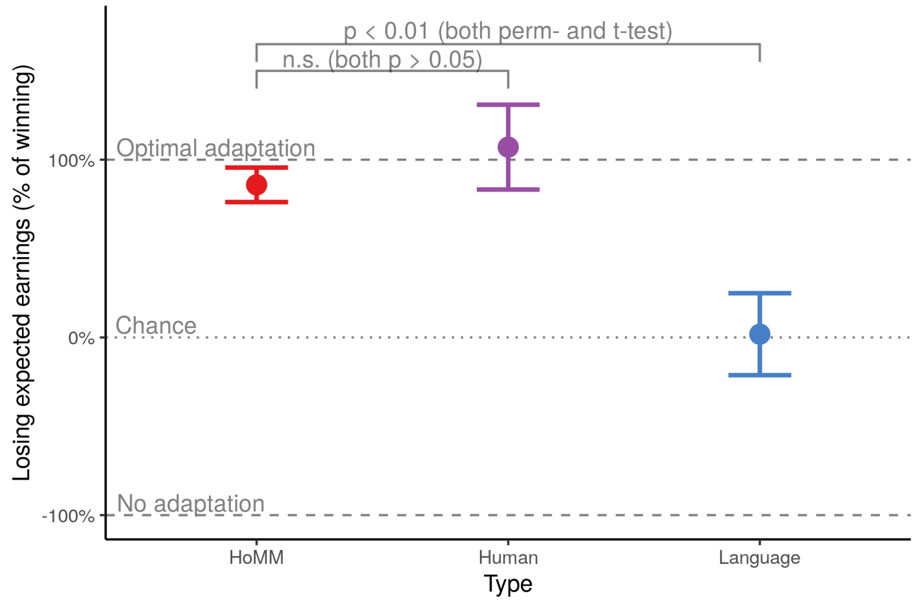


Figure 3.6: Comparing adaptation change scores (as % of score on prior task) between humans and the models. While the HoMM model is slightly sub-optimal, it is not significantly different than humans. By contrast, the language-based model is near chance on the held-out tasks.

sub-optimally, the difference between it and the human subjects is not statistically significant, either by a t-test ($t(18.54) = -1.80$, $p = 0.09$), or a permutation test ($p > 0.05$). However, HoMM is significantly better than the language-based approach, either by a t-test ($t(5.37) = 9.33$, $p < 0.001$), or a permutation test ($p < 0.005$).

3.4 Discussion

In this chapter, I have compared the ability of humans, the HoMM model, and a language baseline to adapt to losing a simple card game. The HoMM model was trained only on variations of 4 simple card games; the human participants have likely experienced a much greater variety of games over their lifetimes. Despite this, the HoMM model was able to adapt well. The human subjects also seemed to be adapting well on average, although there was substantial variability between subjects. The model appears to be adapting slightly worse than the subjects (although the difference is not statistically significant), but without giving the model an equivalent amount of experience with games as varied as those the humans have played, it's difficult to draw a strong conlusion from these results.

Instead, I interpret the comparison to humans and the language model as suggesting that mechanisms like meta-mapping may offer a useful model of human adaptation. In particular, the alternative approach of generalizing based on language did not perform nearly as well, although it suffers from the same objection that it does not have the degree of language experience that humans have. Ultimately, multiple mechanisms likely play a role in explaining adaptation in different people. Meta-mapping may be one important piece of the puzzle, but I do not want to imply that it is the only one.

It's intriguing to note that, while in this setting the language model degraded to near-chance performance on the adapted tasks, this is still much better than if it had failed to adapt at all. It's unclear whether this chance behavior is due to the model behaving systematically in some way that is not correlated with optimal behavior, or whether this performance is actually just random. Investigating this could provide interesting insight into the generalization of these models.

In summary, these results offer some insight into the adaptation capabilities of HoMM relative to humans and the language model, and suggest that meta-mapping may be a promising approach for modeling human adaptation. However, the limited number of tasks each model experienced, as well as the limited complexity of the tasks themselves, also limit the breadth of conclusions that can be drawn from the experiments in this chapter. In the next chapter, I apply the model to more complex tasks in RL and vision, that provide more challenging tests of adaptation, and support the idea that meta-mapping could be a broadly useful framework for deep learning models of cognition.

Chapter 4

Extending meta-mapping to more complex tasks

In the previous chapters we have demonstrated the success of meta-mapping in two simple domains. While those experiments allowed us to demonstrate the efficacy of the approach relative to other baselines, and compare its adaptation to that of humans, there are several reasons to extend beyond them to more complex tasks.

First, the HoMM approach relies on several ideas that it is not obvious would scale to more complex settings. For example, representing an entire task by a single vector could be challenging if the tasks are more complex. Similarly, parameterizing the task network via a HyperNetwork conditioned on a task representation could fail when the computations become more complex. Perhaps most critically, learning meta-mappings from relatively few task examples might be infeasible when the tasks themselves are more complicated. If any of these fails to extend to more complex settings, that could limit the applications of our approach.

A second, opposing, motivation for exploring more complex settings is the limitations inherent to toy experiments. While toy experiments can provide carefully controlled demonstrations of an idea, we have shown in other work that more systematic generalization can emerge when agents are placed in more realistic settings (Hill et al., 2020). This may impact both the meta-mapping approach and the language baseline, so it is important to evaluate the effects of richer environments on both. This will also help inform us as to whether our approach will be useful in more complex settings.

Unfortunately, creating truly realistic environments, and training agents in them, requires complex implementations and substantial computational resources. Thus, in this chapter we demonstrate our results in environments of moderate complexity, and leave the extension to even richer environments for future work.

In particular, we present experiments on extending our ideas to two important settings: reinforcement learning and classification from raw pixel inputs. These settings are important both because they are dominant paradigms for applying deep learning, and because they have deep connections to cognitive modeling and neuroscience (e.g. Yamins et al., 2014; Kriegeskorte, 2015; Momennejad et al., 2017).

4.1 Reinforcement learning

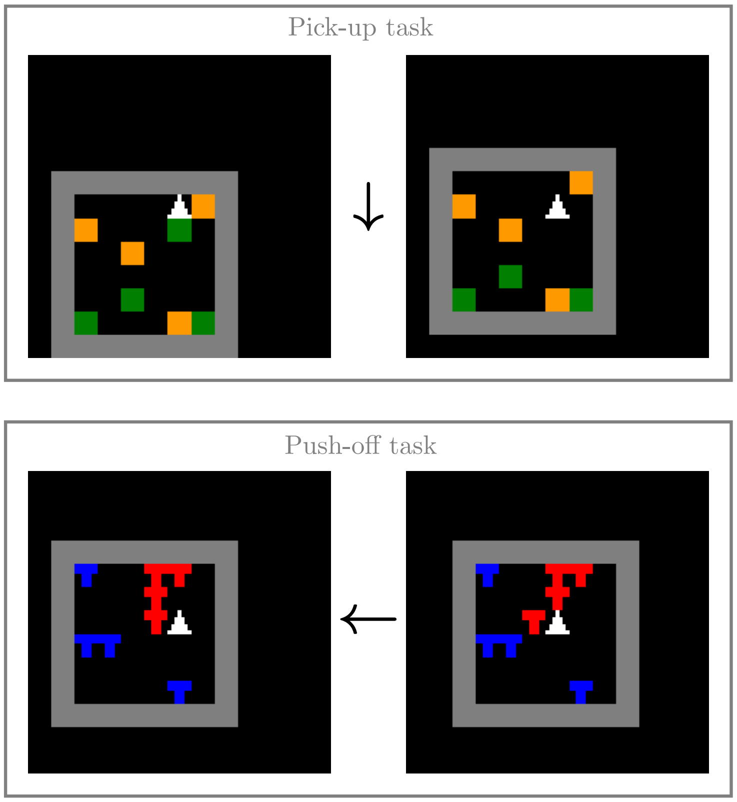


Figure 4.1: Illustrative state, action, state transitions from the RL grid experiments. In the pick-up task example (top), the agent moves downward and picks up the green object. In the push-off task example (bottom), the agent moves left and pushes the red object. Views are the visual input the agent would receive. The agent is the white triangle, note that it is always at the center of the view, because of the egocentric perspective.

Reinforcement learning is an interesting (and challenging) application for meta-mapping for several reasons. First, reinforcement learning has deep roots in neuroscience, and various RL-related computations appear to explain some aspects of neural activity (Sutton and Barto, 2017; Niv, 2009; O'Doherty et al., 2003; Dabney et al., 2020).

Second, reinforcement learning has achieved impressive human-level, performance on complex tasks such as Atari games (Mnih et al., 2015), Go (Silver et al., 2016, 2017), and complex video games like Dota 2 (OpenAI et al., 2019), and Starcraft II (Vinyals et al., 2019). This motivates it as an important place to explore more human-like intelligence.

Third, there has been a rich vein of research on adaptation in reinforcement learning, from using language-conditioned models (Hermann et al., 2017) to the observation that model-based methods or successor representations can allow for adaptation to environment changes (Daw and Dayan, 2014; Momennejad et al., 2017). However, these latter methods assume that a new reward function is given, which requires a substantial portion of the adaptation problem already be solved. Thus, there is substantial room to ask whether meta-mapping can provide good performance in RL tasks, and good motivation for a language-based approach as a comparison.

4.1.1 Tasks

To address these challenges we created a set of RL tasks based on raw visual input, with a relatively simple action space. Refer to Fig. 4.1 throughout this section for images of the visual input the agent would receive. The tasks take place in a 6×6 room with an additional impassable barrier of 1 square on each side. The squares are upsampled at a resolution of 7 pixels per square to provide the raw visual input to the agent. In addition, the agent receives egocentric input, since we have shown in other work that this is beneficial to generalization (Hill et al., 2020). That is, the agent's view is always centered on itself, and the world moves around it as the agent moves.

The agent has four actions available to it, corresponding to moving in the four cardinal directions. If it makes an invalid action, such as trying to move past the edge of the board, the state does not change. The view window is sufficiently large so that the agent can see the entire world, no matter where it is.

The tasks the agent must perform relate to objects which are placed in this space. The objects can appear in 10 different colors. In any given task, the world will have two colors of objects in it. Each color of objects only appears with one other color, so there are in total 5 possible pairs of colors that can appear. In any given task, one of the present colors is "good," and the other is "bad." On some tasks, the good and bad colors in a pair are switched.

There are two types of tasks, a "pick-up" task, and a "push-off" task. In the pick-up task, the agent is rewarded for picking up the good-colored objects by moving to their grid locations, and is negatively rewarded for picking up the bad-colored objects. In the push-off task, the agent is able to push an adjacent object by moving toward it, if there is no other object behind it. The agent is rewarded for pushing the good-colored objects off the edges of the board, and negatively rewarded for pushing the bad colored objects off. The two types of tasks ("pick-up" and "push-off") are visually distinguishable to the agent, because the shape of the objects used for them are different. However, which color is good or bad is not visually discernable, and must be inferred from the example (state, (action, reward)) tuples used to construct the task representation.

There are in total 2 task types $\times$ 5 color pairs $\times$ binary whether the good and bad colors are switched = 20 tasks in total. We trained the system on 18 of these, holding out the switched color combination of ("red", "blue") in both task types. That is, during training the agent is always positively rewarded for interacting with red objects and negatively rewarded for interacting with blue objects, across both task types.

We trained the system on the "switch-good-and-bad-colors" meta-mapping using the remaining three color pairs in the two task types, and then evaluate its ability to perform the held-out tasks zero-shot based on this mapping. Note that this is a quite difficult challenge for a model-free system, since any rewards it receives during training on similar-colored tasks are the opposite of these evaluation rewards.

4.1.2 Model

To accomodate this setting, we essentially combined the DQN architecture (Mnih et al., 2015) with our previous approaches. That is, the input to the model was raw pixels, which were passed through several convolutional layers to produce state embeddings. This visual processing was shared across all tasks. As in the card game tasks discussed in the previous chapter, we used (state, action, reward) tuples as input to the meta-network $\mathcal{M}$. More precisely, we processed actions and rewards through several layers to produce an "action-outcome" embedding, and then concatenated that to the visual embedding as input to $\mathcal{M}$.

Note that this approach abuses the notion that the inputs to $\mathcal{M}$ are always the same type of entities, just at different levels of abstraction. While the meta-mapping example tuples are still really inputs and outputs, the basic RL task example tuples are not, at least in the standard way that DQNs are designed. The standard DQN approach is to output Q values for all possible actions given a state as input. However, the basic task tuples are more like the inputs and outputs of a $Q(s, a)$ function the way it was originally designed in a tabular setting, where the action is an input to the Q function, and the Q function only produces a single output. However, such a function is a perfectly valid way to specify a task. Indeed, we will show that the HoMM architecture is able to accomodate this setting, and still perform well on both basic tasks and meta-mappings, despite the differences in their example representations.

As before, $\mathcal{M}$ produced a task embedding, which was passed through a HyperNetwork $\mathcal{H}$ to parameterize the task network F. The task network took state embeddings output by the convolutional network, and processed these to produce to produce output embeddings. The output embeddings were processed by a linear layer (shared across tasks) to produce Q-values for the different actions. When meta-mapping, the input task embeddings replaced the input state embeddings to $\mathcal{M}$ and F, and the output task embeddings replaced the action-outcome inputs to $\mathcal{M}$. (See Appendix D.1 for further details of the architecture and training, and see Supp. Fig. D.2 for a demonstration that the HyperNetwork-based HoMM architecture outperforms a simpler task-concatenated one, as in the polynomials domain.)

Perhaps because of the difficulty of the generalization problem in this setting, we found that two additional model modifications were useful. First, rather than constructing the task embedding completely from scratch each time, we kept persistent task embeddings cached, and used a random convex combination of the output of $\mathcal{M}$ and the cached embedding to perform the task. We added an additional ℓ_2 loss between the cached and transient embedding that attempted to match each to the other. Having partially persistent embeddings made it easier for the system to overcome the initial conflicting gradients caused by the fact that objects were sometimes positively rewarding and sometimes negatively rewarding, and thus made it easier for the model to discover the overall structure of the tasks.

Second, we found that incorporating weight normalization (Salimans and Kingma, 2016) in the task network increased the stability of the training process. In the simpler settings of the cards tasks, neither of these modifications were necessary. It is likely that the temporally extended nature of these tasks makes the interference between the conflicting tasks worse. However, it is possible that with appropriate hyperparameters, and enough training and time, the model could overcome this and learn without persistent representations or weight normalization in this setting as well.

4.1.3 Results

In Fig. 4.2 we show the results. We optimally stop the model for each task by requiring the training accuracy to be above a threshold (95% for the HoMM model, but 87.5% for the language model, because stricter thresholds resulted in worse language results), and using the other task as a validation set — that is, we evaluate the model on one task when the model performs well on the other task as well as the training tasks. The HoMM model substantially outperforms the language model, achieving 88.0% of optimal rewards (mean, bootstrap 95%-CI [75.0-99.0]) on the held-out pick-up task, and 71.7% (mean, bootstrap 95%-CI [42.0, 94.6]) on the held-out push-off task. By contrast, the language model is showing very little adaptation, with respective performance of -92.8% (mean, bootstrap 95%-CI [-96.3, -88.4]) and -79.7% (mean, bootstrap 95%-CI [-92.8, -59.1]) on the two tasks. This difference between the models is significant in a mixed linear regression controlling for task type and a random effect of run ($t(20.6) = -19.515$, $p < 1 \cdot 10^{-14}$).[1]

The HoMM model also exhibits significantly stronger correlations between its performance on the two tasks, both within runs at different time-points and across runs (Fig. 4.3, Supp. Fig. D.1). Specifically, the HoMM model has a correlation of $r = 0.82$ between performance on the two tasks, while the language model only has a correlation $r = 0.10$, and this difference is significant in a mixed linear model predicting push-off performance from pick-up performance, controlling for task type, epoch, and the random effect of run (main effect of HoMM $t(451.3) = 4.76$, $p < 1 \cdot 10^{-5}$, interaction

[1]Degrees of freedom calculated by the Satterthwaite approximation.

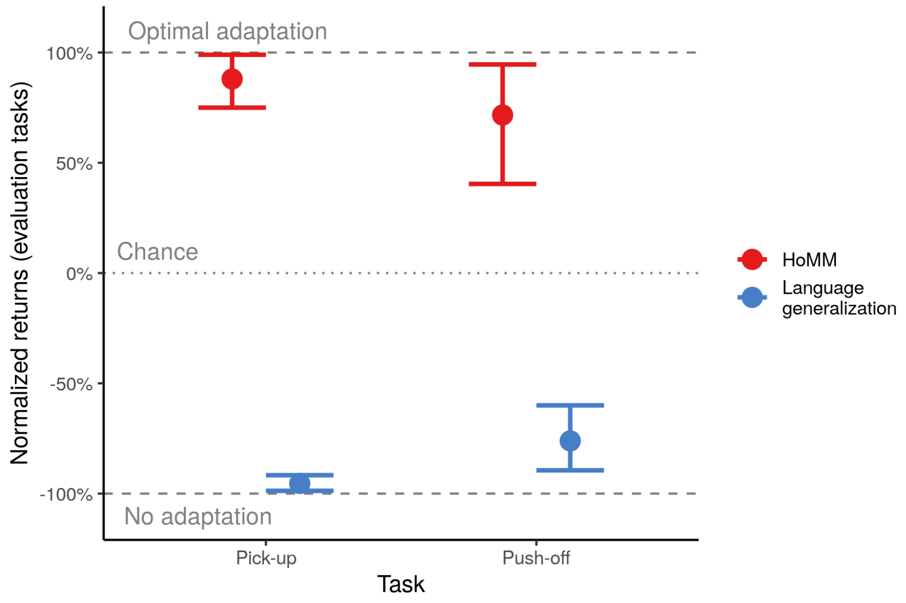


Figure 4.2: Performance on the held-out RL tasks via a meta-mapping and via language generalization. Despite the challenging nature of the adaptation (as evidenced by the language-generalization performance), HoMM is performing quite well. (Results from 5 runs, error-bars are bootstrap 95%-CIs across runs.)

of HoMM with pick-up performance $t(452.0) = 3.43$, $p < 1 \cdot 10^{-3}$). At a surface level, this means that it is easier to select a good stopping point for the HoMM model — even though the language model is achieving less bad (though still at or below chance) performance at some points in some runs, the lack of correlation between the results on the different tasks means there is no fair way to stop training the model at that point. More fundamentally, this suggests that the meta-mapping approach is exhibiting more systematic generalization, in the sense that it is either generalizing well on both tasks, or not generalizing well on both. Again, this may be more like what would be expected from human cognition.

Intriguingly, the language model does transiently exhibit slightly positive generalization very early in learning (Fig. 4.4) — however, as the model beings to master the training tasks, the generalization quickly decays to substantially below chance. The early generalization is not included in the main results in Fig. 4.2 because the train set performance is below even the more generous threshold we set for the language model. Even if it were included, this early performance is significantly worse than the HoMM model's results.

It is also interesting to explore the behavior of the HoMM model after meta-mapping. In Fig. 4.5 we show intriguing behavioral uncertainty in generalization, where the model exhibits more uncertainty (takes longer to solve the task) on novel tasks, even if it performs well. In the figure, we compare mean steps on the held-out task versions (performed via a meta-mapping) to the mean

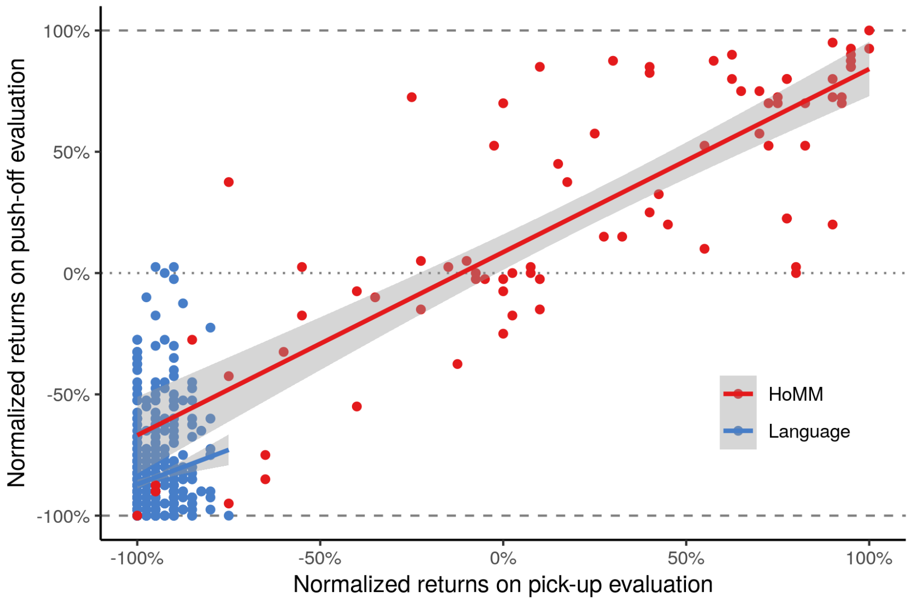


Figure 4.3: Correlation of performance on the held-out RL tasks, across runs and time points where performance on the training tasks is high. The correlation is much stronger in the HoMM model than in the language-generalization model, that is, the HoMM model is behaving more systematically in the sense that it is generalizing similarly on both tasks. (Results from 5 runs, lines are linear model fits within model type.)

steps on the trained versions (performed via a meta-mapping, for a fair comparison). In Fig. 4.5b we show that even when the model achieves similar rewards on the held-out tasks, it is doing so more slowly. Selected recordings of behavior can be found at: `https://github.com/lampinen/homm_grids/tree/master/recordings` — these show the range of behavior across the different tasks and runs. On the held-out tasks, the model often transiently exhibits uncertain behavior (e.g. running in circles) before correctly executing the task behavior. (Note that our use of a softmax policy can both contribute to this behavior, and allow an escape from it!) I have also anecdotally observed similarly longer episode lengths in RL generalization while working on the Hill et al. (2020) project. However, I did not systematically analyze those results. Exploring uncertainty in RL generalization, both with meta-mapping and within other paradigms, would be an interesting direction for future work.

Generalizing from color to shape in RL

We next evaluated the generalization capabilities of HoMM in a more challenging RL experiment. In this experiment, we trained HoMM on tasks similar to those in the main text experiments, but where the good and bad objects could be discriminated by either color (with shape matched) **or** shape (with color) matched. We trained good-and-bad-switched variations of all color tasks, but did

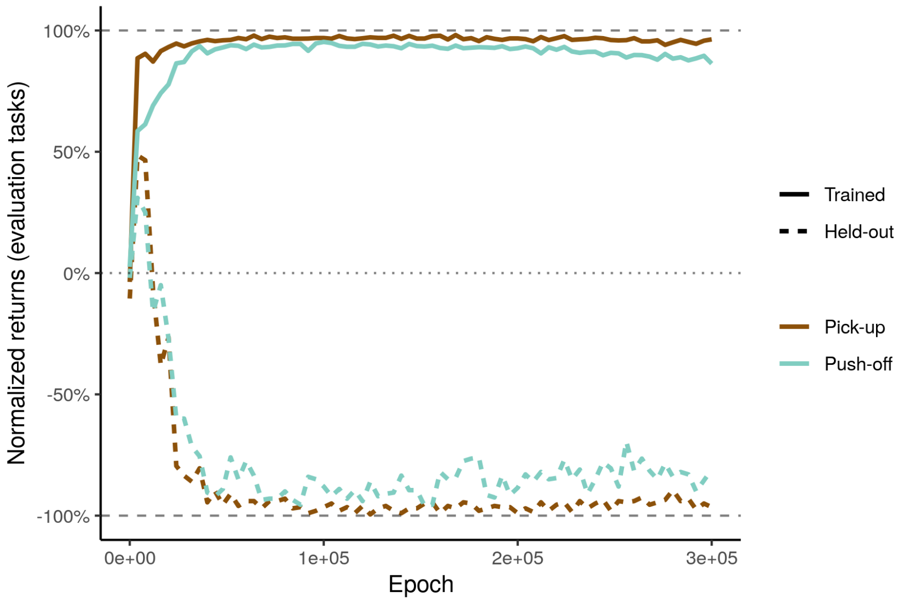
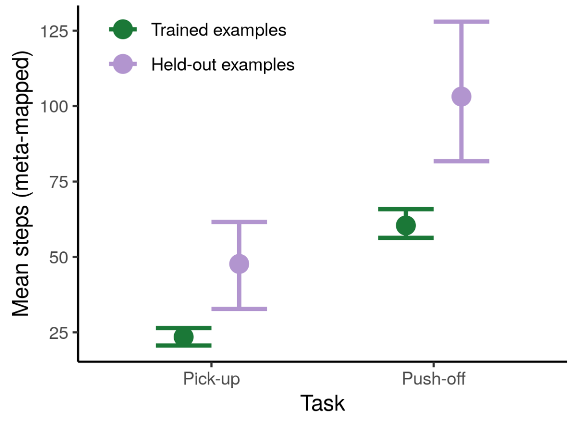


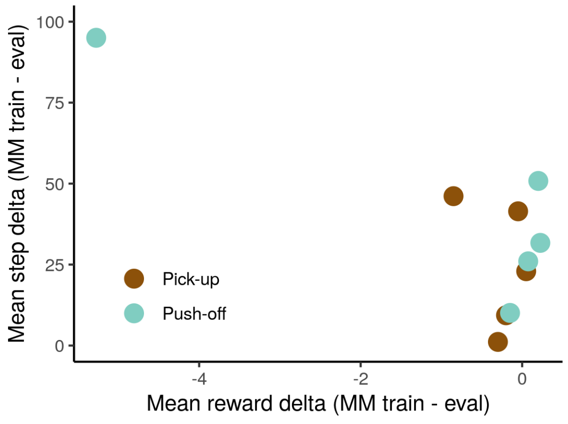
Figure 4.4: Average performance of the language generalization model over training on the RL tasks. The model exhibits intriguing, but transient, generalization early in learning, before it has understood the full structure of the tasks (especially the more difficult and sequential push-off task). However, this quickly decays to below-chance generalization as the model masters the training tasks. This early generalization is not included in the main results since the train accuracy at this time is below the threshold of having adequately learned the tasks.

not train any switched variations of the shape-discrimination tasks. Specifically, we used 8 colors, of which we used 4 for the pick-up tasks and 4 for the push-off tasks (so the task type would still be superficially distinguishable. We trained color-discrimination between two pairs of colors in each type, when presented with either both colors appearing on square shapes, or both appearing on diamond shapes. We also trained switched-good-and-bad variations of all those color discrimination tasks. We then trained four shape discrimination tasks for each game type, one in each of that game type's four associated colors. In the shape discrimination tasks, the tee-shaped objects were always good, and triangular objects were always bad.

We trained the "switch-good-and-bad" meta-mapping on the color discrimination tasks, and evaluated whether HoMM was able to correctly generalize this meta-mapping from switching colors to switching shapes, in order to infer that the triangular objects, which had always been negatively rewarded before, were now beneficial. We found it was useful to increase the initial meta-mapping learning rate to $3 \cdot 10^{-4}$, but otherwise used the same hyperparameters as the main text experiments. See Fig. 4.6 for the results. We found that HoMM was indeed able to perform well above chance at this generalization (average returns across pick-up and pusher 64.3% percent of optimal, 95%-CI [55.1, 72.6]). These experiments show that meta-mapping is able to successfully extrapolate well



(a) Mean step counts.



(b) Differences in steps vs. differences in rewards.

Figure 4.5: The HoMM model exhibits behavioral uncertainty in meta-mapping generalization on the RL tasks, measured by the steps taken to complete each episode. (a) The HoMM model takes more steps to complete episodes from the held-out tasks via a meta-mapping than to complete episodes from tasks used to train the meta-mapping. That is, it appears to be more uncertain about its behavior on the generalization tasks. (b) The behavioral uncertainty effect is not solely driven by the model performing more poorly overall; even on the runs where it performs well, it is almost always taking longer to complete the episodes from the tasks it has never seen before. To show this, we plot the difference in average steps vs. difference in average rewards between train and eval. Note that the step difference is almost always positive (evaluation tasks are slower), even where rewards are comparable. (Panel a: means and bootstrap 95%-CIs across 5 runs. Panel b: each point is one game type within one run.)

beyond the training examples of the mapping, to transform behavior along new dimensions.

Intriguingly, the language model performed less poorly at these experiments than at the main text experiments, although it was not statistically different from chance (average returns 17.8% of optimal, 95%-CI [-4.0, 37.4]). The difference in performance between HoMM and the language model was significant in a mixed model controlling for game-type (and its interaction with model) and the random effect of run ($t(76.01) = -4.60$, $p = 1.7 \cdot 10^{-5}$), while neither the effect of game type on generalization in the HoMM model, nor the interaction of game-type with model type were significant (respectively, $t(76.0) = -0.98$, $p = 0.33$ and $t(76.0) = 1.48$, $p = 0.14$).

4.2 Visual concepts

There is a long history of cognitive research on how people learn concepts or categories (Bourne, 1970; Medin and Schaffer, 1978; Kruschke, 1992; Goodman et al., 2008). This work has focused almost entirely on how people can learn a concept from examples, and more recent work in the area has focused on areas like active learning by choosing which examples to test (Markant and Gureckis,

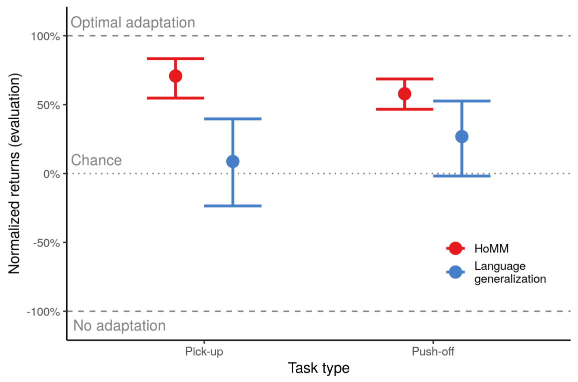


Figure 4.6: HoMM can generalize switching good and bad objects from the color dimension to the shape dimension. In this experiment, we trained HoMM on tasks similar to those in the main text experiments, but where the good and bad objects could be discriminated by either color (with shape matched) **or** shape (with color) matched. We trained good-and-bad-switched variations of all color tasks, but did not train any switched variations of the shape-discrimination tasks, to evaluate whether HoMM was able to infer how to transfer a mapping from switching colors to switching shapes. Indeed, HoMM performs well above chance at this task, though not quite as well as on the simpler generalization in the main text. Intriguingly, the language model also appears to be perfoming somewhat better in this setting, though it is not statistically above chance. (Results from 5 runs, see the text for further details of the experimental setup.)

2014b; Markant et al., 2015). However, humans can also understand concepts without any examples at all. If I teach you that "blickets" are red triangles by examples, and then I tell you that "zipfs are blue blickets," you will instantly be able to recognize a zipf without ever having seen an example (Fig. 4.7). This type of zero-shot performance can be understood as applying a "switch-red-to-blue" meta-mapping to the "blicket" classification function. This motivates applying our approach to the domain of concepts.

4.2.1 Tasks

We constructed stimuli by selecting from 8 shapes (triangle, square, plus, circle, a t shape, an outline of a square, an outline of a triangle, and 4 small squares forming a larger square) and 8 colors (red, green, blue, yellow, purple, pink, cyan, and ocean). We rendered these stimuli at 3 sizes (16, 24, and 32 pixels) at a random position and rotation within a 50×50 pixel image, to produce stimuli like those shown in figure 4.7. See Supplemental Fig. D.3 for renderings of each shape, color, and size.

We defined the basic task mappings as binary classifications of the images (i.e. functions from images to $\{0, 1\}$). We gave the system all the uni-dimensional concepts as training examples of concepts (i.e. one-vs-all classification of each shape, color, and size), so that it would be able to recognize all the basic attributes. We also constructed a set of composite tasks based on conjuctions, disjunctions, and exclusive-disjunctions (XOR) of these basic attributes.

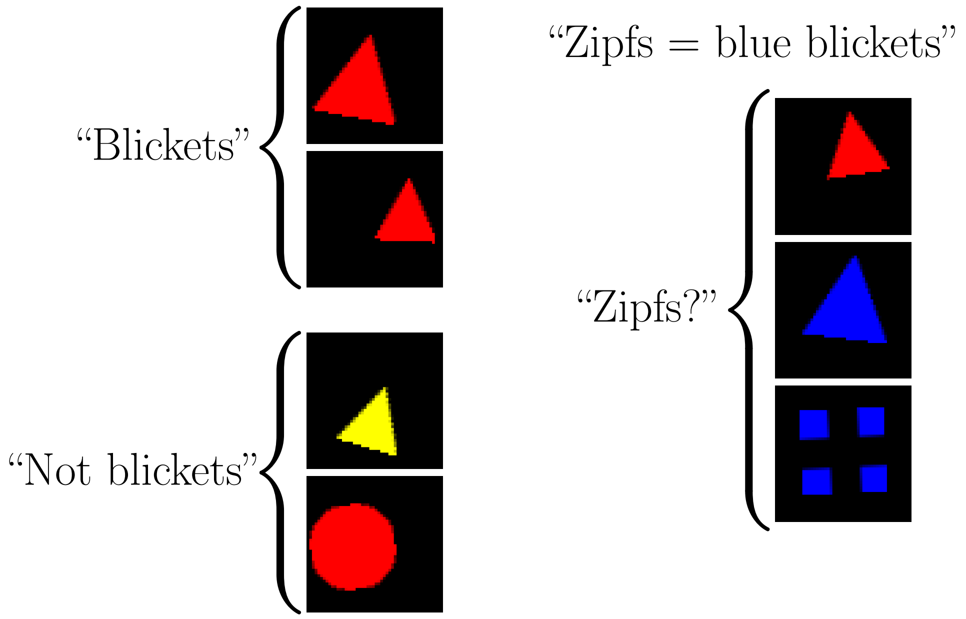


Figure 4.7: An example of zero-shot visual concept understanding that can be captured by a meta-mapping. We can understand what zipfs are if we learn about blickets, and about how zipfs relate to blickets.

For each concept, we chose the examples so that the datasets were balanced (that is, there was an 50% chance that each item in the dataset was a member of the category), both during training and evaluation. We only included negative examples that were one change away from being a member of the category. These careful contrasts may be beneficial during training – recent work has shown contrasting examples to be useful for causing neural networks to extract more general concepts (Hill et al., 2019). They also make the evaluation tasks more challenging, and therefore increase our ability to dscriminate partial understanding of the concept from complete understanding.

We trained the system on a subset of the concepts, and on meta-mappings that switched one shape for another, or switched one color for another. We evaluated the system on its ability to apply these meta-mappings to basic tasks it was trained on in order to perform the held-out basic tasks. Because there are many meta-mappings available in this setting, we were able to hold out one shape meta-mapping and one color meta-mapping for evaluation, and we will also show results on adaptation based on these held-out mappings.

4.2.2 Model

We used a 4-layer convolutional network for input embedding, and a linear task network F, followed by a 2-layer output network (shared across tasks). We found that constructing task representations from language was more effective here than constructing them from examples — see below.

We also found that in this setting the optimal architectures for the task network differed for language-based generalization and meta-mapping-based generalization. In particular, we show in Appendix D.2.1 that, while a linear task network worked better for the meta-mapping model, a nonlinear and deeper one generalized better from language. This is likely because the linear map provides a useful inductive bias for meta-mapping. This raises the possibility that adding linear skip-connections to nonlinear task networks which might allow for better meta-mapping performance as well as better task representations. It is also possible that using examples and language together to infer tasks would improve task representations further. Both of these provide exciting directions for future investigations.

4.2.3 Results

Comparing different sources of task representations

We found that constructing task representations from language worked better than construting task representations from examples in this setting. In Fig. 4.8, we show that language generalization (mean = 0.92, bootstrap 95%-CI $[0.89, 0.94]$) appears better than that obtained from meta-mapping task representations constructed from examples (mean = 0.90, bootstrap 95%-CI $[0.89, 0.91]$). However, this comparison is not statistically significant under a mixed model ($t(8) = -1.5$, $p = 0.18$). However, meta-mapping with language-based task representations performs better (mean = 0.95, bootstrap 95%-CI $[0.94, 0.97]$, mixed-model $t(8) = -2.9$, $p = 0.02$). Thus it appears that language may be a better way of constructing task representations in this setting, but meta-mapping a prior task still results in better zero-shot generalization than language alone.

Why does language result in better task representations in this setting? Of course, language conveys the basic concept more cleanly than examples can, but this was also true in other settings, where language did not seem as advantageous. First, several exciting lines of work are converging on the role of language in shaping our concept representations. Recent work has shown that more "nameable" concepts are easier to understand (Lupyan and Zettersten, 2020), and that language may be beneficial to visual concept learning in machine learning (Mu et al., 2019). Thus, these results may be reflective of broader issues about how concepts are formed. But perhaps more importantly, some concepts can be difficult to discriminate from a small set of examples (such as XOR vs. OR). In order to provide a fair evaluation, we would have to sample the examples much more carefully to ensure that every set we used was unambiguous. We adopted the simpler approach of simply constructing task representations from language for the following experiments.

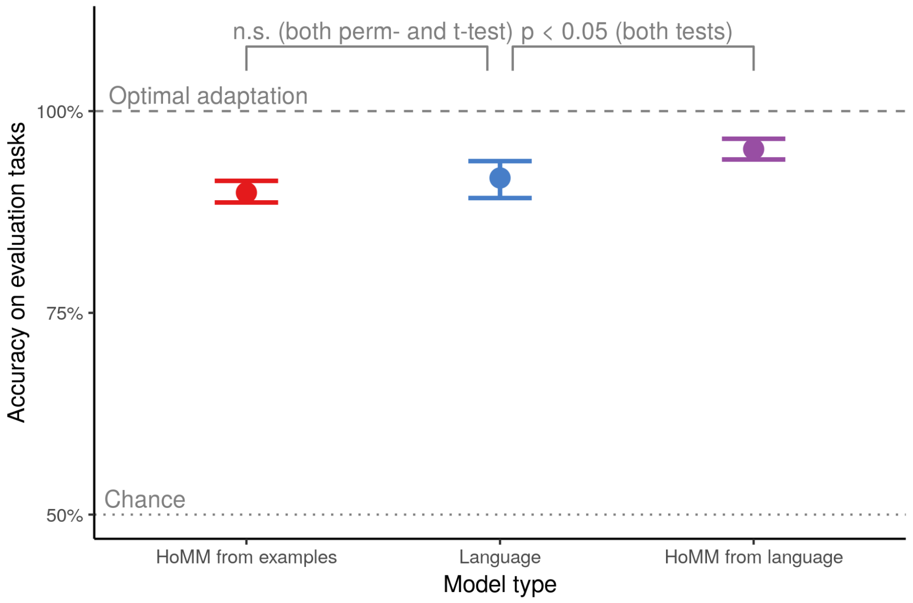


Figure 4.8: Accuracy on held-out visual concepts based on meta-mapping (either from examples or language), or language generalization. While language generalization appears non-significantly better than meta-mapping from examples, transforming language-based task representations with meta-mapping performs significantly better.

Evaluating the effects of training set size

We next evaluated the language-based HoMM model and language generalization with held-out meta-mappings, at varying sample sizes. Specifically, we trained the model with either 4 meta-mappings total (2 switch color, and 2 switch-shape), or 8, 16, 24, or 32. With each mapping, we included 6 training examples of the mapping (one for each pairing of composite rule type and other attribute). We also included 6 other pairs for evaluation, where the source concept was trained, but the target was held-out for evaluation. That is, the number of basic concepts the system encounters during training is roughly 18 per meta-mappings trained (roughly because it can be reduced if the meta-mappings have overlapping examples), and the number of evaluation concepts is roughly 6 per meta-mapping. However, this sampling of the tasks also has a drawback, see below. For example, the system might be trained on mappings like “switch-red-to-blue,” with corresponding examples like AND(red, triangle) $\mapsto$ AND(blue, triangle). It would then be evaluated on closely matched examples like AND(red, circle) $\mapsto$ AND(blue, circle), where the latter is untrained.

We also included a set of training and evaluation basic tasks for two held-out meta-mappings, one “switch shape” and one “switch color.” For a held-out meta-mapping, e.g. “switch-green-to-blue,” the same basic concepts instantiating the meta-mapping were trained as for a trained mapping, but the meta-mapping itself was not. We were thus able to evaluate how increasing training affects both

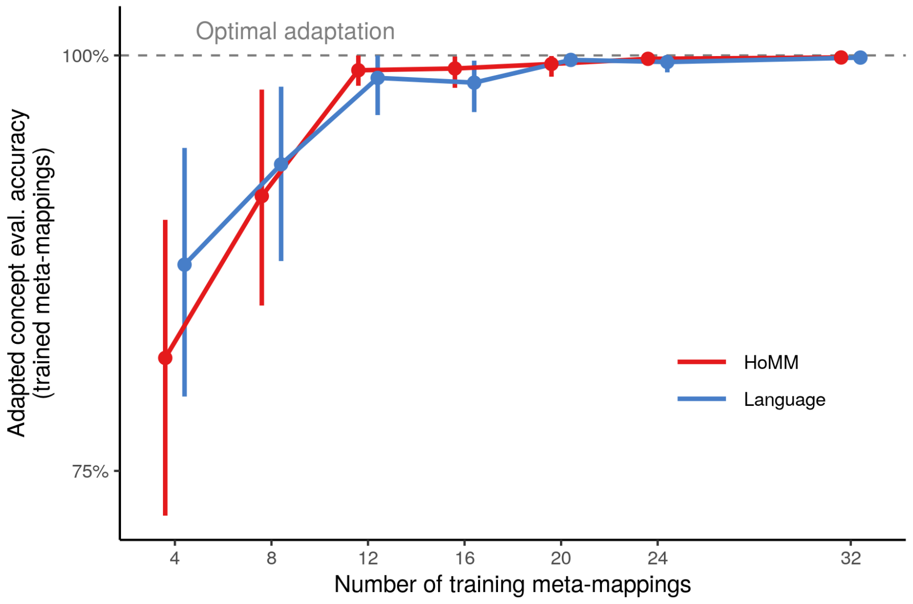


Figure 4.9: Applying HoMM to visual concepts after training on different numbers of meta-mappings (or the equivalent set of training concepts). HoMM and language generalization are well above chance (50% with our balanced evaluation sets, not shown). HoMM is able to generalize trained meta-mappings to perform new tasks zero-shot, and performs comparably to language generalization. (Results are from 10 runs of each model with each training set size. Errorbars are bootstrap 95%-CIs across runs.)

the generalization of the model within a meta-mapping, and its ability to generalize to a held-out meta-mapping.

In Fig. 4.9, we show the results for trained meta-mappings. The language model and the HoMM model perform quite comparably in this domain. The HoMM model may show a slight advantage at moderate training set sizes, but the magnitude of the difference is very small, much smaller than that in the previous experiment. In a mixed linear model, language generalization results in very slightly worse generalization at moderate numbers of training mappings (-1.50%, $t(2612) = -2.775$, $p = 0.006$), and a small interaction with number of training meta-mappings ((-0.25% per trained meta-mapping, $t(2617) = -4.26$, $p < 0.001$). (Effect of one additional trained mapping for HoMM 1.00%, $t(6828) = -5.52$, $p < 0.001$.)

Why did HoMM and language generalization perform similarly in this experiment? One possible explanation is the sampling of basic concepts we chose. By ensuring that each meta-mapping would be supported by a set of examples that span all possible relations and types, we ensured that there is a training task that is very similar to each evaluation task, which may have made it easier for the language model to interpolate. By contrast, in the results shown in Fig. 4.8, we sampled tasks randomly, which means some tasks would have required more extrapolation. See below for further

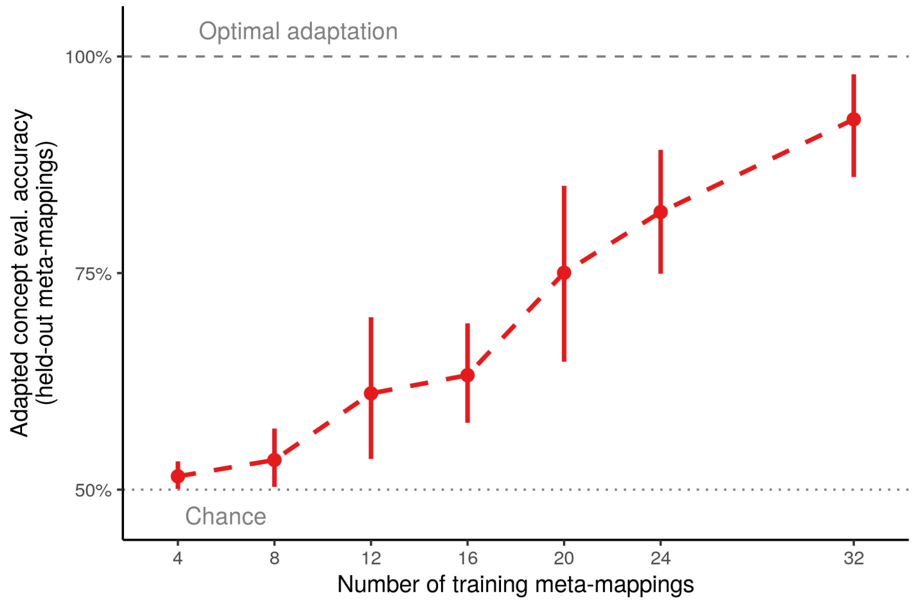


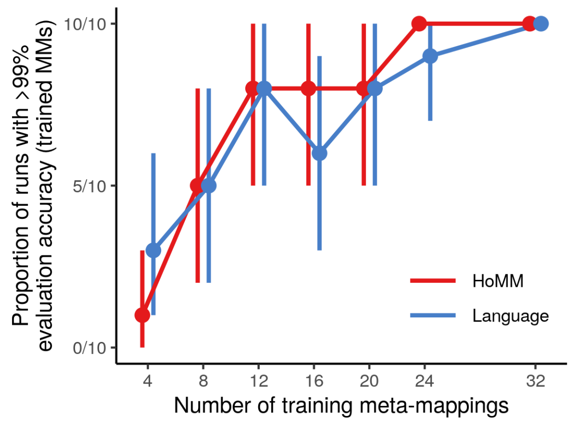
Figure 4.10: Evaluation HoMM on held-out meta-mappings visual concepts at different training set sizes. Results are shown after training the model on various numbers of training meta-mappings. The HoMM model is able to generalize to adaptation based on held-out meta-mappings, once it experiences sufficiently many training meta-mappings. (Results are from 10 runs of each model with each training set size. Errorbars are bootstrap 95%-CIs across runs.)

discussion.

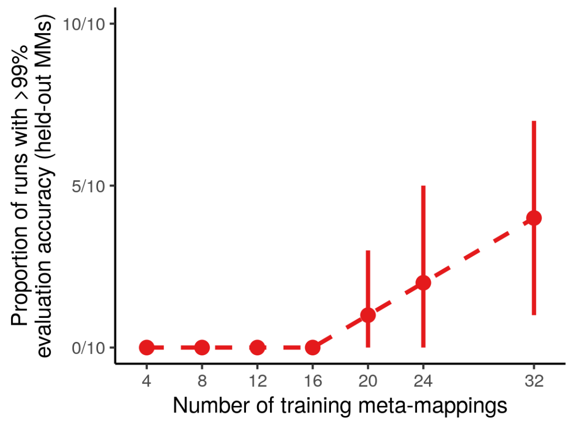
Once the HoMM model has seen a relatively small set of meta-mappings (32), it is able to generalize quite well to held-out meta-mappings from their language description, as shown in Fig. 4.10. Although the average performance from held-out meta-mappings is not perfect, it is perfect in a sizable proportion of the runs (Fig. 4.11b). See Supp. Fig. D.6 for learning curves for each run and sample size. We find these results impressive, given that the model experiences at most 32 training meta-mappings — performing held-out meta-mappings from language is a generalization problem analogous to performing a held-out basic task from langauge, and with a similar number of training basic tasks language generalization was at chance in the card game experiments in the previous chapter. The result is even more impressive when considering that the complexity of a meta-mapping in function space is much greater than the complexity of a basic task mapping. Thus, we find these results to be encouraging.

4.3 Discussion

In this chapter, we have shown that HoMM can perform well in more complex domains than we had previously explored, while still requiring only a relatively small set of training tasks. In the RL



(a) Trained meta-mappings.



(b) Held-out meta-mappings.

Figure 4.11: In the visual concepts domain, the proportion of runs in which each model attained $> 99\%$ accuracy on the transformed concepts. (a) Trained meta-mappings. Both approaches show extremely systematic generalization, even at moderate sample sizes. (b) Held-out meta-mappings. At the largest sample sizes we considered, the HoMM model is able to adapt near-perfectly to new meta-mappings on many runs. Note that even at this largest sample size, the system is generalizing from only 32 trained meta-mappings.

domain, HoMM generalized a single meta-mapping well from only 16 example tasks (18 trained tasks total). In the visual concepts domain, HoMM was able to generalize trained meta-mappings perfectly on every run once a sufficient number of training meta-mappings were provided, and was able to generalize to held-out meta-mappings quite well. Generalization on the held-out meta-mappings continued to improve beyond the point that trained meta-mapping generalization was at ceiling.

These results suggest that HoMM may be a broadly usable approach. HoMM does not seem to require an unreasonable number of training tasks or examples of a meta-mapping, even in more complex settings than we considered previously. Furthermore, HoMM performs well in an RL setting where the “ examples” take the form of (state, action, reward) tuples rather than (input, output) tuples. HoMM also performs well when the task and meta-mapping representations are generated from language, as we showed in the visual concepts domain. Both these results help illustrate the generality of HoMM — in its abstract formulation, it is able to accomodate many different computational paradigms, and these empirical demonstrations show that this abstract possibility can be realized practically.

The results of the language generalization approach that we compared to are particularly striking in this chapter. In the RL tasks, it generalized much worse than chance (once the training tasks were learned), and thus much worse than it did in the previous chapter. By contrast, in the visual concepts setting, it performed competitively with the meta-mapping approach, and thus much better than in the previous experiments. What is the cause of this discrepancy?

There are a few possible factors underlying these results. First, because of the structure of the task spaces, the number of training tasks in the visual concepts domain is much larger. The language model seems to need more training tasks than HoMM to begin to generalize well. However, as noted above, there is another factor that may be driving the differences between the results in this chapter. The training tasks in the RL setting more directly contradict the evaluation tasks, and there is no task that is close to the evaluation tasks. By contrast, in the visual concepts domain the sytem will have encountered many training concepts that overlap with the held-out evaluation in one attribute. Indeed, when we evaluated the effects of sample size, our task sampling scheme ensured that there would be a training concept closely matched to every evaluation concept.

It's also worth noting that in other work where language generalization performs well from small numbers of training tasks (e.g. our work in Hill et al., 2020), the held-out tasks are generally not diametrically opposed to the trained tasks. Instead, they tend to be relatively close interpolations from relatively densely sampled training tasks.

It is therefore possible that the HoMM model generalizes better to tasks farther outside the space of its experience than the language model does. In fact, we observed in the RL domain that the HoMM model was exhibiting an intriguing signature of generalizing more systematically than the language model. Generalization on the two held-out tasks was more tightly correlated in the HoMM model than in the language model. Furthermore, HoMM was able to extrapolate a meta-mapping learned on color discrimination tasks to shape discrimination more accurately than the language model was able to extrapolate to these tasks. These results are suggestive of the more systematic generalization (or systematic failure to generalize appropriately) that humans sometimes exhibit.

It's also worth reflecting at this point on the results of Hill et al. (2020), in which we showed that more realistic environments improve language model generalization. The realism of the environments is matched between models in this setting, and I tried to incorporate features that we improved generalization in that project (such as egocentric perspective on the RL tasks). However, it will be important to evaluate both types of models in more realistic settings in the future. Ideally, both classes of models would perform even better in more realistic settings, but it is possible that one approach will benefit more from realism than the other.

In summary, the HoMM model is promising in more realistic and complex settings that are more representative of the challenges that humans (and modern AI systems) face. It does not seem to require unreasonably large sets of training tasks in these settings, and may even adapt more systematically than the language-based model. But what can we do with this adaptation? In the next chapter, we explore one key idea: zero-shot adaptation provides a good starting-point for later learning.

Chapter 5

Learning across different timescales

A large swath of recent machine learning research can be seen as studying interactions of learning across different time-scales. In particular, meta-learning focuses on the idea that a model can slowly learn over many tasks how to learn rapidly in a new task. Similarly, the results of the HoMM approach in previous chapters show how slowly-accumulated knowledge about tasks and their relationships can allow zero-shot inferences about a new task.

However, both of these approaches examine how slowly learned knowledge can improve rapid learning. Yet one of the core motivations of complementary learning systems theory was that rapidly learned experiences could be integrated into our prior knowledge. There is a lack of research investigating how what a model learns over short time scales, for example in the inner loop of a meta-learning algorithm, can be integrated with its longer term knowledge. In standard meta-learning approaches, the inner loop knowledge is discarded before the next episode, or only a small update is made to incorporate it. Yet when humans learn something new, we can remember it in detail, and use that knowledge in the future.

In machine learning, Integration of knowledge is mostly studied under the framework of continual learning. Most work on continual learning investigates the setting where a model, starting from *tabula rasa*, must learn a sequence of tasks without forgetting (Ven and Tolias, 2018; Atkinson et al., 2018). This is motivated by the clear ability of humans and animals to learn multiple tasks without forgetting. However, humans are not starting from a blank slate when we achieve this. In McClelland et al. (2020), we show how prior knowledge affects what is easier or harder to learn, and show that prior knowledge must be replayed only to the extent that it is similar to (and thus interferes with) new knowledge. Furthermore, Velez and Clune (2017) show that systems can meta-learn to learn without forgetting. Thus works that examine continual learning from a blank slate are misleading, because the structure of prior knowledge changes what is easy or difficult to learn, and prior learning can be an important part of the solution to catastrophic interference.

We have shown in the prior chapters that using knowledge of prior tasks can allow the system to perform well on a new, related task, without any data. Here, we highlight the impacts of this rapid adaptation over longer time-scales of learning. In particular, we demonstrate a technique by which this zero-shot inference can improve learning on the new task, and that the knowledge encoded in the system can allow this learning to occur without even the possibility of interference with prior tasks.

5.1 Starting points for learning

When humans begin a novel task, we often receive some instructions as a starting point. These instructions often describe the relationship of the novel task to prior experiences. This observation served as the motivation for the previous chapters, in which we showed that using meta-mapping could improve zero-shot task performance. However, as soon as we start performing a task, it is no longer zero-shot. That is, zero-shot adaptation is most important insofar as it serves as a useful starting point for later learning. In this chapter, one of our primary goals is to compare the zero-shot "guess" at a task representation to other task representation initializations, to evaluate whether adaptation is a beneficial starting point for learning.

In order to do this, I adopt an approach related to some of my prior work on one-shot learning of word embeddings (Lampinen and McClelland, 2017). In that work, we integrated a novel word into a pre-trained language model by simply optimizing its embedding(s) to improve the model's prediction of it in context. We did this without altering any network parameters other than this embedding. We showed that this allowed reasonable learning of new words, in fact average performance with this method was not statistically different than if the word had been included in the training corpus from the beginning (at least for rare words). Thus in a model that has been pre-trained to understand the latent structure of a system (such as language), optimizing the representation of a single object can be a sufficient way to construct a high-quality representation of the object, without needing to alter the other parameters of the model. This is perhaps the strongest case of learning without interference.[1] By design, learning by optimizing the representation of a specific item cannot alter prior knowledge about other items.

Analogously, in this chapter I explore optimizing the task embedding of a novel task once the system has begun to perform that task. A similar approach to learning new tasks was proposed by Reed and de Freitas (2015). I will show that optimizing the task embedding alone will often allow near-perfect performance on a novel task, provided the model that is pre-trained on sufficiently many other tasks from the same distribution. This means that a new task can be learned without the possibility of interference with prior tasks, because only task-specific parameters are altered. This provides a new perspective on continual learning (though see (Oswald et al., 2020) for some related observations). Rather than thinking about how a system can minimize interference when learning a

[1]The structure of model's representations can also allow for some things to be learned without interference (McClelland et al., 2020).

sequence of tasks from *tabula rasa*, we should perhaps ask how prior knowledge can allow learning without any interference at all.

The important observation from our perspective is that a zero-shot guess at a task embedding provides a useful starting point for this optimization. In particular, we compare to a variety of other initializations, and show that the zero-shot guess provides faster learning, and lower cumulative error. This latter measure can be thought of as analogous to the notion of *regret* in reinforcement learning, a measure of how sub-optimally the algorithm performs while learning to behave optimally. Starting from the output of a meta-mapping results in learning faster, and making fewer mistakes along the way. This may be part of the solution to why humans are able to learn faster and more accurately than deep learning models on novel tasks. It also has the potential to lead to much safer exploration in a new setting, as long as risks can be related to prior experience.

5.2 The polynomials domain

We begin by demonstrating these results in the simple polynomial regression domain that we considered in Chapter 2. We train the model as before, and then attempt to learn 800 held-out polynomials by optimizing a task representation for each one. We also compare to an untrained model, to show the importance of prior knowledge to the optimization procedure.

In the trained model, we compare four different initializations for the optimization. We first consider a small random initialization, as is often used for parameters of a deep model. We next consider initializing each task with the embedding of a random trained task, in case the distribution of these is helpful for later learning. We then compare to initializing with the centroid of all trained task representations, which is perhaps the most reasonable task-agnostic starting point (c.f. Lampinen and McClelland, 2017). Finally, we compare to the estimate of the representation produced by meta-mapping from a prior task.

As noted above, we make this comparison by optimizing the task embeddings for the new tasks. We do this without altering any other parameters in the model. It is not clear that this would be sufficient to produce good performance on a novel task, indeed we show below that in an untrained model it is not. However, if the model has sufficient experience on enough related tasks, this approach suffices.

We used a similar distribution of tasks to our polynomial results presented in Chapter 2, except that we did not include held-out meta-mappings. This eliminates the uncertainty introduced by having to learn a new mapping from examples, as well as applying the transformation, which allows for a more controlled comparison. To be precise, we trained the system on 60 base tasks plus the results of applying 20 meta-mappings to them. We additionally trained the system on 40 new base tasks, and held out the results of applying the 20 meta-mappings to them. We then optimized the task representations for these $40 \times 20 = 800$ novel tasks, which the model never encountered during training.

5.2.1 Basic results

In Fig. 5.1, we show how the log-loss on the new tasks changes over epochs of learning on the new tasks (i.e. optimization of the task representations). We compare the output of the meta-mapping to a variety of sensible initializations, as well as an untrained network. We start by considering the untrained network — while optimizing the task embedding alone is able to achieve performance well above chance, it is not able to capture the finer details of the task even after training for many epochs. This shows that prior knowledge of related tasks is key to learning with this method.

In the trained model, by contrast, all embedding initializations suffice to produce good performance eventually. However, the better starting point provided by a meta-mapping initialization results in both lower initial error, and faster learning early on, so much lower cumulative error. Other sensible initializations, such as the centroid of the representations of the trained tasks, perform much better than random initialization.

To demonstrate this last point, in Fig. 5.2 we plot the average cumulative "regret" on the novel tasks for the different initializations. That is, we plot the integrated error over the course of learning. This measures how much loss the model must suffer in order to achieve perfect performance on the task. Starting from a meta-mapping results in almost an order of magnitude less cumulative error (mean = 24.58, bootstrap 95%-CI $[17.71, 32.08]$) than the next best initialization (centroid of trained task representations, mean = 192.89, bootstrap 95%-CI $[151.98, 234.53]$). That is, meta-mapping reduces the cumulative error by nearly an order of magnitude.

Thus we conclude that, at least in this simple setting, the zero-shot initialization is advantageous in reducing the time to reach near-optimal performance, and the cumulative regret (errors made along the way).

5.2.2 The advantage of hyper networks for later learning

We have compared our hyper-network-based HoMM architecture to the simpler alternative of concatenating a task representation to an input embedding before passing it through a fixed network, in various supplemental analyses (Supplemental Figures B.2, C.4, and D.2). The hyper network approach generally performs at least as well as, and sometimes substantially better than, the simpler approach. Hyper networks may also be particularly beneficial for continual learning (Oswald et al., 2020). Furthermore, they may make it easier to optimize the task representation, by giving it more direct control over the computations of the network. Thus, it seems useful to compare these two architectures in this setting.

We therefore performed the above experiments with the simpler task-network architecture. In Fig. 5.3, we show the learning curves for both architectures for the two best initializations (meta-mapping output, and centroid of the trained task representations. The hyper-network architecture

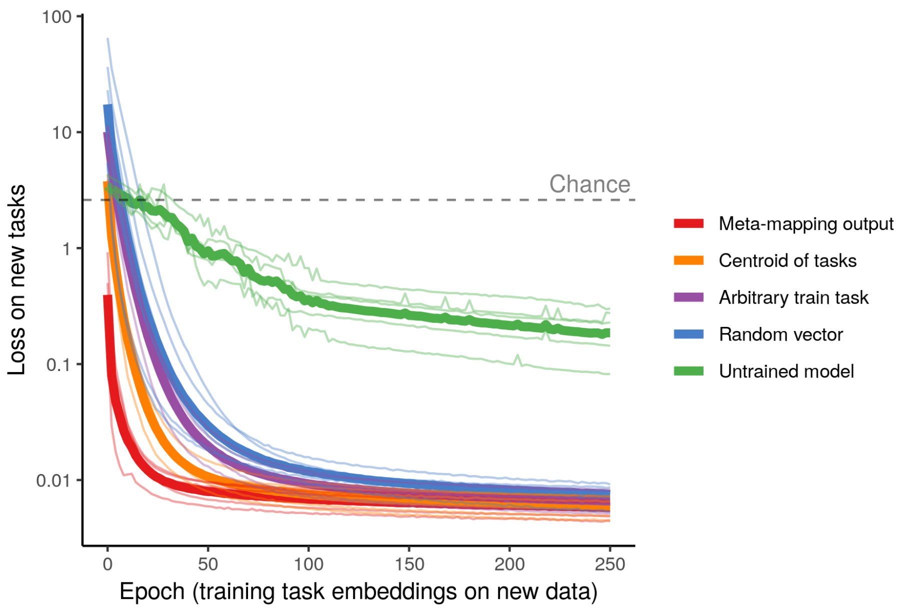


Figure 5.1: Learning curves when optimizing the task embedding for new polynomials from various starting points. The meta-mapping initialization provides a much better starting-point, and reaches near-optimal performance much faster. (Note that the y-axis is log-scale. Results are from 5 runs, individual runs are shown as light curves.)

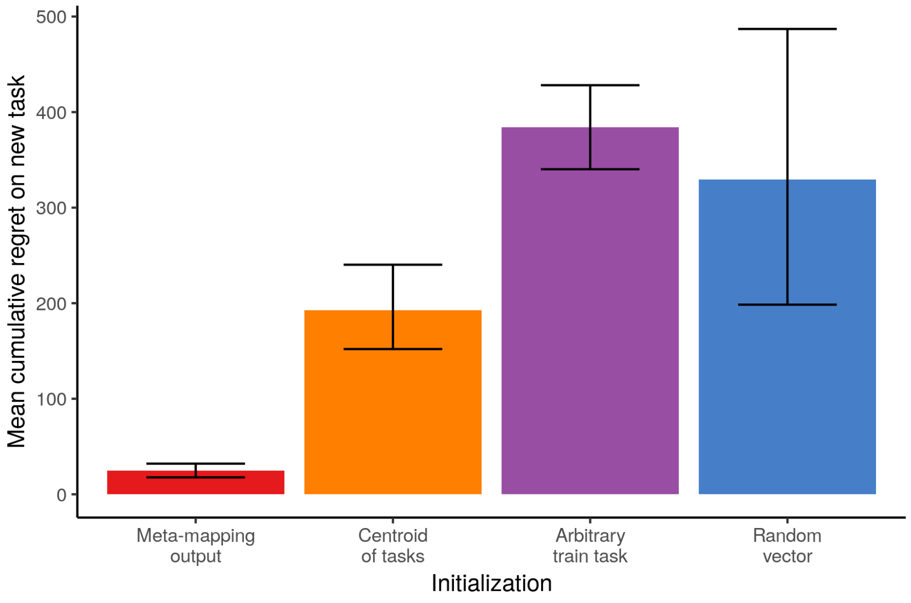


Figure 5.2: Cumulative loss when optimizing the task embedding for held-out polynomials from various starting points. The meta-mapping initialization results in an order of magnitude less cumulative error over the course of learning. (Results are from 5 runs, errorbars are bootstrap 95%-CIs.)

learned much more rapidly than the simpler architecture. The initial meta-mapping outputs do not differ so substantially — most of this effect is due to the slower improvement of the loss when optimizing the task representation in the non-hyper architecture. Indeed, optimization in the non-hyper network architecture appears to be plateauing at a much higher loss value than in the hyper-network architecture.

As before, we quantify this difference by plotting the cumulative loss on the novel tasks in Fig. 5.4. The simpler non-hyper architecture resulted in about five times greater cumulative loss than the hyper network architecture when starting from the meta-mapping output (mean = 133.81, bootstrap 95%-CI $[102.65, 171.10]$), and similarly from the centroid of the trained task representations (mean = 1139.35, bootstrap 95%-CI $[943.60, 1344.52]$).

We therefore conclude that hyper-network-based architectures may be particularly conducive to this perspective on continual learning.

5.3 The visual concepts domain

In order to evaluate the breadth of the approach we proposed in this chapter, we next evaluate learning from a meta-mapping starting point in the visual concepts domain that we considered in Chapter 4. We trained the model with 16 meta-mappings (8 switch-color and 8 switch-shape), and with approximately 18 training tasks per meta-mapping, chosen as in the experiments where we varied numbers of training meta-mappings. However, we did not include any held-out meta-mappings.

5.3.1 Results

In Fig. 5.1, we show how average accuracy on the new tasks evolves over epochs of learning on the new tasks. Intriguingly, while small random initializations performed worse than centroids or arbitrary task embeddings in the polynomials domain, they perform substantially better in the visual concepts domain. However, meta-mapping output still has a substantial advantage, because it is achieving extremely high accuracy zero-shot.

To explore this further, in Fig. 5.2 we again plot the average cumulative errors on the novel tasks for the different initializations. Initializing with a meta-mapping results in very low cumulative error (mean = 0.33, bootstrap 95%-CI $[0.10, 0.57]$) compared to the next best approach (small random initalization, mean = 9.62, bootstrap 95%-CI $[6.63, 13.59]$). This difference is significant in a mixed linear model ($t(4) = 4.628$, $p = 0.01$).

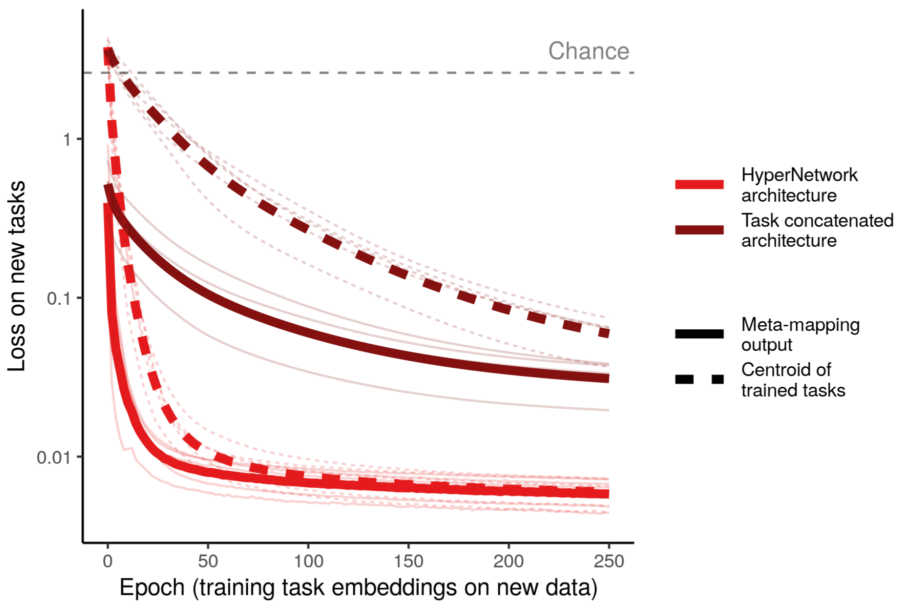


Figure 5.3: Comparing the learning curves of the hyper network architecture and a simpler architecture when optimizing the task representations for new polynomials. The simpler architecture improves much more slowly, and appears to plateau at a higher loss. (Note that the y-axis is log-scale. Results are from 5 runs, individual runs are shown as light curves.)

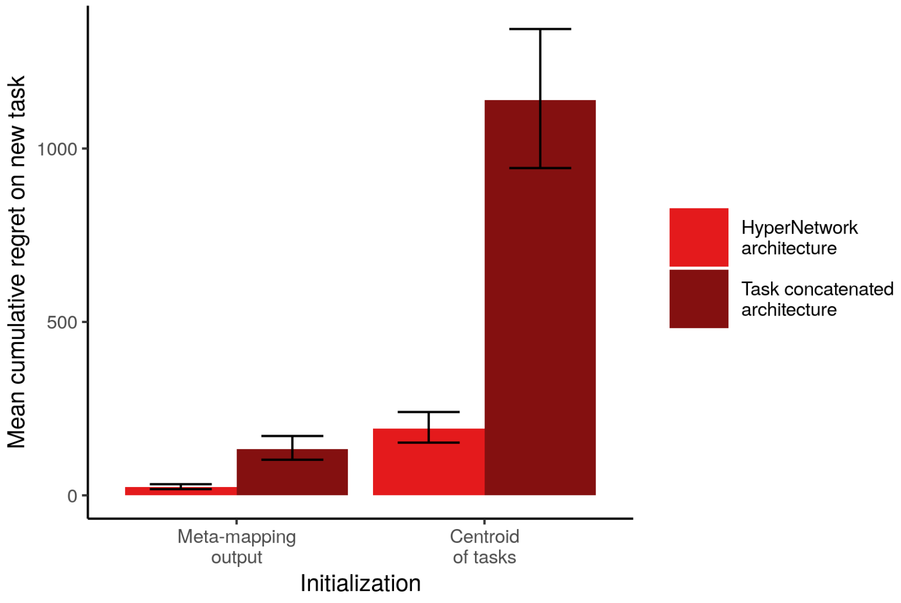


Figure 5.4: Comparing the cumulative losses of the hyper-network architecture and a simpler architecture when optimizing the task representations for new polynomials. The simpler architecture results in substantially more cumulative loss. (Results from 5 runs, errorbars are bootstrap 95%-CIs.)

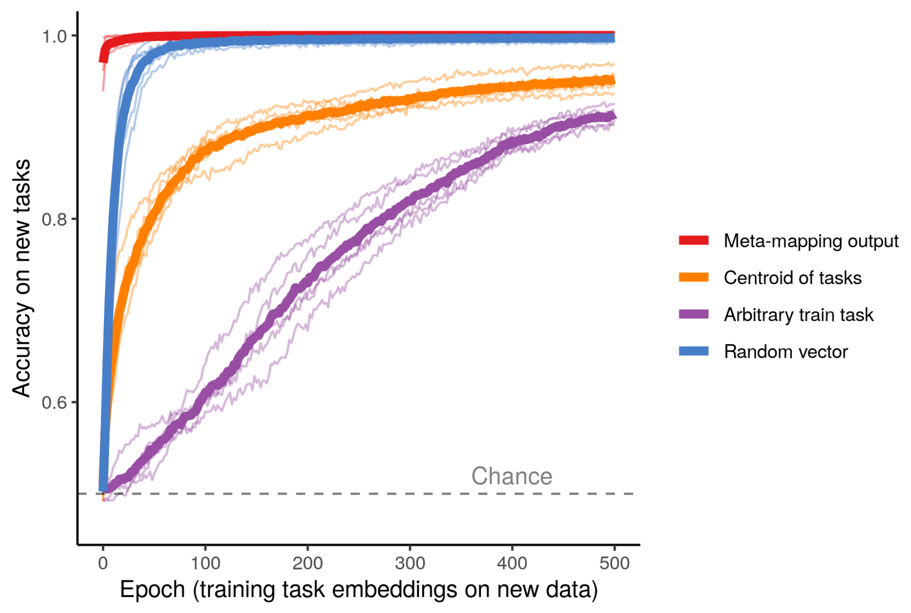


Figure 5.5: Learning curves when optimizing the task embedding for new visual concepts from various starting points. The meta-mapping initialization provides a much better starting-point, and reaches near-optimal performance faster. (Results are from 5 runs, individual runs are shown as light curves.)

5.4 Discussion

In this chapter, we have shown that a zero-shot initialization reduces both the time to learn a novel task, and the mistakes made along the way. We have demonstrated this in both the polynomial regression and visual concept domains. This abilitis is important for multiple reasons.

First, a frequent criticism of deep learning is the idea that it is "data-hungry" (Lake et al., 2017; Marcus, 2018, e.g.). These critiques ignore the success of meta-learning, as we noted in a previous commentary (Hansen et al., 2017). However, as shown in this chapter, zero-shot adaptation provides another perspective on how learning in a novel task can be accelerated. While using deep learning from scratch can be data-hungry, starting from a good task representation output by a meta-mapping might allow deep learning to go on a diet.

Second, starting from a good representation substantially reduces the errors made on the way to mastery. This can be an important goal in its own right, since making errors can be quite costly in settings like robotics, where errors may damage the robot or its surroundings, or even injure bystanders. This has spawned an increasing amount of recent work on "safe exploration" (e.g. Turchetta et al., 2016, 2019). While we have not explored applications to safe exploration in detail, our results suggest that zero-shot adaptation to a novel task might allow for much safer exploration, by reducing the potential errors and allowing the system to explore more productively. This provides

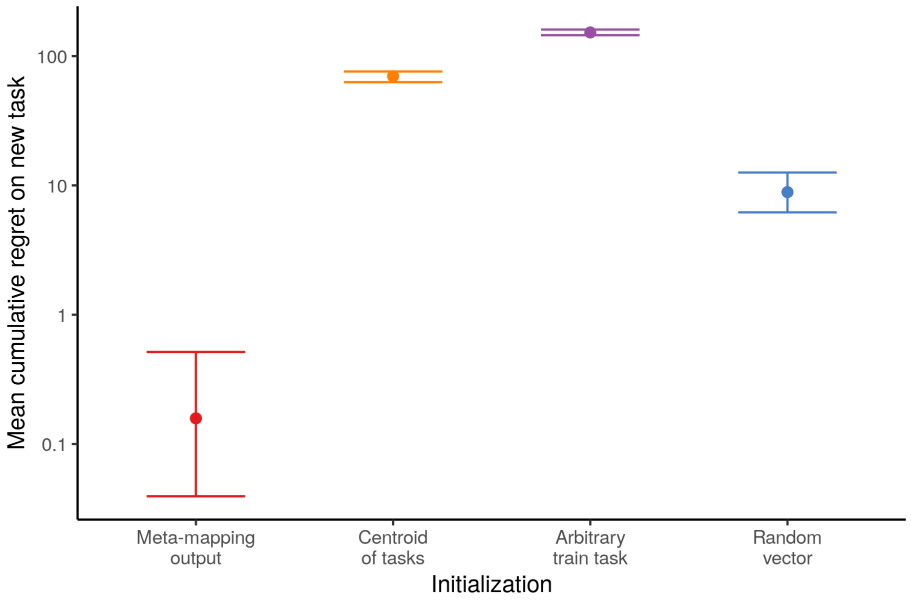


Figure 5.6: Cumulative errors when optimizing the task embedding for held-out visual concepts from various starting points. The meta-mapping initialization substantially reduces cumulative error over the course of learning. (Note that the vertical axis is log-scaled. Error bars are bootstrap 95%-CIs across the 5 runs plotted in Fig. 5.5.)

an exciting direction for future work.

Finally, we have shown that this learning of novel tasks can occur without the possibility of interfering with prior knowledge. In fact, this learning actively requires that prior knowledge, as shown by the comparison to an untrained model. It also relies on a flexible, hyper-network-based architecture, as the optimization is less efficient and appears not to reach a global optimum in a simpler feed-forward architecture.

I suggest that this combination of positive transfer and lack of interference may better reflect human learning than other continual learning paradigms. I suggest that we exhibit *less* interference in domains in which we have more experience. Many continual learning schemes exhibit the opposite pattern of effects. Thus, optimizing task representations in order to learn a new task without intereference could be a useful way to model human learning. It may also offer a useful new approach to continual learning.

However, the present experiments are limited. We have assumed in this chapter that task boundaries and identities are known, but of course there are other settings for continual learning (Ven and Tolias, 2018). One interesting future direction would be to combine our approach with approaches to these settings that try to infer the current task (e.g Nagabandi et al., 2019).

At a higher level, we have assumed a sharp transition between fully-interleaved learning of many tasks, and learning of new tasks without interference by optimizing only the task representations.

However, it is possible that in some cases it would not be possible to optimize the task representation in order to produce optimal behavior on the new task (as it is in the untrained model). For example, this might occur when the model is introduced to a new task from a broader distribution than its training set, as perhaps is the case when a human first encounters a racket sport. To accomodate this, it would be worth exploring a mechanism that would optimize the task embedding to the extent possible, and afterward make minor adjustments to the weights of the model to resolve the remaining error. These later updates might benefit from other continual learning strategies (e.g. Kirkpatrick et al., 2016; Zenke et al., 2017). Exploring these ideas further would be an interesting direction for future research.

Chapter 6

Conclusions & looking ahead

Despite the recent success of deep learning, it still lacks some of the features of human intelligence. In this dissertation, I have focused on how humans are able to reuse our knowledge flexibly in new settings, before we acquire any experience in those settings. I have suggested this flexibility is supported by a computational ability to transform prior task representations to adapt them to a new task. I have proposed meta-mapping – higher-order tasks that transform task representations – as a computational model of this type of adaptation.

In order to evaluate this proposal, I have provided a parsimonious implementation of the meta-mapping framework in the form of Homoiconic Meta-Mapping (HoMM) architectures. I have demonstrated the effectiveness of HoMM by showing its zero-shot task performance across a wide variety of domains, from polynomial regression to visual classification and reinforcement learning. HoMM is often able to achieve 80-90% performance on a new task with no data on that task at all. These results bring deep learning models a step closer to human-like flexibility. My work therefore has implications for both cognitive science and artificial intelligence. In this chapter I will review these contributions, and the broader implications of this project.

6.1 Contributions

I have proposed meta-mappings as a computational account of the human ability to perform a novel task zero-shot (without any data), based on the relationship between the novel task and prior tasks. The fundamental idea is that tasks should be performed from a task representation, and that adaptation can be implemented as a transformation of a task representation. Thus adaptation can be interpreted as a meta-mapping, a higher-order task that maps between representations of more basic tasks.

To instantiate this idea, I have proposed HoMM architectures. These architectures embed data points, tasks, and meta-mappings into a shared representational space, and use shared systems to infer and execute transformations of that space. That is, the same systems are used regardless of the entities (data, tasks, etc.) over which a computation is being executed. Sharing the representational space and transformation systems is parsimonious — it does not multiply networks unnecessarily. Furthermore, I see this proposal as a logical development from the fundamental idea of meta-learning: that tasks themselves can be seen as data points in a higher-order task. This insight leads to the reciprocal idea of transforming task representations just like we manipulate data, i.e. a *homoiconic* approach. This approach is both parsimonious, and outperforms a non-homoiconic approach.

I have shown that meta-mapping, as implemented in the HoMM architecture, performs well across a wide range of settings. The computational paradigms I considered range from regression to classification to reinforcement learning; the inputs range from simple multi-hot vectors to images; and the cues to tasks and meta-mappings range from language to (input, output) tuple examples to (state, action, reward) examples. Across these varied settings, HoMM often achieves 80-90% performance on a new task without data from that task, based on the relationship between the new task and a prior task. When given enough experience with the task space, as in the visual classification settings with enough training tasks, it is able to achieve perfect adaptation. In many runs (though not all), it is even able to do so with held-out meta-mappings.

On the generality of meta-mapping: Generality is a key advantage of the meta-mapping framework. Despite the substantial differences among the computational paradigms I used, the core meta-mapping architectures and approach functioned identically in all domains. This is because the meta-mapping idea only relies on the assumptions that there are task representations, and that adaptation can be represented as a transformation of a representation. Meta-mapping does not require detailed knowledge of the structure of the task space, or the details of the inputs and outputs. These domain specific details can be accomodated by domain-specific systems that would already be necessary for performing the basic tasks (for example, by using convolutional networks for input processing in domains with visual input). This makes meta-mapping relatively easy to apply in new domains.

Language generalization: I compared HoMM to the standard paradigm of zero-shot learning — constructing a task representation from natural language (e.g. Larochelle et al., 2008, also see below). While language-based generalization can be effective, our HoMM approach is generally more sample efficient at generalizing to tasks far outside its training experience, at least in the settings I considered. That is, HoMM needs fewer training tasks to generalize well zero-shot. While the language model performs comparably at interpolating to closely related tasks, as in the visual concepts domain, HoMM appears to offer stronger extrapolation to tasks farther from those on which it has been trained. This effect is demonstrated clearly when the new tasks directly contradict prior tasks, as in the card games and RL domains.

Furthermore, HoMM sometimes seems to exhibit more systematic generalization than the language-conditioned models. HoMM exhibited more strongly correlated performance on the held-out RL tasks — when it was performing well on one of the tasks, it was performing well on the other. It was also better able to apply adaptation learned from tasks with switched colors to switching shapes. These results may reflect the relatively systematic behavior that humans sometimes exhibit.

It will require further exploration to determine with certainty why meta-mapping exhibits greater systematicity in our experiments. One possibility is that task transformation offers a more useful inductive bias than constructing a task representation from language alone in a new setting, and that this is what allows the greater systematicity of our model. That is, task transformation can allows more effective exploitation of prior task knowledge, and more targeted adaptation to the task at hand.

Some notes of caution: However, these results should not be interpreted as suggesting that language is not important or useful. Instead, language and meta-mapping should be seen as complementary. They may be applicable in different domains, and could potentially be mutually supporting. Cognition is multi-faceted, and any single model is guaranteed to be an oversimplified approximation of human cognitive processes in real-world situations. Indeed, an interesting future direction would be to consider how meta-mapping and language can mutually constrain one another when adapting to a new situation. I am not claiming that meta-mapping is the only cognitive mechanism for adaptation. Instead, my results demonstrate that meta-mapping may be useful as one tool for building models with more human-like adaptability.

Adaptation as a starting point: I would also like to highlight the results showing that meta-mapping provides a useful starting point for later learning. While meta-learning approaches often construct a good starting point for learning any task from the known distribution, they do not use task relationships to offer a uniquely valuable starting point for each novel task. My results show that starting from adapting a prior task can substantially reduce the errors made along the way to mastering the new task. The efficiency of human learning may be partly explained by adaptation before beginning the task.

On cognitive modeling: Cognitive modeling is always a trade-off between capturing details of the system and phenomena, and simplifying the system to make it more comprehensible; emergent behavior is generally harder to analyze than the behavior of simpler models. HoMM builds in an intermediate amount of structure, and may therefore offer a useful level of complexity that allows for flexible behavior to be learned (rather than built in) across complex tasks, while also enforcing a structure that makes the representational and computational basis for that flexibility available to analysis.

Summary: HoMM provides a model of a possible computational mechanism underlying cognitive adaptability, and the role that adaptability may play in future learning. Language likely plays an important role as well, and future work should explore uniting these approaches.

6.2 Limitations of the present explorations

While I have explored HoMM in a relatively broad range of computational paradigms, I have still only explored a single family of architectures within a small subset of the computational paradigms that can be found in the literature. In this section I will outline a few limitations of the present work, and corresponding ideas for future research, but note these are simply examples of the many directions in which this project that could be explored further.

Datasets and tasks: First, I have demonstrated HoMM within relatively simple, small domains. The model adapts quite well, but has not achieved the fidelity of adaptation that would be expected from adult humans. I suggest that this result is partly because the model has experienced much less diversity and range of training than humans do by adulthood. Indeed, models that achieve human-level performance are generally trained on datasets several orders of magnitude larger than those I considered here, for example the the million examples in ImageNet (Deng et al., 2009) or the millions of expert replays used to initialize a StarCraft model (Vinyals et al., 2019). While these datasets are quite large, the scale of data that a human has encountered by adulthood should not be underestimated. Eighteen years corresponds to $6 \cdot 10^8$ seconds, which suggests that a lower-bound on the number of unsupervised visual training examples an adult could have had access to would be at least 10^8. Like all deep learning models, HoMM would likely perform better with larger datasets. Furthermore, recent work shows that more realistic environments can improve generalization (Hill et al., 2020). Thus, evaluating meta-mapping approaches in richer, more realistic settings, with training more similar to human experience, will be an important future direction.

One challenge for this future work will be the construction of large sets of tasks, along with identifying the systematic relationships between them necessary for meta-mapping. The need to annotate relationships among tasks is another limitation of the meta-mapping approach. While large meta-learning datasets do exist, they do not generally contain relationships among the tasks (and often lack any interesting structure on which such relationships could be constructed). However, it is possible that in regimes with larger amounts of data, looser analogies between the tasks would be acceptable training data (just as humans don't need a perfect isomorphism to infer an analogy). This possibility will need to be explored in future work as well.

Meta-mapping: In addition, the flexibility offered by meta-mappings is still limited in some crucial ways. It requires exactly identifying the prior task to use as a source for the adaptation, and when that adaptation should occur (i.e. where the task boundary is). It would be more realistic to relax these assumptions. Approaches to tracking task-changes have been proposed in domains like meta-learning (e.g. Nagabandi et al., 2019), and could likely be adapted to this setting. Exploring this would be an interesting direction for future research.

HoMM architectures and algorithms: There are a number of architectural and algorithmic aspects of the approach that could potentially be altered. Some of these are discussed in Sec. 2.4.1. Here, I highlight only the broad observations that cognitive processing is much more complex than our model. While I relied on simple feed-forward computations for our simple tasks, using a recurrent task network, or even a more complex architecture involving external memory — such as the Differentiable Neural Computer (Graves et al., 2016) — would likely increase the ability of the model to perform and adapt on complex tasks. More complex processing would likely be necessary to reach human-level performance in many domains.

Reliance on human choices: Meta-mapping (like almost all machine learning approaches) is fundamentally restricted to computational paths that are carefully chosen by the humans who implement it. The model cannot decide when it is appropriate to meta-map, it is forced to do so when I thought it was appropriate. It would be interesting to explore whether a model could learn when meta-mapping was appropriate in an end-to-end manner given appropriate capabilities and input. That is, if a model had the *capacity*, at any point in time, to transform prior task representations to adapt the present situation, could it learn when it *should* do so? There are a number of linguistic cues that humans can exploit for when they should try to adapt, and a system that experiences many such transitions might learn when adaptation is warranted. This could potentially bring the flexibility of the model closer to that of humans.

Summary: There are a number of limitations to the present work, and to many modern deep learning models, that should be explored in greater detail in the future. However, I believe this work also offers some useful perspective on various issues within the fields of cognitive science and artificial intelligence, which I will discuss in the subsequent sections.

6.3 On flexibility in natural and artificial intelligence

In the introduction, I noted how researchers in cognitive science have critiqued deep learning for its lack of flexibility (e.g. Lake et al., 2015, 2017; Lake and Baroni, 2018; Marcus, 2018). I have addressed one challenging aspect of flexibility in this work – the ability to take our knowledge of a task, and adapt to some variation. While this adaptation might be challenging for standard deep-learning models, I have shown that the general framework of meta-mapping makes it possible. Thus, at their most basic level, my results present a challenge for those who would say deep-learning models are too inflexible to be accurate cognitive models.

Indeed, I see my project as following in the tradition of work that explores how systematic, structured generalization can emerge from the structure of learning experience, without needing to be built into the model itself (McClelland, 2010; McClelland et al., 2010; Hansen et al., 2017). This tradition is a challenge to arguments that cognition must rely on strictly compositional representations in order to exhibit systematic and productive generalization (e.g. Fodor, 2001a, 2008; Lake and Baroni, 2018). Without building in compositional representations of tasks, our model can learn to exploit the shared structure in the concept of "losing" across a few card games to achieve 85% performance in losing a game it has never tried to lose before. Similarly, it can achieve perfect adaptation to held-out visual concepts via trained meta-mappings, and near-perfect adaptation from held-out meta-mappings. Hard-coded compositional structure does not appear necessary to achieve good adaptation.

Furthermore, there are a number of potential benefits to letting the compositional structure emerge rather than building it in. First, the structure does not need to be hand-engineered specially for each domain. Our system required no special knowledge about the domains beyond the basic tasks and the relationships between them. The fact that some of these relationships corresponded to e.g. permutations of variables in the polynomial domain did not need to be hard-coded, instead the model was able to discover it from the patterns of the mappings (as indexed by its ability to generalize well to held-out permutations).

The second advantage of letting compositionality emerge is that it can potentially allow for novel decompositions at test time. The ability of our model to perform well on held-out meta-mappings supports this hope. Furthermore, the ability of the model to extrapolate a meta-mapping learned on color tasks to shape tasks in the RL domain provides further promising evidence. These results are suggestive of the ability of HoMM to extrapolate beyond what it has experienced with flexibility and systematicity closer to that of human cognition.

In summary, I suggest that meta-mapping offers a way to create deep learning models with flexibility slightly closer to that of the human mind. While I have demonstrated these results in some simple settings, one of the powerful features of deep learning is that its results tend to improve as datasets grow more complex and realistic (Hill et al., 2020; Radford et al., 2019; Sutton, 2019). I hope that this research will help guide the way to building even more flexible models in more realistic domains.

6.4 Relating to cognitive science

Our work provides a tool for modeling human adaptability, which has many potential direct applications. It offers an explanation for how humans might be able to adapt, albeit imperfectly, when told "watch out, the floor is slippery," or recognize a pink-and-green striped car even if they have never seen one before, by transforming their task representations based on prior experience. This adaptability is a fundamental aspect of human intelligence, but is often omitted from cognitive models. However, our work has broader relevance as well.

Fast and slow transfer: In Chapter 1, I reviewed the cognitive science and machine learning literatures from a Complementary Learning Systems perspective. In particular, I sugested that humans' slow learning of shared structure in the world can itself provide transfer benefits, but also helps set up the representations necessary for faster transfer mechanisms. This idea is reflected in the organization of the HoMM architecture. A great deal of perceptual and action processing is shared across tasks (though see below), so that the model can exploit the shared visual features of different games or objects. The representations constructed by these shared systems are used for both task inference and for task performance. This approach allows the system to perform a novel task from a few examples (as in standard meta-learning), or based solely on its relationship to prior tasks, by meta-mapping. The fast zero-shot transfer achieved by meta-mapping thus relies on the representations of tasks and data that are constructed over the full development of the network.

Furthermore, this fast transfer ability itself must be learned over time. However, once it has been learned, it can then generalize to new examples and even new meta-mappings. This approach reflects my suggestion that humans not only learn good representations for fast transfer, but actively practice the act of adaptation. I hope my work will inspire broader thought about how different systems of transfer, operating over different timescales, can support each other in order to achieve the flexibility of human intelligence.

Abstraction & recursion: Abstraction & recursion offer one exciting area where our model could potentially offer a new modeling framework. It would be interesting to explore how concepts can be recursively built upon other concepts, as happens in learning of mathematics (Wilensky, 1991; Hazzan, 1999; Lampinen and McClelland, 2018). For example, addition can be seen as repeated succession, multiplication can be seen as repeated addition, exponentiation as repeated multiplication, and this process is recursively continued in up-arrow notation. A homoiconic system like HoMM may take us a step closer to capturing this recursive construction of concepts. It would be interesting to explore how our architectures could model this type of recursive construction of concepts.

Relatedly, I believe that my model moves closer to capturing some of the recursive processing that Fodor and others have considered to be important (e.g. Fodor, 2008). I have drawn particular inspiration from the idea that humans re-represent our knowledge into more generalizable forms (Karmiloff-Smith, 1986; Clark and Karmiloff-Smith, 1993). Karmiloff-Smith examines fascinating developmental trajectories where, even after initial behavioral mastery of some concept is achieved, various implicit and explicit measures of understanding continue to evolve. How could this process, which Karmiloff-Smith calls representational redescription, actually work? It would be interesting to explore whether auxiliary learning objectives over task representations and meta-mappings, and the shaping effects of language (see below), could model some of these phenomena. Could the change in behavior on a particular task be driven by the evolution of its task representation while learning a meta-mapping involving that task? Could unsupervised learning over task representations make clusters and structure within the space of tasks salient, thereby regularizing and structuring behavior? Exploring these questions will be an exciting direction for future work.

Consciousness: I have also been inspired by computational models of how conscious knowledge may be built on top of implicit knowledge (Cleeremans, 2014), as well as by the Global Workspace Theory (Baars, 2005). HoMM's shared representational space for data points, tasks, and meta-mappings can be seen as a global workspace, over which task-specific computations can be executed. Indeed, the HoMM architecture could potentially shed light on issues about explicit vs. implicit knowledge — it is plausible to assume that a task representation captures what we know explicitly about a task, while implicit knowledge could be captured both by the task representation and elsewhere (such as in the default weights of the task network). Exploring these connections further could provide another exciting direction for future work.

Modularity vs. generality: The issues above also relate to ideas Fodor expressed about the architecture and computations of the mind, for example the view of mental processes as "transforming internal representations" and that what is accessible about the stimulus is only "what is given in [...] its *proximal* representations" (Fodor, 1975, pp. 200-201). The division of the HoMM architecture into input and output systems, with flexible, task-specific computations in the middle may seem very reminiscient of the type of modular architecture that he sometimes advocated (Fodor, 1983). However, I chose this implementation as a simplifying assumption — I believe that in reality processes such as perception are not completely task-independent, but involve the interaction of top-down and bottom-up constraints (McClelland et al., 2014).

Reciprocally, I also believe that higher-level computations are influenced and constrained by the modalities in which they are supported. This computational feature can emerge in the HoMM model, as despite the fact that different types of data and tasks are embedded in a shared latent space, the model generally learns to organize distinct types of inputs into somewhat distinct regions of this space. This organization means that the task-specific processing can potentially usefully exploit domain-specific features of the input, as for example humans do when they use gestures to think and learn in spatial contexts like mathematical reasoning (Goldin-Meadow, 1999; Wakefield et al., 2018). At the same time, the shared space can allow a graded overlap in the structure that is shared across different input domains. That is, the HoMM model is able to learn what should be shared and what should be separated, whereas approaches that build in such divisions are fundamentally more limited. HoMM has the desirable property that "modularity may not be built in [but] may result from the relationship among representations" (Tanenhaus and Lucas, 1987).

Language: I noted above that our results should not be taken as a rejection of the role of language. Instead, they suggest that meta-mapping and language could be mutually supporting. It would be interesting to explore whether combining the representations produced by the language system and meta-mapping system could result in better performance than either alone, especially if this combination were weighted by some measure of uncertainty in the estimates. Furthermore, while I only considered language as an input in the present work, language output (explaining behavior) can play an important role in learning, both in humans (Chi et al., 1994), and in neural networks (Mu et al., 2019). While my use of task-representation-classification in some settings may have captured some aspects of this structuring, it did not appear to have a substantial effect. Requiring the model to produce richer explanations during learning will likely be important for achieving truly human-like representations and behavior. Architectures like the one I proposed could provide a basis for exploring the interactions between language and reasoning over development, and how the connection between language and thought "originates, changes, and grows" (Vygotsky, 1934).

Cognitive control: Although it is not my primary focus in this project, the HoMM architecture may have interesting connections to cognitive control. Even without meta-mapping, HoMM instantiates an architecture that can compute flexibly in response to task demands, provided as examples or natural language. Furthermore, the "default" task-network weights output by the HyperNetwork could be used to model more automatic processing, which more cognitive, task-specific processing might need to override. I showed some initial experiments related to this in Chapter 2. Meta-mapping adds many additional potential connections to control — for example, a failure to meta-map perfectly could capture some of the challenges of task-switching. Exploring these ideas further would also be an interesting direction for future work.

Neuroscience: Finally, a major advantage of neural network models is their ability to make predictions about neuroscience. For example, neural network models have been used to understand aspects of the neural basis for perception (Yamins and Dicarlo, 2016), semantic cognition (Rogers and McClelland, 2004), and cognitive control (Shenhav et al., 2013). I have not engaged with this level of analysis, but doing so would be an exciting direction for future work. Our HoMM architecture offers a framework that can unify perception, task representation, control, and decision/action, all within a single model. It would be possible to relate the different components of our model to different brain regions — visual perception to visual cortex, higher level perceptual features to more semantic regions, the action network to motor cortex, and the meta/hyper/task networks to frontal regions associated with task representation/control/working memory. Thus, our architecture could potentially provide an integrative model spanning a wide range of brain regions, although it would likely require some modifications to account for neural data well (some of which are discussed in the limitations section above).

Summary: I take an emergent perspective on the structure of the mind, and believe that all cognitive and perceptual processes are mutually influencing and supporting. For simplicity and clarity my model does not always fully reflect these perspectives. Furthermore, I believe my approach may be broadly useful, even to researchers with different perspectives. The functional approach relates to the ideas of Fodor and Karmiloff-Smith, the perspective on adaptation draws inspiration from prior work on analogy and transfer, and the HoMM architecture could even have interesting implications for researchers interested in cognitive control or neuroscience. I hope that researchers from many of these areas will find my work to provide a useful perspective when addressing these issues.

6.5 Relating to artificial intelligence

There are a number of potential direct applications in artificial intelligence, from building more flexible vision models to building better systems for robotics. Domains like robotics are especially interesting from the meta-mapping perspective, because exploration in real world settings is costly and must be safe (Turchetta et al., 2016), and so the substantial reduction in errors made when using meta-mapping as the starting point for learning a new task may be valuable.

Reinforcement learning: Applying meta-mapping to different types of adaptation in RL also opens many possibilites, especially in combination with model-based methods. Meta-mapping could be used as a principled way of adapting transition functions or successor representations (c.f. Madarasz and Behrens, 2019), beyond the approach of adapting model-free reward or value estimates that I demonstrated. While adapting pure model-free RL will likely be challenging in more complex task spaces, combining meta-mapping with other insights could yield much greater flexibility. For example, meta-mapping could be used with hierarchical models where language has been used as a task or sub-task representation (e.g. Jiang et al., 2019). Similarly, it could be applied in planning based models, for example using monte-carlo tree search (as in e.g. Silver et al., 2016, 2017), but with task-representation-conditioned policy and value functions. More ambitiously, meta-mapping could be explored in models that learn to plan (Guez et al., 2019) rather than having that planning hand-engineered into the architecture. Many contemporary RL frameworks could potentially be augmented with meta-mapping.

Abstraction: Many of the directions of future investigation from a cognitive perspective relate to pressing problems in artificial intelligence as well. The issue of flexible abstraction is challenging in deep learning — while feed-forward neural networks generally construct progressively more abstract representations in higher layers, the relationships between those representations are fixed by the fixed computational pathway. The shared representational space and meta/hyper networks in our model provide a suggestion for how concepts at different levels of abstraction could be integrated and used more flexibly. A model that demonstrated this ability would be another important step towards human-like flexibility.

Continual learning: The work in Chapter 5 suggests new directions in continual learning. By off-loading much of the task-specific computation to a flexible hyper-network-based architecture, and inferring and optimizing a task representation, we can enable learning of a new task without even the possibility of interfering with prior tasks. Furthermore, we can leverage our knowledge of prior tasks to learn faster than we would have in an untrained model, or without meta-mapping. This positive transfer is very different from the standard in continual learning, which mostly focuses on stemming the catastrophic loss of accuracy on prior tasks caused by learning new tasks, even in more recent works that have also incorporated hyper networks (Oswald et al., 2020). Given the importance of learning rapidly on new tasks, without interfering with prior knowledge, this approach to continual learning is also an exciting future direction.

Hyper networks for multiple task domains: As machine-learning research has begun to focus on multi-task- and meta-learning, an increasing number of approaches have been developed which construct a task representation (Hermann et al., 2017; Zintgraf et al., 2018; Rusu et al., 2019), although not all those works explicitly identify the task representation as such. Some of these works, such as Rusu et al. (2019), use the task representation to parameterize a task-specific-network, as in HoMM. However, most simply concatenate the task representation to other network inputs. I compared both of these architectural approaches, and found that while both performed well on the basic tasks, the hyper-network architecture performed better at meta-mapping, and allowed for better optimization of task representations. This raises the possibility that hyper-network approaches may be better able to accomodate performing qualitatively different *types* of tasks in a shared architecture. It would be interesting to investigate this as multi-task learning moves beyond single-domain task distributions (such as a collection of vision tasks) to the generality of task domains that a human might encounter in a day (vision, language, control, and so on).

Summary: In addition to the direct applications of the meta-mapping framework in building more flexible artificial intelligence systems, it suggests many exciting future directions in reinforcement learning, abstraction, multi-task learning, and continual learning. I hope that this project will help inspire the development of deep learning systems that can adapt and learn more like humans.

6.6 Looking ahead

The next ten years will likely bring greater clarity about how far deep learning is from achieveing human-like intelligence. As I suggested in the introduction, it may be that greater scale and complexity of our architectures and training regimes will bring forth more flexible behavior from many deep neural network models. Indeed, flexibility will almost certainly increase to some extent — overparameterization and larger datasets both tend to improve the generalization performance of deep neural networks. If translation abilities can emerge from a word-prediction model given enough data (Radford et al., 2019; Brown et al., 2020), couldn't the ability to adapt emerge from simple architectures trained in complex enough environments? It is not yet clear. What will the contribution of this dissertation be to our future knowledge? Will it be more than another "bitter lesson" demonstrating that "building in how we think we think does not work in the long run" (Sutton, 2019)?

From an artificial intelligence perspective, even if more flexible behavior emerges in other architectures when they are placed in richer training regimes, the architectural and training innovations I have proposed in this work may still provide useful insights that allow flexibility to emerge at more feasible data scales. It is not clear how much data would be required to learn human-like flexibility by brute force, or whether it is even possible at all. Many applications of artificial intelligence (from medical diagnosis to self-driving cars) would benefit from models with greater flexibility within a family of closely related tasks. The approach I have proposed may be useful in such cases. The perspective of building and transforming task representations may also inspire future work that leverages similar abstractions — perhaps architectures that use task-representation and HyperNetwork-based approaches like ours will be better able to accommodate diverse sets of tasks from different domains, or will be useful for challenging new evaluations of intelligence (e.g. Chollet, 2019).

From a cognitive perspective, my work attempts to provide a computational basis for understanding the flexibility of the human mind. The HoMM approach allows for learning of flexible adaptation in complex task settings, within a framework that makes this behavior interpretable. Thus, HoMM may provide a useful tool for understanding human adaptation, that could be applied to many cognitive tasks. My work may also provide a framework for instantiating theories about the neural basis of higher-level cognitive processing.

Ultimately, I have presented one computational perspective on how natural and artificial intelligence could flexibly adapt to new situations. I am excited to see the new perspectives the future will bring, and I hope my work will provide some inspiration for some of them.

Appendix A

Model details & hyperparameters for all experiments

A.1 Model & hyperparameters

In Fig. A.1, we show the flow of inference (forward) and gradients (backward) through the HoMM architecture on basic task and meta-mapping training steps.

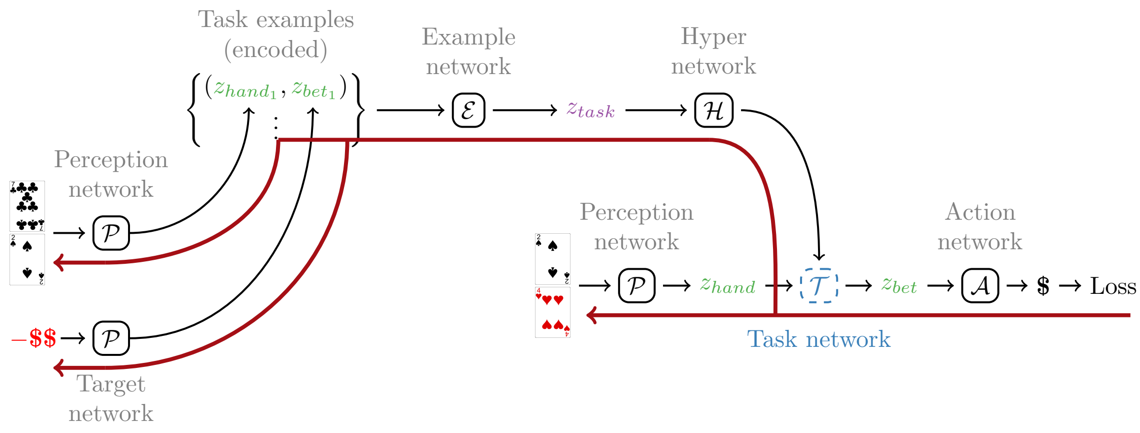
See table A.1 for detailed architectural description and hyperparameters for each experiment. Hyperparameters were generally found by a heuristic search, where mostly only the optimizer, learning rate annealing schedule, and number of training epochs were varied. Some of the parameters take the values they do for fairly arbitrary reasons, e.g. the polynomial experiments were run earlier, before 1-layer task networks were found to be useful in some settings. While it would be ideal to fully search the space of parameters for all models, unfortunately our computational resource limitations prohibited it. Thus the results in the paper should be interpreted as a lower bound on what would be possible.

Each epoch consisted of a separate learning step on each task (both base and meta), in a random order. In each task, the meta-learner would receive only a subset (the "batch size" above) of the examples to generate a function embedding, and would have to generalize to the remainder of the examples in the dataset. The embeddings of the basic tasks used for meta-mappings were computed and cached once per epoch, so as the network learned over the course of the epoch, these task-embeddings would get "stale," but this did not seem to be too detrimental. In the case of the RL tasks, where there were persistent task embeddings, they were used insteadd.

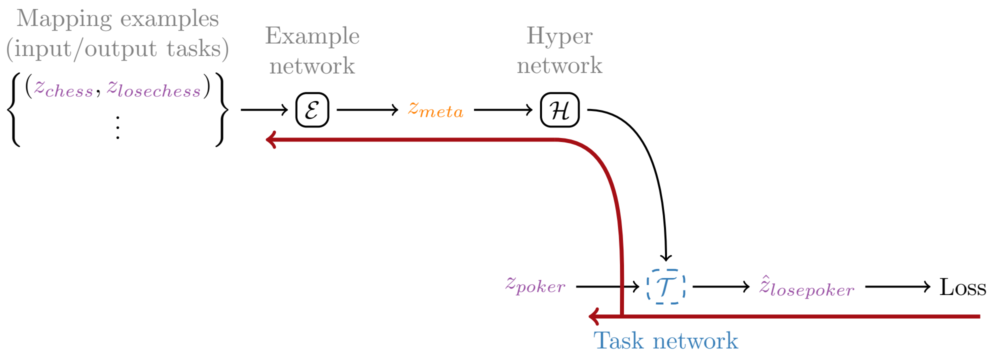
The results reported in the figures in this paper are averages across multiple runs, with different trained and held-out tasks (in the polynomial and visual concepts cases) and different network initializations and training orders each epoch (in all cases), to ensure the robustness of the findings.

	Polynomials	Cards	Visual	RL
$\mathcal{Z}$-dimension	512	512	512	512
$\mathcal{I}$ num. layers	2	2	2	2
$\mathcal{I}$ num. hidden units	128	128	128	128
$\mathcal{I}$ conv. layers. (num filters, size, all strides are 2)	-	-	(64, 5), (128, 4), (256, 4), (512, 2), max pool	(64, 7), (64, 4), (64, 3)
$\mathcal{L}$ architecture	-	2-layer LSTM + 2 fully-connected	2-layer LSTM + 2 fully-connected	2-layer LSTM + 2 fully-connected
$\mathcal{L}$ num. hidden units	-	512	512	512
$\mathcal{T}$ num. layers	1	3	1	3
$\mathcal{T}$ num. hidden units	-	128	-	128
$\mathcal{E}$ architecture	2 layers per-datum, max pool across, 2 layers	2 layers per-datum, max pool across, 2 layers	2 layers per-datum, max pool across, 2 layers	2 layers per-datum, max pool across, 2 layers
$\mathcal{H}$ architecture	4 layers	4 layers	4 layers	4 layers
$\mathcal{E}$ num. hidden units	512	512	512	1024
$\mathcal{H}$ num. hidden units	512	512	512	512
Task, MM representations from	Examples	Examples	Language	Examples
$\mathcal{F}$ num. layers	3	1	HoMM: 1, Lang: 3	3
$\mathcal{F}$ num. hidden units	64	64	64	128
$\mathcal{F}$ init scale	1	1	30	10
$\mathcal{F}$ weight norm. (Salimans and Kingma, 2016)	No	No	No	Yes
$\mathcal{A}$ num. layers	1	1	2	1
$\mathcal{A}$ num. hidden units	-	-	128	-
Nonlinearities	Leaky ReLU most places, except no non-linearity at final layer of networks outputting to Z, sigmoid for classification outputs, and softmax over actions.	Leaky ReLU most places, except no non-linearity at final layer of networks outputting to Z, sigmoid for classification outputs, and softmax over actions.	Leaky ReLU most places, except no non-linearity at final layer of networks outputting to Z, sigmoid for classification outputs, and softmax over actions.	Leaky ReLU most places, except no non-linearity at final layer of networks outputting to Z, sigmoid for classification outputs, and softmax over actions.
Base task loss	ℓ_2	ℓ_2 (masked)	Cross-entropy	ℓ_2 (masked)
Meta-mapping loss	ℓ_2	ℓ_2	ℓ_2	ℓ_2
Partially-persistent task embeddings	No	No	No	Yes
Persistent embedding match loss weight	-	-	-	0.2
Optimizer	Adam	RMSProp	RMSProp	RMSProp
Learning rate (base)	$3 \cdot 10^{-5}$	$1 \cdot 10^{-5}$	$3 \cdot 10^{-5}$	$1 \cdot 10^{-4}$
Learning rate (meta)	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-4}$
L.R. decay rate (base)	$\times 0.85$	$\times 0.85$	$\times 0.8$	$\times 0.8$
L.R. decay rate (meta)	$\times 0.85$	$\times 0.9$	$\times 0.85$	$\times 0.95$
L.R. min (base)	$3 \cdot 10^{-8}$	$3 \cdot 10^{-8}$	$1 \cdot 10^{-8}$	$3 \cdot 10^{-8}$
L.R. min (meta)	$1 \cdot 10^{-7}$	$3 \cdot 10^{-8}$	$1 \cdot 10^{-8}$	$3 \cdot 10^{-7}$
L.R. decays every	100 epochs	200 epochs	400 epochs	10000
Num. training epochs	5000	100000 (optimally stopped)	10000 for 4 train mappings, 7500 for 8, 5000 for others	300000 (optimally stopped)
Num. runs	5	5	10	5
Num. base tasks (training)	1300 ($= 60 + 60 \times 20 + 40$)	36	Varies	18
Num. base tasks (held out for MM eval)	800 ($= 40 \times 20$)	4	Varies	2
Num. meta classifications	6	8	8	-
Num. train MMs	20	3	Varies	1
Num. held-out MMs	16	0	2	0
Base dataset size	1024	1024	336	64
Base examples size	50	768	-	32
Meta dataset size (train)	60	36	Varies	18
Meta examples (train)	Half of train dataset	Half of train dataset	-	Half of train dataset
Meta examples (eval)	All of train dataset	All of train dataset	-	All of train dataset
Base datasets refreshed	Every 50 epochs	Every 50 epochs	Every 20	Every 1500
Target network updated	-	-	-	Every 10000 epochs
RL discount	-	-	-	0.85
RL explore prob. (ϵ)	-	-	-	Init: 1, decay: -0.03
Action softmax β	-	8	-	8

Table A.1: Detailed hyperparameter specification for different experiments. A "-" indicates a parameter that does not apply to that experiment. As a reminder: the shared representational space is denoted by Z. Input encoder: $\mathcal{I}$: input $\rightarrow Z$. Action decoder $\mathcal{A} : Z \rightarrow$ output. Target encoder $\mathcal{T}$: targets $\rightarrow Z$. Meta-network $\mathcal{E} : \{(Z, Z), ...\} \rightarrow Z$ maps examples to a task representation. Hyper-network $\mathcal{H} : Z \rightarrow$ parameters. Task network $F : Z \rightarrow Z$ is parameterized by $\mathcal{H}$. Language encoder: $\mathcal{L}$: language $\rightarrow Z$.



(a) Basic task inference/training (from examples).



(b) Meta-mapping inference/training (from examples).

Figure A.1: Schematic of architecture, showing inference and gradient flow through the model on a training step. Thin black lines moving rightward represent inference, thick red lines moving leftward represent gradients. (a) Inference and gradients for the basic tasks. (a) Inference and gradients for meta-mappings. The gradients end at the examples of the meta-mapping, rather than propagating through to alter how those representations are constructed, due to GPU memory constraints. In the future, it might be useful to explore whether allowing further propagation would improve results for both basic tasks and meta-mappings. (These figures depict the inference/gradient flow when performing tasks and meta-mappings from examples, performing from language is similar, except that the example inputs and example network are replaced with language inputs and the language processing network.)

A.1.1 Clarifying meta-mapping: a definitional note

When we discussed meta-mappings in the main text, we equivocated between tasks and behaviors for the sake of brevity. For a perfect model, this is somewhat justifiable, because each task will have a corresponding optimal behavior, and the sytem's embedding of the task will be precisely the embedding which produces this optimal behavior. However, behavior-irrelevant details of the task, like the color of the board, may not be embedded, so this should not really be thought of as a task-to-task mapping. This problem is exacerbated when the system is imperfect, e.g. during learning. It is thus more precise to distinguish between a ground-truth meta-mapping, which maps tasks to tasks, and the computational approach to achieving that meta-mapping, which really maps between representations which combine both task and behavior.

A.2 Source repositories

The full code for the experiments and analyses can be found on github:

- HoMM library: `https://github.com/lampinen/HoMM`
- Polynomials: `https://github.com/lampinen/HoMM_polynomial_analysis`
- Cards (models): `https://github.com/lampinen/HoMM_cards`
- Cards (human experiment): `https://github.com/lampinen/cards_for_humans`
- Concepts: `https://github.com/lampinen/categorization_HoMM`
- RL: `https://github.com/lampinen/HoMM_grids`
- Stroop results (below): `https://github.com/lampinen/stroop`

Appendix B

Supplemental material for Chapter 2

B.1 Details of polynomial task domain

We randomly sampled the train and test polynomials as follows:

1. Sample the number of relevant variables (k) uniformly at random from 0 (i.e. a constant) to the total number of variables.
2. Sample the subset of k variables that are relevant from all the variables.
3. For each term combining the relevant variables (including the intercept), include the term with probability 0.5. If so give it a random coefficient drawn from $\mathcal{N}(0, 2.5)$.

The data points on which these polynomials were evaluated were sampled uniformly from $[-1, 1]$ independently for each variable, and for each polynomial. The datasets were resampled every 50 epochs of training.

Meta-tasks: For meta-tasks, we trained the network on 6 task-embedding classification tasks:

- Classifying polynomials as constant/non-constant.
- Classifying polynomials as zero/non-zero intercept.
- For each variable, identifying whether that variable was relevant to the polynomial.

We trained on 20 meta-mapping tasks, and held out 16 related meta-mappings.

- Squaring polynomials (where applicable).
- Adding a constant (trained: -3, -1, 1, 3, held-out: 2, -2).

- Multiplying by a constant (trained: -3, -1, 3, held-out: 2, -2).
- Permuting inputs (trained: 1320, 1302, 3201, 2103, 3102, 0132, 2031, 3210, 2301, 1203, 1023, 2310, held-out: 0312, 0213, 0321, 3012, 1230, 1032, 3021, 0231, 0123, 3120, 2130, 2013).

B.2 Architecture & training experiments

In this section we consider a few variations of the architecture and training, to justify the choices made in the paper.

B.2.1 Inadequacy of vector analogies for meta-mapping

One possible implementation of meta-mapping would be to just construct an analogy vector and use that for the mapping. This idea is motivated by work showing that word vector representations often support vector analogical reasoning; for example if we denote the vector for the word king as $\vec{v}_{king}$, relationships like $\vec{v}_{queen} \approx \vec{v}_{king} + (\vec{v}_{man} - \vec{v}_{woman})$ often hold (Mikolov et al., 2013). Thus, adopting a similar strategy for meta-mapping would be superficially plausible. For example, in the polynomials domain, the meta-mapping "Permute (w, z, x, y)" could be estimated by taking the vector differences between the representations of inputs and targets, computing an average difference vector, and adding that to the held-out examples to produce an output for each one.

However, in this section, we prove that such an approach cannot accurately represent all the meta-mappings in the polynomials domain. Furthermore, we sketch a proof by construction that a linear task network (i.e. an affine transformation, matrix multiplication plus a bias vector) parameterized independently for each meta-mapping suffices.

Proof that vector analogies are inadequate: In essence, the proof is simply that many of our meta-mappings are non-commutative, while vector addition is commutative. Consider the mappings for adding 1 to a polynomial, and multiplying by 2. Assume there were vector representations for these mappings, respectively $\vec{m}_{+1}$ and $\vec{m}_{\times 2}$. Let $\vec{f}_x$ be the vector representation for the polynomial $f(w, x, y, z) = x$. Then $\vec{f}_x + \vec{m}_{+1} = \vec{f}_{x+1}$, $\vec{f}_x + \vec{m}_{\times 2} = \vec{f}_{2x}$. But then:

$$\vec{f}_{2(x+1)} = \left(\vec{f}_x + \vec{m}_{+1}\right) + \vec{m}_{\times 2} = \vec{f}_x + \vec{m}_{+1} + \vec{m}_{\times 2} = \left(\vec{f}_x + \vec{m}_{\times 2}\right) + \vec{m}_{+1} = \vec{f}_{2x+1}$$

Thus such a representation would result in contradictions, such as $2x + 1 = 2x + 2$. Similar issues occur for permutation and other non-commutative mappings.

Proof sketch that affine transformations in an appropriate vector space suffice: Suppose that we have a vector representation for the polynomials, where there is a basis dimension corresponding to each monomial, so that the polynomial can be represented as a vector of its coefficients. (This is the standard vector-space representation for polynomials.) Then permutation corresponds to permuting these monomials, i.e. a permutation of the basis dimensions, which is a linear transformation. Adding a constant corresponds to adding to one dimension, which requires only the vector addition part of the affine transformation. Multiplying by a constant requires multiplying each dimension, i.e. a block-diagonal linear transformation.

Squaring polynomials is slightly more complex, and requires augmenting the vector space with components whose values are the product of the coefficients of each pair of monomials. In this case, squaring corresponds to a simple linear transformation. However, this augmentation makes the other meta-mappings more complex. The most difficult case is adding a constant, which requires shifting each pair term containing a constant by the product of the constant and the coefficient of the other monomial, but this again reduces to simply an appropriately parameterized affine transformation — each pair term containing a constant term simply needs the added constant as a weight times the component for the other monomial. Thus affine transformations suffice in this setting.

Of course, with a sufficiently complex, deep, recurrent, and non-linear task network, any meta-mapping could be computed in principle, since a sufficiently complex network is Turing-complete (Siegelman and Sontag, 1992). Thus, our approach to meta-mapping is fully general, conditioned on a sufficiently complex task network, while simpler approaches may not be.

B.2.2 Basic meta-learning in the polynomials domain

In Fig. B.1, we show that the basic meta-learning is working well in the cards domain. That is, we show that after the example network is presented with a set of example input, output pairs, the system is generalizing well to other points from that polynomial. At the end of training, the mean loss on trained polynomials is 0.025 (bootstrap 95%-CI [0.02, 0.03]), and for held-out polynomials it is 0.58 (bootstrap 95%-CI [0.45, 0.70]). Since chance loss is 11.76 for the trained polynomials, and 11.10 for the eval, this corresponds to about 99.8% of optimal on the trained polynomials, and 94.8% on the held-out.

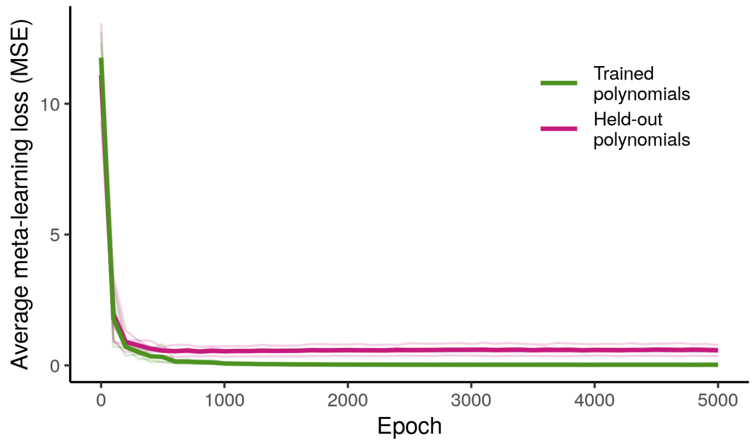


Figure B.1: Basic meta-learning performance in the polynomials domain over learning. The system is generalizing at the meta-learning level. That is, this graph shows that, after the example network receives a set of (input, output) example tuples, it is generating a sufficiently good representation to regress held-out points from that polynomial. This is true both for polynomials it was trained with (green), and for polynomials that are held-out and never encountered during training (pink). (Thick lines are averages over 5 runs, shown as thin light curves.)

B.2.3 Hyper network vs. conditioned task network

Instead of having the task network F parameterized by the hyper network $\mathcal{H}$, we could simply have a task network with learned weights which takes a task embedding as another input. In Fig. B.2, we show that this architecture fails to learn the meta-mapping tasks, although it can successfully perform the basic tasks. We suggest that this is because it is harder for this architecture to prevent interference between the comparatively larger number of basic tasks and the smaller number of meta-tasks. While it might be possible to succeed with this architecture, it was more difficult in the hyper-parameter space we searched. See also Fig. C.4, where we show that both architectures perform similarly for language generalization.

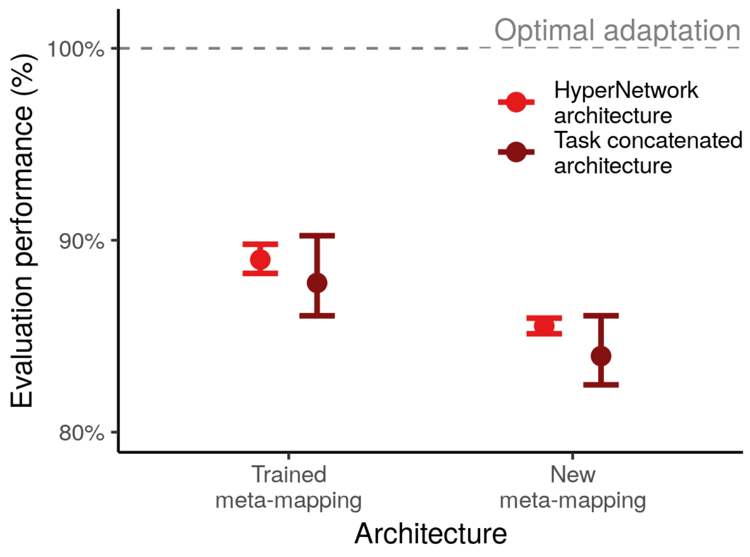


Figure B.2: The HyperNetwork-based architecture we propose in the main text performs better and more consistently on meta-mappings than a simpler architecture that simply concatenates a task representation to the input before passing it through a fixed MLP. Results are in the polynomial domain, c.f. Fig. 2.4. Note that the task-concatenated architecture performs just as well at the trained basic tasks (not shown), it is adapting via meta-mappings that proves challenging for it. See Supp. Fig. D.2 for a more dramatic comparison in the RL domain.

B.2.4 Meta-classification lesion

In Fig. B.3, we show that meta-classification training is not beneficial in the polynomials domain. Specifically, on trained meta-mappings the HoMM model is achieving a normalized performance of 88.99% (bootstrap 95%-CI [88.20, 89.98]), while without meta-classification it is achieving a normalized performance of 89.7% (bootstrap 95%-CI [88.87, 90.61]). On new meta-mappings the HoMM model is achieving a normalized performance of 85.54% (bootstrap 95%-CI [85.14, 85.94]), while without meta-classification it is achieving a normalized performance of 86.29% (bootstrap 95%-CI [85.54, 86.79]). While these differences are significant (paired t-tests, respectively $t(4) = 6.95, p = 0.002$ and $t(4) = 3.06, p = 0.038$), the effect is small. See also Fig. C.5 for marginal evidence that meta-classification may be helpful in the cards domain, where there are fewer training tasks.

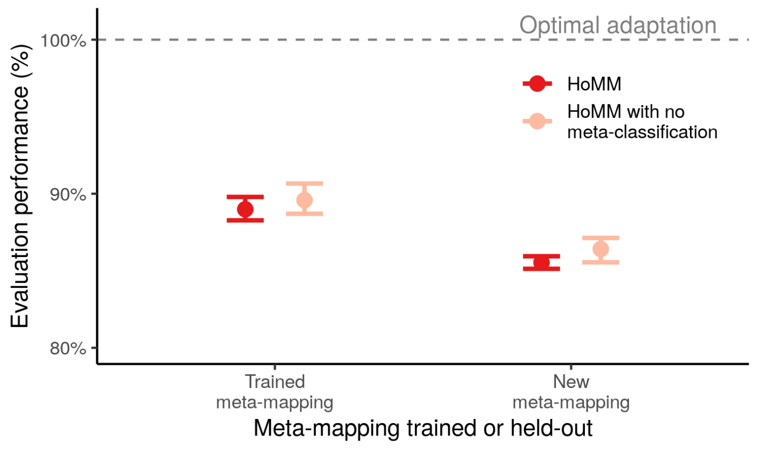


Figure B.3: In the polynomials domain, the HoMM model performs slightly better without meta-classification training. This effect appears for both trained and held-out meta-mappings. However, the effect is small.

Appendix C

Supplemental material for Chapter 3

C.1 Details of human experiment

The experiment was implemented online, and run on Amazon Mechanical Turk. The code was based on the jsPsych library (de Leeuw, 2015). The full code for the experiment and analysis can be found at `https://github.com/lampinen/cards_for_humans`. In this section we provide some details in a more easily readable format.

C.1.1 Detailed experiment outline

Introduction & rules

- Page 1:
    - Hi, welcome to our HIT. We are researchers from the Stanford Department of Psychology, conducting an experiment on game playing.
    - The first part of this experiment should take 5-10 minutes. The base pay is $1, and if you pass the end of the first phase, there will be a second phase that will take about 10 minutes. You will be paid a bonus of $1.50 for making it to this second phase, there will be an extra bonus based on performance in the second phase.
    - If you do not wish to participate, you may return the HIT at any time, but you will not be compensated unless you complete it.
- Page 2:

  - If you make it to the second phase of the experiment, you'll be playing a simple card game. In this first phase of the experiment, you'll learn the rules.
  - We'll test your understanding of the rules at the end of the first phase, and if you pass you'll make it to the second phase, where you'll earn a $1 bonus + an extra performance bonus.
  - Make sure you follow all instructions very carefully, in order to make it to the second phase of the experiment and earn the maximum bonus pay.

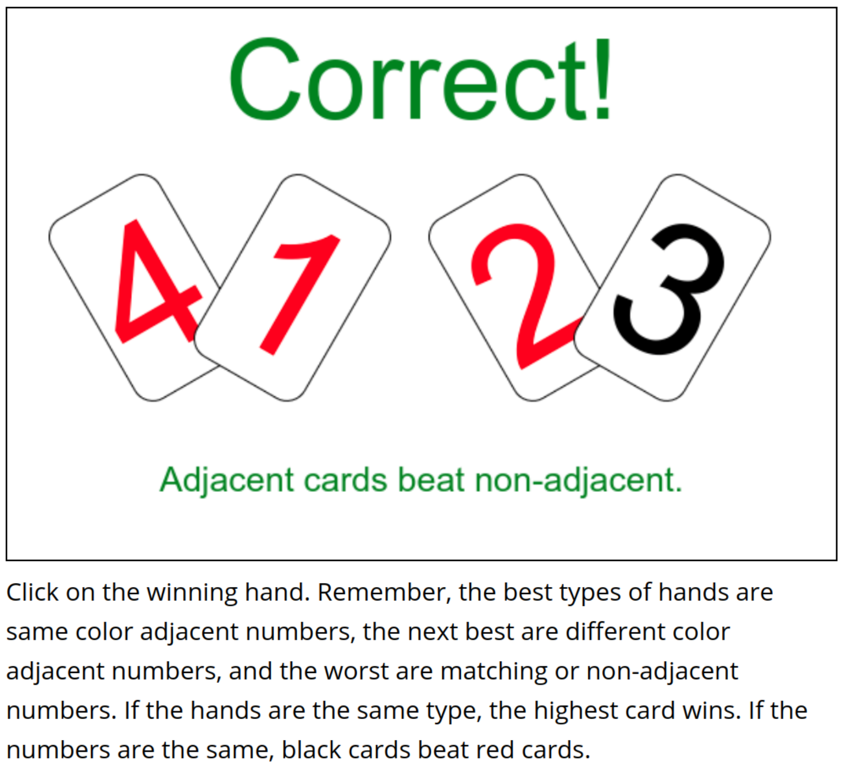
- Page 3:
  - In the card game, you will receive a hand of two cards, each of which has a number (1-4) and a color (red or black). There are several decks in play, so there are multiple copies of each card, and two or more can appear in the same round.
- Page 4:
  - You will be playing against an opponent, trying to win money. You'll get to make a bet of 0, 5, or 10 cents.
  - If your hand beats your opponent's hand, you will win the amount you bet. If your opponent wins, you'll lose the amount you bet. If you bet nothing, you won't win or lose anything. Also, if you tie, you won't win or lose either.
  - In the second phase, we'll pay you a bonus equal to your net earnings (or 0 if your earnings are negative), on top of the $1 bonus for making it to the second phase."
- Page 5:
  - In the card game, **the best types of hands are two adjacent numbers of the same color**, for example black 2 and black 3.
  - **The next best hands are those with two adjacent numbers of different color,** for example black 3 and red 4.
  - **The worst types of hands are those with matching numbers or non-adjacent numbers**, like 4, 4 or 1, 3.
  - **Hands of better types always beat worse hands.**
- Page 6:
  - **If two hands are of the same type, the one with the highest card wins.** If the highest cards of the two hands tie, the tie is broken by the lower cards.
  - **If both cards are tied, black cards beat red cards,** highest first, lowest if the high cards are the same color. If the hands are perfectly tied, you don't win or lose money.

Game understanding check

- Instructions:
  - We'll now test your understanding by giving you a few example pairs of hands. Just click on the hand from each pair that would win. **If you make more than one mistake in this section, the experiment will end. Make sure you fully understand the instructions before proceeding, otherwise you may not make it to the second phase!**
  - If you need to, you can go back to the earlier instructions to refresh your memory before proceeding.
  - Click next to start the test.
- Trials (see fig. C.1, hand position was randomized):
  - black 3 and red 2 vs. red 4 and 1. Explanation: "Adjacent cards beat non-adjacent."
  - red 1 and 2 vs. red 4 and black 3. Explanation: "Same-suit adjacent beats different suit."
  - black 3 and red 3 vs. red 4 and 1. Explanation: "Highest card breaks ties."
  - black 4 and 3 vs. red 4 and 3. Explanation: "Black cards beat red cards if the numbers are tied."
- Evaluation: If the participants got 3 out of 4 trials correct, they passed.
  - **Passed:** Congratulations, you passed the test, and will get to proceed to phase 2! You have earned a \$1.50 bonus, and will be awarded a performance bonus based on your bets in the next phase. Press any key to continue.
  - **Failed:** Sorry, you made more than one mistake, and did not pass the test. The experiment will now end. Press any key to continue.

Block 1: with feedback

- Instructions:
  - Now you get to play a few hands. After you bet, we'll show you your opponent's hand and how much you won (or lost), and at the end of these hands we'll tell you your total earnings. Press any key to continue.
- Trials:
  - 32 trials of playing the game and seeing the result of each hand, with the participants hand distributed evenly across 16 bins of hand win probability. Opponents hands were randomly sampled.



Correct!

Adjacent cards beat non-adjacent.

Click on the winning hand. Remember, the best types of hands are same color adjacent numbers, the next best are different color adjacent numbers, and the worst are matching or non-adjacent numbers. If the hands are the same type, the highest card wins. If the numbers are the same, black cards beat red cards.

Figure C.1: An example hand-comparison trial from the understanding check.

- Block end:
  - Participants were shown their block earnings, as well as their total earnings so far.

Block 2: without feedback

- Instructions:
  - To test how well you understand the game, we'll now give you a series of hands where you won't see your results after you bet. You will just see your earnings at the end of the set of hands. Press any key to continue.
- Trials:
  - 24 trials of playing the game with the result grayed out, with the participants hand distributed evenly across 8 bins of hand win probability. Opponents hands were not sampled, participants were paid their expected earnings for each hand, with the final block total rounded to the closest 10 cents.
- Block end:
  - Participants were shown their block earnings, as well as their total earnings so far.

Block 3: losing variation, no feedback

- Instructions:
  - **Now, we want you to try to lose the game! For the remainder of the experiment, if you bet and lose, you'll gain the amount you bet, and if you bet and win, you'll lose the amount you bet.**
  - As before, you won't win or lose anything if you tie your opponent, or if you don't bet.
  - Press any key to continue.
- Attention check:
  - To make sure you understand, please answer this question. From now on, I will earn money if I:
  - "Bet and my hand wins.", "Bet and my hand loses."
- Instructions 2:
  - Great, now you get to play a few hands! As before, you won't see your results after you bet. You will just see your earnings at the end of the set of hands. Press any key to continue.

- Trials:
  - 24 trials of playing the game with the result grayed out, with the participants hand distributed evenly across 8 bins of hand win probability. Opponents hands were not sampled, participants were paid their expected earnings for each hand, with the final block total rounded to the closest 10 cents.
- Block end:
  - Participants were shown their block earnings, as well as their total earnings so far.

Debrief

We asked participants about their age, education, gender, and race/ethnicity. However, we did not analyze these data.

C.2 Details of model tasks & training

The full code for the model training and analysis can be found at: `https://github.com/lampinen/HoMM_cards`.

C.2.1 Tasks

Our card games were played with two suits, and 4 values per suit. In our setup, each hand in a game has a win probability (proportional to how it ranks against all other possible hands). The agent is dealt a hand, and then has to choose to bet 0, 1, or 2 (the three actions it has available). We considered a variety of games which depend on different features of the hand:

- **Straight flush:** Most valuable is adjacent numbers in same suit, i.e. 4 and 3 in most valuable suit (royal flush) wins against every other hand. This is the game we tested in human participants.
- **High card:** Highest card wins.
- **Pairs** Same as high card, except pairs are more valuable, and same suit pairs are even more valuable.
- **Match:** The hand with cards that differ least in value (suit counts as 0.5 pt difference) wins.
- **Blackjack:** The hand's value increases with the sum of the cards until it crosses 5, at which point the player "goes bust," and the value becomes negative.

We also considered three binary attributes that could be altered to produce variants of these games:

- **Losers:** Try to lose instead of winning! Reverses the ranking of hands. This is the mapping we evaluated in human participants.
- **Suits rule:** Instead of suits being less important than values, they are more important (essentially flipping the role of suit and value in most games).
- **Switch suit:** Switches which of the suits is more valuable.

Any combination of these options can be applied to any of the 5 games, yielding 40 possible games.

C.2.2 Training

Meta-mappings: We trained the network on meta-mappings that toggled each of the binary attributes, but evaluated primarily on switching to losing the Straight Flush game (since that corresponded to the human experiment).

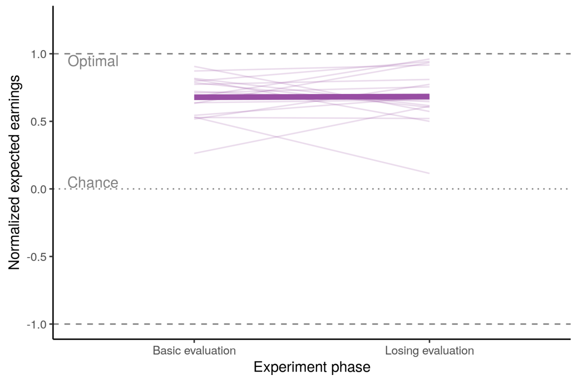
Meta-classifications: For meta-tasks, we gave the network 8 task-embedding classification tasks (one-vs-all classification of each of the 5 game types, and of each of the 3 attributes)

Language: We encoded the tasks in language by sequences of the form `[‘‘game’’, <game_type>, ‘‘losers’’, <losers-value>, ‘‘suits rule’’, <suits-rule-value>, ‘‘switch suit’’, <switch-suit-value>]`.

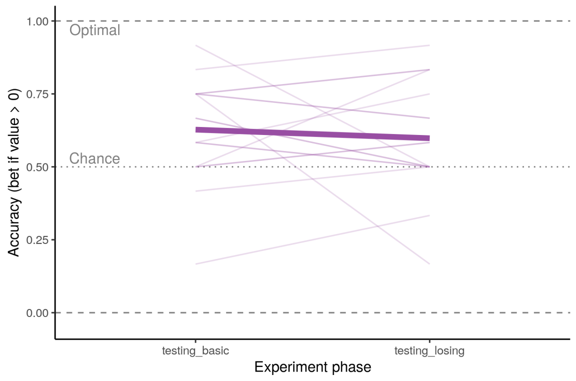
C.3 Supplementary analyses

C.3.1 Human suboptimality

As Jarvstad et al. (2013) note, how "optimal" human performance seems to be depends on how you measure performance. In particular, performance seems better when measured in terms of expected earnings than when measured in terms of how accurately participants decided whether or not to bet (fig. C.2). This is because the participants were more accurate on trials with higher (absolute) expected value, and less accurate on trials where they had less to gain or lose. We chose the more optimistic performance measure as the basis for our comparison to the HoMM model.



(a) Performance measured in terms of earnings.



(b) Performance measured in terms of accuracy, i.e. the percent of hands where an optimal decision was made

Figure C.2: How optimal human performance appears depends on the metric used to evaluate it.

C.3.2 Basic meta-learning in the cards domain

In Fig. C.3, we show that the basic meta-learning is working well in the cards domain. That is, we show that after the example network is presented with a set of example (hand, bet, reward) tuples, the system is generalizing well to other hands of that game. At the end of training, the mean reward on trained games is 99.20% of optimal (bootstrap 95%-CI [98.90, 99.40]), and for held-out games it is 83.82% (bootstrap 95%-CI [80.50, 86.00]).

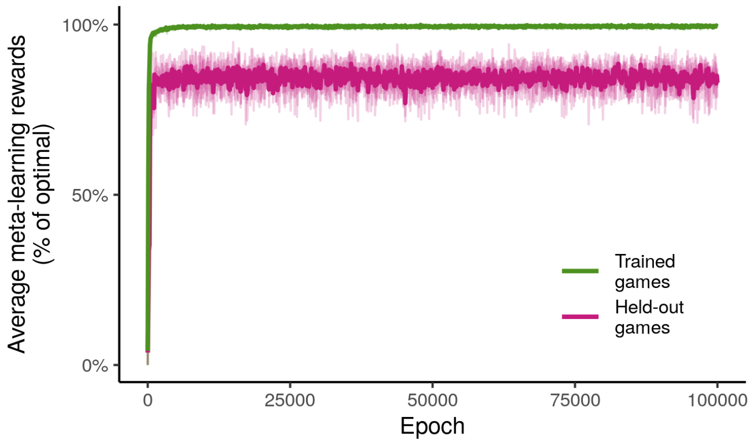


Figure C.3: Basic meta-learning performance in the cards domain over learning. The system is generalizing at the meta-learning level. That is, this graph shows that, after the example network receives a set of (hand, bet, reward) example tuples from a game, it is generating a sufficiently good representation of that game to play held-out hands. This is true both for gamess it was trained with (green), and for games that are held-out and never encountered during training (pink). (Thick dark curves are averages over 5 runs, shown as light curves.)

C.3.3 Comparing to a simpler architecture for language generalization

In Fig. C.4 we show that language generalization is comparable in the HyperNetwork-based architecture we used for HoMM and a simpler architecture which simply concatenates the task representation to the input representation before passing them through a fixed feed-forward task network. Specifically, the HyperNetwork architecture achieves a mean expected reward of 1.79% (bootstrap 95%-CI [-12.31, 15.88]), while the simpler architecture achieves a mean expected reward of -8.59% (bootstrap 95%-CI [-20.42, -0.99]). See Supp. Figs. B.2 and D.2 for the same architecture comparison for HoMM itself.

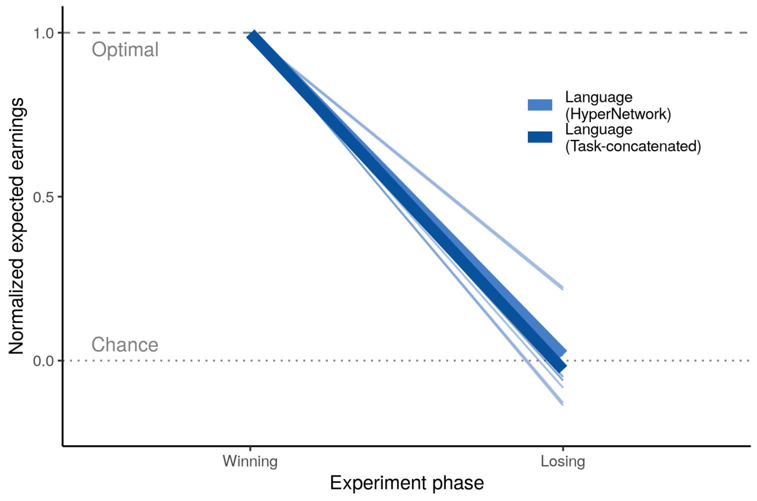


Figure C.4: Language generalization is similar in the cards domain with either the HyperNetwork architecture used by HoMM, or a simpler task-concatenated architecture. Compare to Fig. 3.5 for the human and HoMM results.

C.3.4 HoMM without meta-classification

In Fig. C.5 we show that the HoMM model may be performing slightly better with meta-classification training than without it, although the difference is only marginally significant (paired t-test, $t(4) = 2.23, p = 0.09$). Specifically, the HoMM model is achieving an average expected reward of 85.38% (bootstrap 95%-CI [79.49, 90.32]), while without meta-classification it is achieving an average expected reward of 78.68% (bootstrap 95%-CI [71.01, 85.97]). See Fig. B.3 for a similar comparison in the polynomials domain, where meta-classification appears deleterious (possibly because there are many more training tasks, so it is not needed).

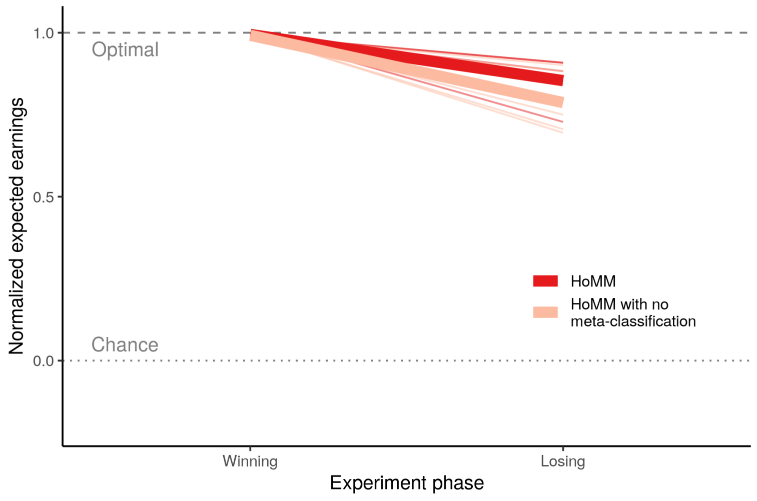


Figure C.5: The HoMM model performs marginally worse without meta-classification training. Thus this training may allow the model to adapt more robustly to new tasks.

C.3.5 Non-homoiconic meta-mapping

In Fig. C.6 we compare HoMM to a non-homoiconic meta-mapping approach, as in Fig. 2.9 in the polynomials domain. In the cards domain, the non-homoiconic approach may perform slightly worse, but the difference is not significant. Specifically, the HoMM model is achieving an average expected reward of 85.38% (bootstrap 95%-CI [79.49, 90.32]), while non-homoiconic meta-mapping is achieving an average expected reward of 79.49% (bootstrap 95%-CI [69.50, 87.34]).

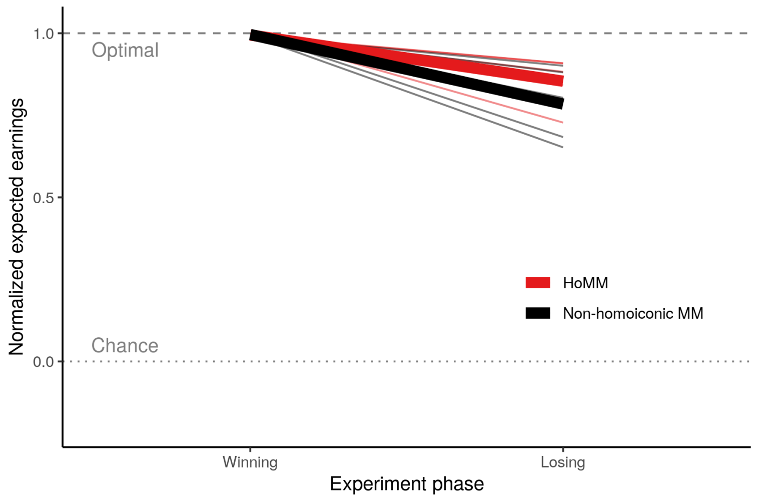


Figure C.6: HoMM performs comparably to a non-homoiconic lesion in the cards domain. This figure compares the meta-mapping performance of HoMM with a nonhomoiconic model that instantiates separate copies of the example network ($\mathcal{E}_{base}, \mathcal{E}_{meta}$) and hyper network ($\mathcal{H}_{base}, \mathcal{H}_{meta}$) for the basic tasks and the meta-mappings.

Appendix D

Supplemental material for Chapter 4

D.1 RL tasks

All implementation and analysis code can be found at `https://github.com/lampinen/HoMM_grids`.

D.1.1 Correlation in generalization across tasks

In Fig. D.1, we show the correlation in performance on the pick-up and push-off generalization tasks within each run (at different time points in learning). Points are only included if train performance is above the threshold used for selection — 3.8 for the HoMM model, 3.5 for the language model. Stricter thresholds for the language model result in weaker (sometimes negative) correlations (not shown).

D.1.2 HyperNetwork-based architecture

In Fig. D.2, we show that the HyperNetwork-based architecture outperforms a task-network-concatenating architecture at meta-mapping on the RL tasks, as in the polynomials domain.

D.2 Categorization tasks

All implementation and analysis code can be found at `https://github.com/lampinen/categorization_HoMM`.

In Fig. D.3 we show all shapes (triangle, square, plus, circle, tee, inverseplus, emptysquare, emptytriangle), colors (blue, pink, purple, yellow, ocean, green, cyan, red), and sizes (16, 24, and

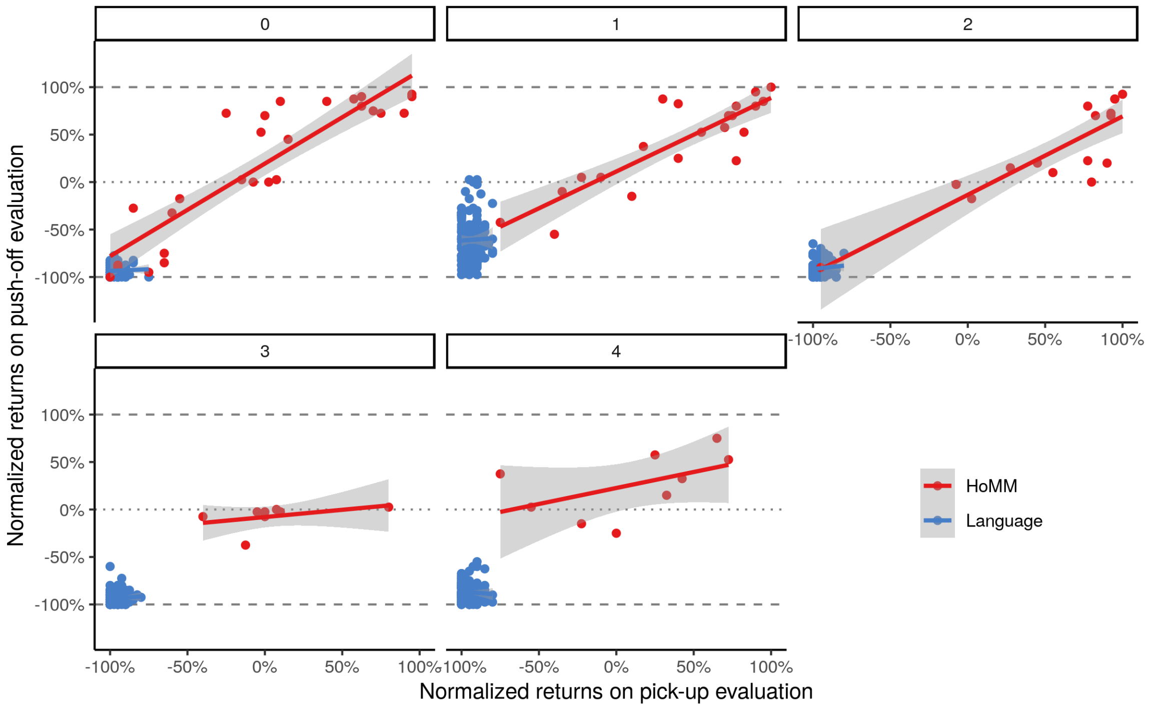


Figure D.1: Correlation of performance on the two RL tasks, broken down by run. The correlation is higher in the HoMM model, both within and across runs.

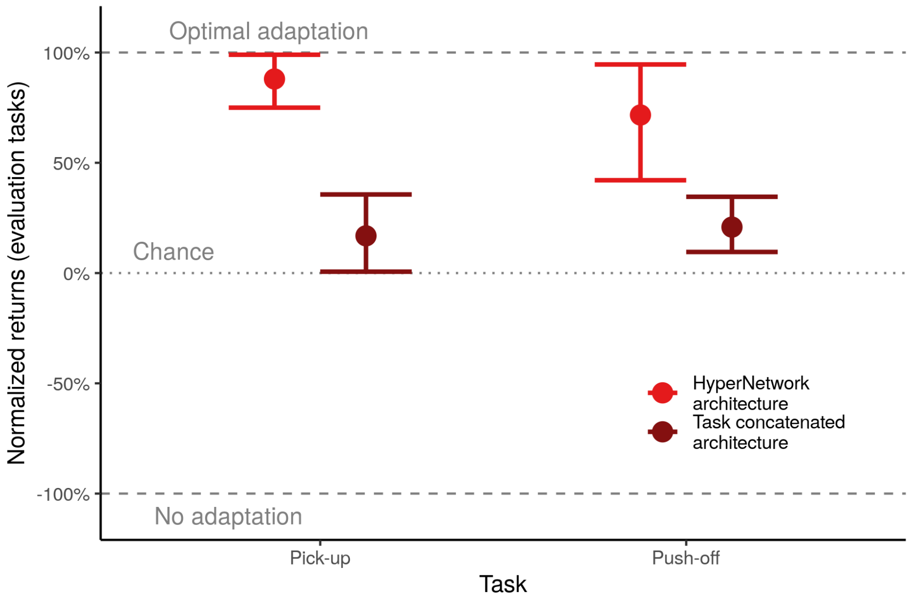


Figure D.2: In the RL domain, the HyperNetwork-based architecture performs better on meta-mappings than an architecture that simply concatenates a task representation to the input before passing it through a fixed MLP. We showed similar (though less dramatic) results for the polynomials domain in Supp. Fig. B.2

32 pixels) that we used in our experiments. All stimuli were rendered at random positions within a 50×50 image (constrained so that the full shape remained within the frame), and at random angles within $\pm 20°$ of their canonical orientation.

D.2.1 Language model architecture

In the categorization experiments, we used a different task network architecture for the meta-mapping based architectures than for the language generalization architectures. Here, we justify that choice by showing that the model architecture we used for the meta-mapping approach results in worse language generalization, in Fig. D.4. In particular, the linear task network resulted in worse generalization performance (mean = 0.85, bootstrap 95%-CI [0.82, 0.88]) than the deep nonlinear task network (mean = 0.92, bootstrap 95%-CI [0.89, 0.94]). This difference was significant under a linear mixed-model ($t(4) = 3.615$, $p = 0.02$), and under a permutation test.

D.2.2 More detailed result visualizations

In Fig. D.5 we show the zero-shot generalization accuracy of the models across runs, at different training set sizes. At moderate sample sizes, the HoMM model results in a sharper peak at perfect accuracy — i.e. more qualitatively "getting it" or "not getting it."

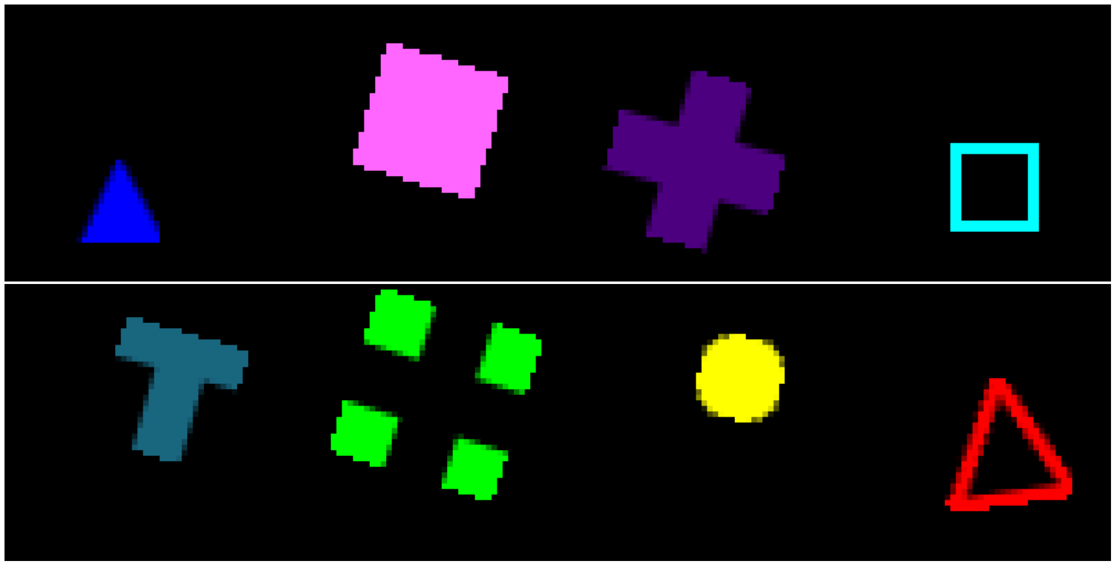


Figure D.3: Sample stimuli for categorization tasks, showing all shapes, colors, and sizes.

In Fig. D.6 we show learning curves for meta-mapping performance across all runs. Performance is highly variable at small training-set sizes, especially on held-out meta-mappings, but becomes increasingly systematic as training set size increases.

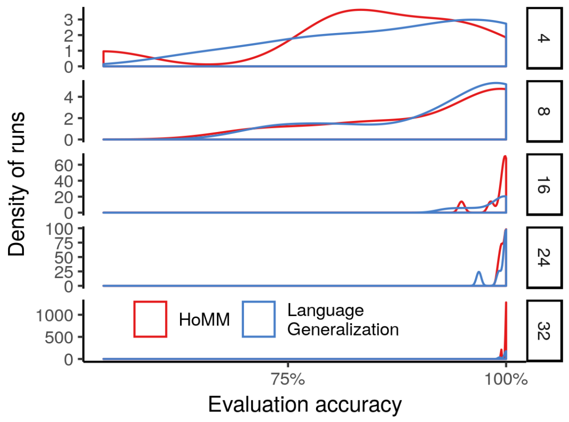


Figure D.5: In the visual concepts domain, meta-mapping results in more qualitative "getting it" or "not getting it" behavior, in the middle ranges of dataset size. Here we plot the density of the zero-shot evaluation accuracy across runs for the HoMM model and language generalization. The HoMM model exhibits sharper peaking at one at moderate sample sizes, whereas the language generalization is more smeared out — i.e. the HoMM model is either systematically getting everything correct or is making a large number of mistakes, whereas the language generalization is more stochastic. This qualitative, systematic difference in performance that HoMM exhibits is more like what would be expected from human cognition.

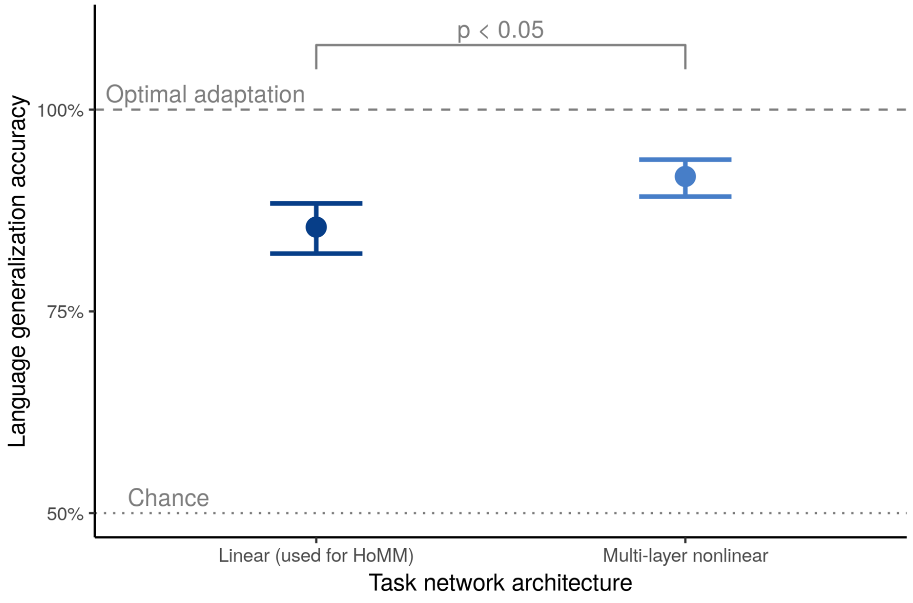
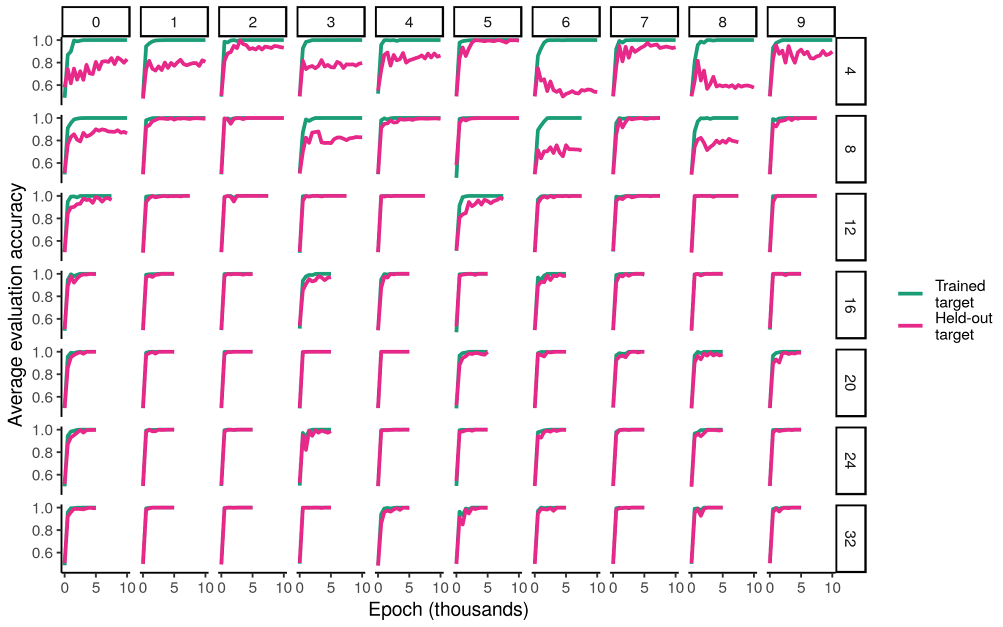
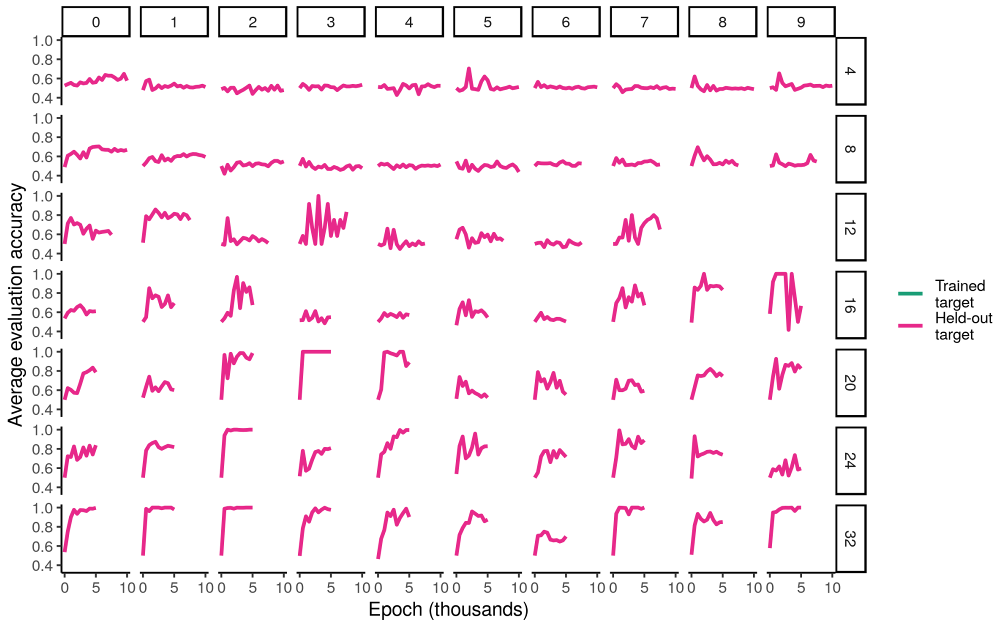


Figure D.4: Comparing language generalization with a linear task network to a deep, nonlinear architecture. Although the linear task network worked best for the meta-mapping approaches (not shown), the nonlinear task network generalized better to new language instructions.



(a) Trained meta-mappings.



(b) Held-out meta-mappings.

Figure D.6: Meta-mapping learning curves in the visual concepts domain broken down by number of training meta-mappings (rows), and by run (columns). The green lines are performance when the transformed task was encountered during training, the pink lines are performance on transformed tasks that were never encountered during training. Panel (a) shows the results for trained meta-mappings, and panel (b) shows the results for held-out meta-mappings. With more training meta-mappings, HoMM both generalizes better when applying the trained meta-mappings to held-out examples (a), and when applying held-out meta-mappings (b). However, even with smaller sample sizes, HoMM is achieving perfect generalization on the trained meta-mappings on many runs.

Bibliography

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *International Conference on Machine Learning*.

Achille, A., Eccles, T., Matthey, L., Burgess, C. P., Watters, N., Lerchner, A., and Higgins, I. (2018). Life-Long Disentangled Representation Learning with Cross-Domain Latent Homologies. In *Advances in Neural Information Processing Systems*.

Achille, A., Lam, M., Tewari, R., Ravichandran, A., Maji, S., Fowlkes, C., Soatto, S., and Perona, P. (2019). Task2Vec: Task Embedding for Meta-Learning. *arXiv preprint*.

Achille, A., Rovere, M., and Soatto, S. (2017). Critical Learning Periods in Deep Neural Networks. *arXiv preprint*.

Anderson, M. L. (2003). Embodied Cognition : A field guide. *Artificial Intelligence*, 149:91–130.

Anderson, P. W. (1972). More Is Different. *Science*, 177(4047):393–396.

Andreas, J., Klein, D., and Levine, S. (2016a). Modular Multitask Reinforcement Learning with Policy Sketches. *arXiv preprint*.

Andreas, J., Klein, D., and Levine, S. (2017a). Learning with Latent Language. *arXiv preprint*, (Figure 1).

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016b). Learning to Compose Neural Networks for Question Answering. *arXiv preprint*.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2017b). Deep Compositional Question Answering with Neural Module Networks.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., Mcgrew, B., Tobin, J., Abbeel, P., and Zaremba, W. (2017). Hindsight Experience Replay. *Neural Information Processing Systems*, (Nips).

Arora, S., Cohen, N., and Hazan, E. (2018a). On the Optimization of Deep Networks: Implicit Acceleration by Overparameterization.

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. (2018b). Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint*, pages 1–39.

Aslin, R. N. and Newport, E. L. (2012). Statistical Learning: From Acquiring Specific Items to Forming General Rules. *Current Directions in Psychological Science*, 21(3):170–176.

Atkinson, C., McCane, B., Szymanski, L., and Robins, A. (2018). Pseudo-Recursal: Solving the Catastrophic Forgetting Problem in Deep Neural Networks. *arXiv preprint*.

Ba, J., Hinton, G., Mnih, V., Leibo, J. Z., and Ionescu, C. (2016). Using Fast Weights to Attend to the Recent Past. In *Advances in Neural Information Processing Systems*, pages 1–10.

Baars, B. J. (2005). Global workspace theory of consciousness: Toward a cognitive neuroscience of human experience. *Progress in Brain Research*, 150:45–53.

Baranes, A. and Oudeyer, P. Y. (2013). Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1):49–73.

Barlow, H. B. (1975). Visual experience and cortical development. *Nature*, 258.

Barnett, S. M. and Ceci, S. J. (2002). When and where do we apply what we learn? A taxonomy for far transfer. *Psychological bulletin*, 128(4):612–637.

Barsalou, L. W. (2007). Grounded Cognition. *Annual Review of Psychology*, 59(1):617–645.

Bartlett, P. L. and Mendelson, S. (2002). Rademacher and Gaussian Complexities: Risk Bounds and Structural Results. *Journal of Machine Learning Research*, 3:463–482.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks. pages 1–38.

Bellemare, M. G. and Dabney, W. (2017). A Distributional Perspective on Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning.*

Bengio, Y. (2012). Evolving culture vs local minima. *arXiv preprint arXiv:1203.2990*, 2006:1–28.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Blodgett, H. C. (1929). The effect of the introduction of reward upon the maze performance of rats. *University of California Publications in Psychology*, 4:113–134.

Botvinick, M., Ritter, S., Wang, J. X., Kurth-nelson, Z., Blundell, C., and Hassabis, D. (2019). Reinforcement Learning , Fast and Slow. *Trends in Cognitive Sciences*, pages 1–15.

Botvinick, M. M., Niv, Y., and Barto, A. C. (2009). Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective. *Cognition*, 113(3):262–280.

Bourne, L. E. (1970). Knowing and using concepts. *Psychological Review*, 77(6):546–556.

Bransford, J. D. and Schwartz, D. L. (1999). Rethinking Transfer : A Simple Proposal With Multiple Implications. *Review of Research in Education*, 24(1):61–100.

Brock, A., Lim, T., Ritchie, J. M., and Weston, N. (2018). SMASH: One-Shot Model Architecture Search through HyperNetworks. In *International Conference on Learning Representations*.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. *arXiv preprint*.

Burger, W. and Shaughnessy, J. (1986). Characterizing the van Hiele Levels of Development in Geometry. *Journal for research in mathematics . . .*, 17(1):31–48.

Burgess, J., Lloyd, J. R., and Ghahramani, Z. (2016). One-Shot Learning in Discriminative Neural Networks. *Workshop on Bayesian Deep Learning, NIPS 2016*, pages 3–5.

Cao, S., Wang, X., and Kitani, K. M. (2019). Learnable Embedding Space for Efficient Neural Architecture Compression. In *International Conference on Learning Representations*, pages 1–17.

Carpenter, M., Moll, H., Tomasello, M., Call, J., and Behne, T. (2005). Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and Brain Sciences*, 28(05):675–735.

Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(q).

Chalmers, D. J. (2006). Strong and Weak Emergence. In Clayton, P. and Davies, P., editors, *The re-emergence of emergence*.

Chang, M. B., Gupta, A., Levine, S., and Griffiths, T. L. (2019). Automatically Composing Representation Transformations as a Means for Generalization. In *International Conference on Learning Representations*, pages 1–23.

Changpinyo, S., Chao, W. L., Gong, B., and Sha, F. (2016). Synthesized classifiers for zero-shot learning. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:5327–5336.

Chen, Z. and Klahr, D. (1999). All Other Things Being Equal : Acquisition and Transfer of the Control of Variables Strategy. *Child Development*, 70(5):1098–1120.

Chi, M. T., De Leeuw, N., Chiu, M. H., and Lavancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science*, 18(3):439–477.

Chi, M. T. H., Lewis, M. W., Reimann, P., and Glaser, R. (1989). How Students Study and Use Examples in Learning to Solve Problems. *Cognitive Science*, 13(2):145–182.

Chollet, F. (2019). On the Measure of Intelligence. *arXiv preprint*, pages 1–64.

Cisek, P. (1999). Beyond the computer metaphor: Behavior as interaction. *Journal of Consciousness Studies*, 6(11-12).

Cisek, P. (2019). Resynthesizing behavior through phylogenetic refinement. *Attention, Perception, & Psychophysics*.

Clark, A. and Karmiloff-Smith, A. (1993). The Cognizer's Innards: A Psychological and Philosophical Perspective on the Development of Thought. *Mind & Language*, 8(4):487–519.

Cleeremans, A. (2014). Connecting conscious and unconscious processing. *Cognitive Science*, 38(6):1286–1315.

Cleeremans, A. and Jiménez, L. (2002). Implicit learning and consciousness: A graded, dynamic perspective. In *Implicit Learning and Consciousness: An Empirical, Philosophical and Computational Consensus in the Making (Frontiers of Cognitive Science)*.

Clune, J. (2019). AI-GAs: AI-generating algorithms, an alternate paradigm for producing general artificial intelligence. *arXiv preprint*.

Cohen, J. D., Dunbar, K., and McClelland, J. L. (1990). On the control of automatic processes: A parallel distributed processing account of the stroop effect. *Psychological Review*, 97(3):332–361.

Dabney, W., Kurth-nelson, Z., Uchida, N., Starkweather, C. K., Hassabis, D., Munos, R., and Botvinick, M. (2020). A distributional code for value in dopamine-based reinforcement learning. *Nature*, (January):1–32.

Dauphin, Y., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *arXiv*, pages 1–14.

Davis, E. (2019). The Use of Deep Learning for Symbolic Integration: A Review of (Lample and Charton , 2019). *arXiv preprint*, pages 1–7.

Daw, N. D. and Dayan, P. (2014). The algorithmic anatomy of model-based evaluation. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1655).

Day, S. B. and Goldstone, R. L. (2011). Analogical Transfer From a Simulated Physical System. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 37(3):551–567.

De Bock, D., Deprez, J., Van Dooren, W., Roelens, M., and Verschaffel, L. (2011). Abstract or Concrete Examples in Learning Mathematics? A Replication and Elaboration of Kaminski, Sloutsky, and Heckler's Study. *Journal for Research in Mathematics Education*, 42(2):109.

de Leeuw, J. R. (2015). jsPsych: A JavaScript library for creating behavioral experiments in a web browser. *Behavior Research Methods*, 47(1):1–12.

Dehaene, S., Lau, H., and Kouider, S. (2017). What is consciousness, and could machines have it? *Science*, 358(6362):484–489.

Deng, J. D. J., Dong, W. D. W., Socher, R., Li, L.-J. L. L.-J., Li, K. L. K., and Fei-Fei, L. F.-F. L. (2009). ImageNet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2–9.

Detterman, D. K. (1993). The Case for the Prosecution: Transfer as an Epiphenomenon. In *Transfer on Trial: Intelligence, Cognition, and Instruction*, pages 1–24.

Devlin, J., Bunel, R., Singh, R., Hausknecht, M., and Kohli, P. (2017). Neural Program Meta-Induction. In *Neural Information Processing Systems*.

Dicarlo, J. J. and Cox, D. D. (2007). Untangling invariant object recognition. *Trends in Cognitive Sciences*, 11(8).

Doebel, S. and Zelazo, P. D. (2015). A meta-analysis of the Dimensional Change Card Sort: Implications for developmental theories and the measurement of executive function in children. *Developmental Review*, 38:241–268.

Dong, D., Wu, H., He, W., Yu, D., and Wang, H. (2015). Multi-Task Learning for Multiple Language Translation. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, 1:1723–1732.

Dove, G. (2016). Three symbol ungrounding problems : Abstract concepts and the future of embodied cognition. *Psychonomic Bulletin & Review*, 23(4):1109–1121.

Drachman, D. A. (2005). Do we have brain to spare ? *Neurology*, 64.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. (2016). RL$ˆ2$: Fast Reinforcement Learning via Slow Reinforcement Learning. *arXiv preprint*, pages 1–14.

Dubinsky, E. (1991). Reflective abstraction in advanced mathematical thinking. *Advanced mathematical thinking*, pages 95–121.

Dujmovic, M., Malhotra, G., and Bowers, J. (2020). What do adversarial images tell us about human vision? *arXiv preprint*.

Elman, J. L. (1993). Learning and development in neural networks: the importance of starting small. *Cognition*, 48(2):71–99.

Elsayed, G. F., Goodfellow, I., Sohl-Dickstein, J., and Brain, G. (2018a). Adversarial Reprogramming of Neural Networks. *arXiv preprint*.

Elsayed, G. F., Shankar, S., Cheung, B., Papernot, N., Kurakin, A., Goodfellow, I., and Sohl-Dickstein, J. (2018b). Adversarial Examples that Fool both Computer Vision and Time-Limited Humans.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., and Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*.

Ericsson, K. A. (2017). Expertise and individual differences: the search for the structure and acquisition of experts' superior performance. *Wiley Interdisciplinary Reviews: Cognitive Science*, 8(1-2):1–6.

Ericsson, K. A., Krampe, R. T., and Tesch-Römer, C. (1993). The role of deliberate practice in the acquisition of expert performance. *Psychological Review*, 100(3):363–406.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. (2019). Diversity is all you need: learning skills without a reward function. In *International Conference on Learning Representations*, pages 1–22.

Falkenhainer, B., Forbus, K. D., and Gentner, D. (1989). The Structure-Mapping Engine : Algorithm and Examples. *Artificial Intelligence*, 41(1):1–63.

Fausey, C. M., Jayaraman, S., and Smith, L. B. (2016). From faces to hands: Changing visual input in the first two years. *Cognition*, 152:101–107.

Feltovich, P. J., Prietula, M. J., and Ericsson, K. A. (2012). Studies of Expertise from Psychological Perspectives. In *The Cambridge Handbook of Expertise and Expert Performance*, pages 41–68.

Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., Pritzel, A., and Wierstra, D. (2017). PathNet: Evolution Channels Gradient Descent in Super Neural Networks. *arXiv*.

Finn, C., Abbeel, P., and Levine, S. (2017a). Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th Annual Conference on Machine Learning*.

Finn, C., Levine, S., and Abbeel, P. (2016). Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization. 48.

Finn, C., Xu, K., and Levine, S. (2018). Probabilistic Model-Agnostic Meta-Learning. *arXiv preprint*.

Finn, C., Yu, T., Zhang, T., Abbeel, P., and Levine, S. (2017b). One-Shot Visual Imitation Learning via Meta-Learning. In *1st Conference on Robot Learning*, pages 1–12.

Florensa, C., Held, D., Geng, X., and Abbeel, P. (2018). Automatic goal generation for reinforcement learning agents. *arXiv preprint*.

Fodor, J. A. (1975). *The language of thought*, volume 5. Harvard university press.

Fodor, J. A. (1983). *The modularity of mind*. MIT press.

Fodor, J. A. (2001a). Language, thought and compositionality. *Mind and Language*, 16(1):1–15.

Fodor, J. A. (2001b). *The mind doesn't work that way: The scope and limits of computational psychology*. MIT press.

Fodor, J. A. (2008). *LOT 2: The language of thought revisited*. Oxford University Press on Demand.

Fodor, J. A. and Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71.

Forbus, K. D., Ferguson, R. W., and Lovett, A. (2017). Extending SME to Handle Large-Scale Cognitive Modeling. 41:1152–1201.

Frank, M. C., Everett, D. L., Fedorenko, E., and Gibson, E. (2008). Number as a cognitive technology: Evidence from Pirahã language and cognition. *Cognition*, 108(3):819–824.

Garcia, V. and Bruna, J. (2018). Few-Shot Learning with Graph Neural Networks. In *International Conference on Learning Representations (ICLR)*.

Garnelo, M., Rosenbaum, D., Maddison, C. J., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D. J., and Eslami, S. M. A. (2018). Conditional Neural Processes. *arXiv preprint*.

Gentner, D. (2003). Why We're So Smart. In *Language in mind: Advances in the study of language and thought.*, pages 195–235.

Gentner, D. and Asmuth, J. (2017). Metaphoric extension , relational categories , and abstraction. *Language, Cognition and Neuroscience*, 0(0):1–10.

Gick, M. L. and Holyoak, K. J. (1980). Analogical Problem Solving. *Cognitive P*, 12:306–355.

Gleitman, L. R. and Gleitman, H. (1970). *Phrase and paraphrase: Some innovative uses of language.*

Goldberg, A. E. (2015). Compositionality. In *Routledge Handbook of Semantics*, pages 419–433.

Goldin, A. P., Pezzatti, L., Battro, A. M., and Sigman, M. (2011). From ancient Greece to modern education: Universality and lack of generalization of the socratic dialogue. *Mind, Brain, and Education*, 5(4):180–185.

Goldin-Meadow, S. (1999). The role of gesture in communication and thinking. *Trends in Cognitive Sciences*, 3(11):419–429.

Goldin-Meadow, S., Alibali, M. W., and Church, R. B. (1993). Transitions in concept acquisition: Using the hand to reach the mind. *Psychological Review*, 100(2):279–297.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. *Iclr 2015*, pages 1–11.

Goodman, N. A., Tenenbaum, J. B., Feldman, J., and Griffiths, T. L. (2008). A rational analysis of rule-based concept learning. *Cognitive Science*, 32(1):108–154.

Gopnik, A. and Sobel, D. M. (2014). Detecting Blickets: How Young Children Use Information about Novel Causal Powers in Categorization and Induction Detecting Blickets: HowYoung Children Use Information about Novel Causal Powers in Categorization and Induction. *Source: Child Development Child Development*, 71(5):1205–1222.

Gordon, P. (2004). Numerical Cognition Without Words: Evidence from Amazonia. *Science*, 306(October):496–499.

Graves, A., Bellemare, M. G., Menick, J., Munos, R., and Kavukcuoglu, K. (2017). Automated curriculum learning for neural networks. *34th International Conference on Machine Learning, ICML 2017*, 3:2120–2129.

Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Gómez Colmenarejo, S., Grefenstette, E., Ramalho, T., Agapiou, J., Badia, A. P., Moritz Hermann, K., Zwols, Y., Ostrovski, G., Cain, A., King, H., Summerfield, C., Blunsom, P., Kavukcuoglu, K., and Hassabis, D. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature Publishing Group*, 538(7626):471–476.

Gregor, K., Danihelka, I., Graves, A., and Wierstra, D. (2015). DRAW: A Recurrent Neural Network For Image Generation. *arXiv preprint*, pages 1–16.

Gregor, K., Rezende, D. J., Besse, F., Wu, Y., Merzic, H., and van den Oord, A. (2019). Shaping Belief States with Generative Environment Models for RL. *arXiv preprint.*

Gregor, K., Rezende, D. J., and Wierstra, D. (2016). Variational Intrinsic Control. *arXiv preprint*, pages 1–15.

Grice, H. (1975). Logic and Conversation. In *Syntax And Semantics*, number January, pages 41–58.

Guez, A., Mirza, M., Gregor, K., Kabra, R., Racanière, S., Weber, T., Raposo, D., Santoro, A., Orseau, L., Eccles, T., Wayne, G., Silver, D., and Lillicrap, T. (2019). An investigation of model-free planning. *arXiv preprint*.

Gülçehre, Ç. and Bengio, Y. (2013). Knowledge Matters : Importance of Prior Information for Optimization. *arXiv preprint arXiv:1301.4083*, pages 1–12.

Ha, D., Dai, A., and Le, Q. V. (2016). HyperNetworks. *arXiv*.

Hansen, S. S., Lampinen, A., Suri, G., and McClelland, J. L. (2017). Building on prior knowledge without building it in. *Behavioral and Brain Sciences*, 40.

Hazzan, O. (1999). Reducing Abstraction Level When Learning Abstract Algebra Concepts. *Educational Studies in Mathematics*, 40:71–90.

He, K., Girshick, R., and Dollar, P. (2018). Rethinking ImageNet Pre-training. *arXiv preprint*.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. (2018). Deep reinforcement learning that matters. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 3207–3214.

Hermann, K. M., Hill, F., Green, S., Wang, F., Faulkner, R., Soyer, H., Szepesvari, D., Czarnecki, W. M., Jaderberg, M., Teplyashin, D., Wainwright, M., Apps, C., and Hassabis, D. (2017). Grounded Language Learning in a Simulated 3D World. *arXiv preprint*, pages 1–22.

Hersh, R. (1997). *What is Mathematics, Really?* Oxford University Press.

Higgins, I., Amos, D., Pfau, D., Racaniere, S., Matthey, L., Rezende, D., and Lerchner, A. (2018). Towards a Definition of Disentangled Representations. *arXiv preprint*, pages 1–29.

Hill, F., Lampinen, A., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., and Santoro, A. (2020). Environmental drivers of generalization in a situated agent. *International Conference on Learning Representations*.

Hill, F., Santoro, A., Barrett, D., Morcos, A., and Lillicrap, T. (2019). Learning to make analogies by contrasting abstract relational structure. In *ICLR*.

Hinton, G. (1986). Learning distributed representations of concepts.

Hinton, G. E. and Plaut, D. C. (1982). Using Fast Weights to Deblur Old Memories. *Proceedings of the 9th Annual Conference of the Cognitive Science Society*, (1987).

Hsu, K., Levine, S., and Finn, C. (2019). Unsupervised Learning Via Meta-Learning. In *International Conference on Learning Representations*.

Hu, W., Lin, Z., Liu, B., Tao, C., Tao, Z., Zhao, D., and Yan, R. (2019). Overcoming catastrophic forgetting for continual learning via model adaptation. In *International Conference on Learning Representations*, pages 1–13.

Huang, Q., Smolensky, P., He, X., Deng, L., and Wu, D. (2017). Tensor Product Generation Networks for Deep NLP Modeling. In *Proceedings of the NAACL-HLT*, pages 1263–1273.

Huh, M., Agrawal, P., and Efros, A. A. (2016). What makes ImageNet good for transfer learning? *arXiv preprint*.

Hung, C. C., Lillicrap, T., Abramson, J., Wu, Y., Mirza, M., Carnevale, F., Ahuja, A., and Wayne, G. (2019). Optimizing agent behavior over long time scales by transporting value. *Nature Communications*, 10(1):1–12.

Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castañeda, A. G., Beattie, C., Rabinowitz, N. C., Morcos, A. S., Ruderman, A., Sonnerat, N., Green, T., Deason, L., Leibo, J. Z., Silver, D., Hassabis, D., Kavukcuoglu, K., and Graepel, T. (2019). Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 364(May):859–865.

Jamrozik, A., McQuire, M., Cardillo, E. R., and Chatterjee, A. (2016). Metaphor: Bridging embodiment to abstraction. *Psychonomic Bulletin and Review*, 23(4):1080–1089.

Jarvstad, A., Hahn, U., Rushton, S. K., and Warren, P. A. (2013). Perceptuo-motor, cognitive, and description-based decision-making seem equally good. *Proceedings of the National Academy of Sciences of the United States of America*, 110(40):16271–16276.

Jiang, Y., Gu, S., Murphy, K., and Finn, C. (2019). Language as an Abstraction for Hierarchical Deep Reinforcement Learning. *arXiv preprint*.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. (2016). Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *arXiv*, pages 1–16.

Kaminski, J. a., Sloutsky, V. M., and Heckler, A. F. (2008). The Advantage of Abstract Examples in Learning Math. *Science*, 320(5875):454–455.

Karmiloff-Smith, A. (1986). From meta-processes to conscious access: Evidence from children's metalinguistic and repair data. *Cognition*, 23(2):95–147.

Karmiloff-Smith, A. (1992). *Beyond modularity: A developmental perspective.* The MIT Press.

Keil, F. C. (2006). Explanation and understanding. *Annual Review of Psychology*, 57:227–254.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. (2016). Overcoming catastrophic forgetting in neural networks. *arXiv preprint.*

Kornblith, S., Shlens, J., and Le, Q. V. (2019). Do Better ImageNet Models Transfer Better? *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2661–2671.

Kotovsky, K., Hayes, J. R., and Simon, H. A. (1985). Why are some problems hard? Evidence from Tower of Hanoi. *Cognitive Psychology*, 17(2):248–294.

Kriegeskorte, N. (2015). Deep neural networks: a new framework for modelling biological vision and brain information processing. *Annual Review of Vision Science*, pages 417–446.

Krizhevsky, A. and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems.*

Kruschke, J. K. (1992). ALCOVE: an exemplar-based connectionist model of category learning.

Kubricht, J. R., Lu, H., and Holyoak, K. J. (2017). Individual differences in spontaneous analogical transfer. *Memory and Cognition*, 45(4):576–588.

Kumaran, D., Hassabis, D., and McClelland, J. L. (2016). What Learning Systems do Intelligent Agents Need? Complementary Learning Systems Theory Updated. *Trends in Cognitive Sciences*, 20(7):512–534.

Kumaran, D. and McClelland, J. L. (2012). Generalization through the recurrent interaction of episodic memories: A model of the hippocampal system. *Psychological Review*, 119(3):573–616.

Lake, B. M. and Baroni, M. (2018). Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 1–12.

Lake, B. M., Linzen, T., and Baroni, M. (2019). Human few-shot learning of compositional instructions. *arXiv preprint.*

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. (2017). Building Machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–55.

Lakoff, G. and Johnson, M. (1980). *Metaphors we live by.* University of Chicago press.

Lakoff, G. and Johnson, M. (1999). *Philosophy in the flesh: the embodied mind and its challenge to western thought.*

Lampinen, A., Hsu, S., and Mcclelland, J. L. (2017). Analogies Emerge from Learning Dyamics in Neural Networks. *Proceedings of the 39th Annual Conference of the Cognitive Science Society*, pages 2512–2517.

Lampinen, A. K. and Ganguli, S. (2019). An analytic theory of generalization dynamics and transfer learning in deep linear networks. In *International Conference on Learning Representations*, pages 1–20.

Lampinen, A. K. and McClelland, J. L. (2017). One-shot and few-shot learning of word embeddings. *arXiv preprint.*

Lampinen, A. K. and McClelland, J. L. (2018). Different Presentations of a Mathematical Concept Can Support Learning in Complementary Ways. *Journal of Educational Psychology.*

Lampinen, A. K. and McClelland, J. L. (2019). Zero-shot task adaptation by homoiconic meta-mapping. *NeurIPS Learning Transferable Skills Workshop*, pages 1–27.

Lample, G. and Charton, F. (2019). Deep learning for symbolic mathematics. *arXiv preprint*, pages 1–24.

Landrum, R. E. (2005). Production of negative transfer in a problem-solving task. *Psychological Reports*, 97:861–866.

Landy, D. and Goldtone, R. L. (2007). Formal notations are diagrams: evidence from a production task. *Memory & cognition*, 35(8):2033–2040.

Laroche, R. and Barlier, M. (2017). Transfer Reinforcement Learning with Shared Dynamics. In *Proceedings of the Thirty First AAAI Conference on Artificial Intelligence*, pages 2147–2153.

Larochelle, H., Erhan, D., and Bengio, Y. (2008). Zero-data learning of new tasks. *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, (Miller 2002):646–651.

Le Cun, Y. (2016). Predictive Learning (NIPS Keynote). In *NIPS.*

Li, H., Dong, W., Mei, X., Ma, C., Huang, F., and Hu, B.-G. (2019). LGM-Net: Learning to Generate Matching Networks for Few-Shot Learning. *Proceedings of the 36th International Conference on Machine Learning.*

Liu, S., Davison, A. J., and Johns, E. (2019). Self-Supervised Generalisation with Meta Auxiliary Learning. *arXiv preprint.*

Locatello, F., Bauer, S., Lucic, M., Rätsch, G., Gelly, S., Schölkopf, B., and Bachem, O. (2019). Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. *Proceedings of the 36th International Conference on Machine Learning.*

Lombrozo, T. (2006). The structure and function of explanations. *Trends in Cognitive Sciences*, 10(10):464–470.

Luchins, A. S. (1942). Mechanization in problem solving. *Psychological Monographs.*

Luong, M.-T., Le, Q. V., Sutskever, I., Vinyals, O., and Kaiser, L. (2016). Multi-task Sequence to Sequence Learning. *ICLR*, pages 1–9.

Lupyan, G. and Zettersten, M. (2020). Does vocabulary help structure the mind? *PsyArXiv preprint.*

Mac Lane, S. (1986). *Mathematics: Form and Function.* Springer-Verlag.

MacLeod, C. M. (1991). Half a century of reseach on the stroop effect: An integrative review. *Psychological Bulletin*, 109(2):163–203.

Madarasz, T. J. and Behrens, T. E. (2019). Better transfer learning with inferred successor maps. *Advances in Neural Information Processing Systems*, (NeurIPS).

Mao, J., Gan, C., Kohli, P., Tenenbaum, J. B., and Wu, J. (2019). The neuro-symbolic concept learner: interpreting scenes, words, and sentences from natural supervision. In *International Conference on Learning Representations*, pages 1–28.

Marcus, G. (2018). Deep Learning: A Critical Appraisal. *arXiv preprint*, pages 1–27.

Markant, D. B. and Gureckis, T. M. (2014a). Is it better to select or to receive? Learning via active and passive hypothesis testing. *Journal of Experimental Psychology: General*, 143(1):94–122.

Markant, D. B. and Gureckis, T. M. (2014b). Is it better to select or to receive? Learning via active and passive hypothesis testing. *Journal of experimental psychology. General*, 143(1):94–122.

Markant, D. B., Settles, B., and Gureckis, M. (2015). Self-Directed Learning Favors Local , Rather Than Global , Uncertainty. *Cognitive Science*, 40(1):1–21.

Marmanis, D., Datcu, M., Esch, T., and Stilla, U. (2016). Deep Learning Earth Observation Classification Using ImageNet Pretrained Networks. *IEEE Geoscience and Remote Sensing Letters*, 13(1):105–109.

McClelland, J. and Patterson, K. (2002). Rules or connections in past-tense inflections: what does the evidence rule out? *Trends in cognitive sciences*, 6(11):465–472.

McClelland, J. L. (2010). Emergence in Cognitive Science. *Topics in Cognitive Science*, 2:751–770.

McClelland, J. L. (2013). Incorporating rapid neocortical learning of new schema-consistent information into complementary learning systems theory. *Journal of experimental psychology. General*, 142:1190–210.

McClelland, J. L., Botvinick, M. M., Noelle, D. C., Plaut, D. C., Rogers, T. T., Seidenberg, M. S., and Smith, L. B. (2010). Letting structure emerge: connectionist and dynamical systems approaches to cognition. *Trends in Cognitive Sciences*, 14(8):348–356.

McClelland, J. L., Hill, F., Rudolph, M., Baldridge, J., and Schütze, H. (2019). Extending machine language models toward human-level language understanding. *arXiv preprint*, pages 1–8.

McClelland, J. L., McNaughton, B. L., and Lampinen, A. K. (2020). Integration of new information in memory: new insights from a complementary learning systems perspective. *Proceedings of the Royal Society B*, pages 1–34.

McClelland, J. L., McNaughton, B. L., and O'Reilly, R. C. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419–457.

McClelland, J. L., Mickey, K., Hansen, S., Yuan, A., and Lu, Q. (2016). A Parallel-Distributed Processing Approach to Mathematical Cognition.

McClelland, J. L., Mirman, D., Bolger, D. J., and Khaitan, P. (2014). Interactive activation and mutual constraint satisfaction in perception and cognition. *Cognitive Science*, 38(6):1139–1189.

McClelland, J. L. and Plaut, D. C. (1999). Does generalization in infant learning implicate abstract algebra-like rules? *Trends in Cognitive Sciences*, 3(5):166–168.

McCloskey, M. and Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation*, 24.

McNeil, N. M. and Alibali, M. W. (2005). Why won't you change your mind? Knowledge of operational patterns hinders learning and performance on equations. *Child development*, 76(4):883–899.

Medin, D. L. and Schaffer, M. M. (1978). Context theory of classification learning. *Psychological Review*, 85(3):207–238.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013). Linguistic regularities in continuous space word representations. *Proceedings of NAACL-HLT*, (June):746–751.

Mitchell, T., Cohen, W., Hruschka, E., Talukdar, P., Betteridge, J., Carlson, A., Dalvi, B., Gardner, M., Kisiel, B., Krishnamurthy, J., Lao, N., Mazaitis, K., and Al., E. (2018). Never-Ending Learning. *Communications of the ACM*, pages 2302–2310.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. a., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Mollica, F., Diachek, E., Mineroff, Z., Kean, H., Siegelman, M., Piantadosi, S. T., and Futrell, R. (2020). Composition is the core driver of the language-selective network. *Neurobiology of Language*, pages 1–28.

Momennejad, I., Russek, E. M., Cheong, J. H., Botvinick, M. M., Daw, N. D., and Gershman, S. J. (2017). The successor representation in human reinforcement learning. *Nature Human Behaviour*, 1(9):680–692.

Monsell, S. (2003). Task switching. *Trends in Cognitive Sciences*, 7(3):134–140.

Mu, J., Liang, P., and Goodman, N. (2019). Shaping visual representations with language for few-shot classification. *Visually Grounded Interaction and Language Workshop, NeurIPS.*

Munkhdalai, T. and Yu, H. (2017). Meta Networks. *arXiv preprint.*

Nagabandi, A., Finn, C., and Levine, S. (2019). Deep online learning via meta-learning: Continual adaptation for model-based RL. *International Conference on Learning Representations*, pages 1–15.

Nathan, M. J. (2008). An embodied cognition perspective on symbols, gesture, and grounding instruction. In *Symbols and embodiment: Debates on meaning and cognition*, volume 18, pages 375–396.

Newell, a. and Simon, H. a. (1961). Computer simulation of human thinking. *Science (New York, N.Y.)*, 134(3495):2011–2017.

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. (2018). Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks. *arXiv preprint.*

Ng, A. Y. and Russell, S. (2000). Algorithms for Inverse Reinforcement Learning. In *International Conference on Machine Learning.*

Nichol, A., Achiam, J., and Schulman, J. (2018). On First-Order Meta-Learning Algorithms. *arXiv preprint*, pages 1–15.

Niv, Y. (2009). Reinforcement learning in the brain. *Journal of Mathematical Psychology.*

Norouzi, M., Mikolov, T., Bengio, S., Singer, Y., Shlens, J., Frome, A., Corrado, G. S., and Dean, J. (2014). Zero-shot learning by convex combination of semantic embeddings. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, pages 1–9.

Núñez, R. E. (2011). No innate number line in the human brain. *Journal of Cross-Cultural Psychology*, 42(4):651–668.

O'Doherty, J. P., Dayan, P., Friston, K., Critchley, H., and Dolan, R. J. (2003). Temporal difference models and reward-related learning in the human brain. *Neuron*, 38(2):329–337.

Oh, J., Singh, S., Lee, H., and Kohli, P. (2017). Zero-Shot Task Generalization with Multi-Task Deep Reinforcement Learning. *Proceedings of the 34th International Conference on Machine Learning.*

OpenAI, Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pinto, H. P. d. O., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., and Zhang, S. (2019). Dota 2 with Large Scale Deep Reinforcement Learning. *arXiv preprint.*

Oswald, J. V., Henning, C., Sacramento, J., and Grewe, B. F. (2020). Continual learning with hypernetworks. *International Conference on Learning Representations*, pages 1–25.

Pal, A. and Balasubramanian, V. N. (2019). Zero-Shot Task Transfer. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Park, J. and Brannon, E. M. (2013). Training the Approximate Number System Improves Math Proficiency. *Psychological Science*, 24(10):2013–2019.

Patel, P. and Varma, S. (2018). How the Abstract Becomes Concrete: Irrational Numbers Are Understood Relative to Natural Numbers and Perfect Squares. *Cognitive Science*, 42(5):1642–1676.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Pérez, G. V., Camargo, C. Q., and Louis, A. A. (2019). Deep learning generalizes because the parameter-function map is biased towards simple functions. In *International Conference on Learning Representations.*

Petroski, F., Vashisht, S., Edoardo, M., Joel, C., Kenneth, L., and Jeff, O. S. (2018). Deep Neuroevolution : Genetic Algorithms are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning. *arXiv preprint.*

Platanios, E. A., Sachan, M., Neubig, G., and Mitchell, T. M. (2017). Contextual Parameter Generation for Universal Neural Machine Translation. *arXiv preprint*.

Potts, C. (2019). A case for deep learning in semantics: Response to pater. *Language*, 95(1):e115–e124.

Racaniere, S., Lampinen, A. K., Santoro, A., Reichert, D. P., Firoiu, V., and Lillicrap, T. P. (2020). Automated curricula through setter-solver interactions. *International Conference on Learning Representations*, pages 1–17.

Racanière, S., Weber, T., Reichert, D. P., Buesing, L., Guez, A., Rezende, D., Badia, A. P., Vinyals, O., Heess, N., Li, Y., Pascanu, R., Battaglia, P., Hassabis, D., Silver, D., and Wierstra, D. (2017). Imagination-augmented agents for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2017-Decem(Nips):5691–5702.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *arXiv preprint*.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv preprint, arXiv:1910.10683*, pages 1–53.

Rao, D., Visin, F., Rusu, A. A., Teh, Y. W., Pascanu, R., and Hadsell, R. (2019). Continual Unsupervised Representation Learning. *Advances in Neural Information Processing Systems*.

Ravi, S. and Larochelle, H. (2017). Optimization as a Model for Few-Shot Learning. *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*, pages 1–11.

Ravichandran, A., Bhotika, R., and Soatto, S. (2019). Few-Shot Learning with Embedded Class Models and Shot-Free Meta Training. *arXiv preprint*.

Reed, S., Chen, Y., Paine, T., van den Oord, A., Eslami, S. M. A., Rezende, D., Vinyals, O., and de Freitas, N. (2017). Few-shot Autoregressive Density Estimation: Towards Learning to Learn Distributions. (2016):1–11.

Reed, S. and de Freitas, N. (2015). Neural Programmer-Interpreters. *arXiv preprint*, pages 1–12.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., and Zemel, R. S. (2018). Meta-Learning for Semi-Supervised Few-Shot Classification. pages 1–15.

Richland, L. E., Stigler, J. W., and Holyoak, K. J. (2012). Teaching the Conceptual Structure of Mathematics. *Educational Psychologist*, 47(3):189–203.

Rittle-Johnson, B., Siegler, R. S., and Alibali, M. W. (2001). Developing conceptual understanding and procedural skill in mathematics: an iterative process. *Journal of Educational Psychology*, 93(2).

Rogers, R. D. and Monsell, S. (1995). Costs of a Predictable Switch Between Simple Cognitive Tasks. *Journal of Experimental Psychology: General*, 124(2):207–231.

Rogers, T. T. and McClelland, J. L. (2004). *Semantic Cognition: A Parallel Distributed Processing Approach*. MIT Press.

Rogers, T. T. and McClelland, J. L. (2008). Precis of Semantic Cognition : A Parallel Distributed Processing Approach. *Behavioral and Brain Sciences*, 31:689–749.

Rohde, D. and Plaut, D. (1997). Simple recurrent networks and natural language: How important is starting small. *Proceedings of the 19th annual conference of the Cognitive Science Society*, pages 656–661.

Romera-Paredes, B. and Torr, P. H. (2015). An embarrassingly simple approach to zero-shot learning. *32nd International Conference on Machine Learning, ICML 2015*, 3:2142–2151.

Rosch, E., Mervis, C. B., Gray, W. D., Johnson, D. M., and Boyes-Braem, P. (1976). Basic Objects in Natural categories. *Cognitive Psychology*, 8:382–439.

Rosenthal, D. M. (1990). A Theory of Consciousness. *Zentrum für interdisziplinäre Forschung*.

Rumelhart, D. E. and Todd, P. M. (1993). Learning and connectionist representations. *Attention and performance XIV: Synergies in experimental psychology, artificial intelligence, and cognitive neuroscience*, pages 3–30.

Rusu, A. A., Gomez Colmenarejo, S., Gulcehre, C., Desjardins, G., Kirkpatrick, J., Pascanu, R., Mnih, V., Kavukcuoglu, K., and Hadsell, R. (2015). Policy Distillation. *arXiv*, pages 1–12.

Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., and Hadsell, R. (2019). Meta-Learning with Latent Embedding Optimization. *International Conference on Learning Representations*, pages 1–17.

Salimans, T. and Kingma, D. P. (2016). Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. *Advances in Neural Information Processing Systems*.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. (2016). Meta-Learning with Memory-Augmented Neural Networks. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48.

Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. (2017). A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems*.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., and Cox, D. D. (2018). On the Information Bottleneck Theory of Deep Learning. In *International Conference on Learning Representations*, pages 1–27.

Saxe, A. M., Mcclelland, J. L., and Ganguli, S. (2019). A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*.

Schapiro, A. C. and Mcclelland, J. L. (2009). A connectionist model of a continuous developmental transition in the balance scale task. *Cognition*, 110:395–411.

Schwartz, D. L. and Goldstone, R. L. (2015). Learning as Coordination. In *Handbook of educational psychology*, chapter 5. 3rd edition.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. (2017). Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. pages 1–19.

Shenhav, A., Botvinick, M. M., and Cohen, J. D. (2013). The Expected Value of Control: An Integrative Theory of Anterior Cingulate Cortex Function. *Neuron*, 79(2):217–240.

Shwartz-Ziv, R. and Tishby, N. (2017). Opening the Black Box of Deep Neural Networks via Information. *arXiv*.

Siegelman, H. T. and Sontag, E. D. (1992). On the computational power of neural nets. *Proceedings of the fifth annual workshop on computational learning theory*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Driessche, G. V. D., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., and Kavukcuoglu, K. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7585):484–489.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., Van Den Driessche, G., Graepel, T., and Hassabis, D. (2017). Mastering the game of go without human knowledge. *Nature*, 550.

Singley, M. K. and Anderson, J. R. (1989). *The transfer of cognitive skill*.

Smith, L. B. and Slone, L. K. (2017). A developmental approach to machine learning? *Frontiers in Psychology*, 8(DEC):1–10.

Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46(1-2):159–216.

Smolensky, P. and Goldrick, M. (2014). Optimization and Quantization in Gradient Symbol Systems : A Framework for Integrating the Continuous and the Discrete in Cognition. *Cognitive Science*, 38:1102–1138.

Sobel, D. M., Tenenbaum, J. B., and Gopnik, A. (2004). Children's causal inferences from indirect evidence: Backwards blocking and Bayesian reasoning in preschoolers. *Cognitive Science*, 28(3):303–333.

Socher, R., Ganjoo, M., Manning, C. D., and Ng, A. Y. (2013). Zero-shot learning through cross-modal transfer. *Advances in Neural Information Processing Systems*.

Sprechmann, P., Jayakumar, S. M., Rae, J. W., Pritzel, A., Uria, B., Vinyals, O., Hassabis, D., Pascanu, R., and Blundell, C. (2018). Memory-based parameter Adaptation. In *International Conference on Learning Representations*.

Stadie, B. C., Yang, G., Houthooft, R., Chen, X., Duan, Y., Wu, Y., Abbeel, P., and Sutskever, I. (2018). Some Considerations on Learning to Explore via Meta-Reinforcement Learning. *arXiv preprint*.

Stroop, J. R. (1935). Studies of interference in serial verbal reactions. *Journal of Experimental Psychology*, 18(6):643–662.

Such, F. P., Rawal, A., Lehman, J., Stanley, K. O., and Clune, J. (2019). Generative teaching networks: accelerating neural architecture search by learning to generate synthetic training data. *arXiv preprint*, pages 1–26.

Sutton, R. (2019). The bitter lesson.

Sutton, R. S. and Barto, A. G. (2017). *Reinforcement learning: An introduction*.

Sutton, R. S., Modayil, J., Degris, M. D. T., Pilarski, P. M., White, A., and Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. *10th International Conference on Autonomous Agents and Multiagent Systems 2011, AAMAS 2011*, 2(1972):713–720.

Sutton, R. S., Precup, D., and Singh, S. (1999). Between MDPs and semi-MDPs : A framework for temporal abstraction in reinforcement learning. 112:181–211.

Szabó, Z. G. (2017). Compositionality. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2017 edition.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June:1–9.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. In *ICLR*.

Talvitie, E. (2014). Model regularization for stable sample rollouts. *Uncertainty in Artificial Intelligence - Proceedings of the 30th Conference, UAI 2014*, pages 780–789.

Tamar, A., Wu, Y., Thomas, G., Levine, S., and Abbeel, P. (2017). Value iteration networks. *Neural Information Processing Systems*, (Nips):4949–4953.

Tanenhaus, M. K. and Lucas, M. M. (1987). Context effects in lexical processing. *Cognition*, 25:213–234.

Tessler, C., Givony, S., Zahavy, T., Mankowitz, D. J., and Mannor, S. (2016). A Deep Hierarchical Approach to Lifelong Learning in Minecraft. *arXiv preprint*.

Tishby, N. and Zaslavsky, N. (2015). Deep Learning and the Information Bottleneck Principle. In *IEEE Information Theory Workshop (ITW)*.

Tomasello, M., Kruger, A. C., and Ratner, H. H. (1993). Cultural Learning. *Behavioral and Brain Sciences*, (16).

Tunney, R. J. and Altmann, G. T. (2001). Two Modes of Transfer in Artificial Grammar Learning. *Journal of Experimental Psychology: Learning Memory and Cognition*, 27(3):614–639.

Turchetta, M., Berkenkamp, F., and Krause, A. (2016). Safe exploration in finite Markov decision processes with Gaussian processes. *Advances in Neural Information Processing Systems*, pages 4312–4320.

Turchetta, M., Berkenkamp, F., and Krause, A. (2019). Safe Exploration for Interactive Machine Learning. *Advances in Neural Information Processing Systems.*

Van Damme, R., Wilson, R. S., Vanhooyconck, B., and Aerts, P. (2002). Rational imitation in preverbal infants. *Nature*, 415(6873):755.

Vankov, I. I., Bowers, J. S., and Vankov, I. I. (2019). Training neural networks to encode symbols enables combinatorial generalization. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 375(1791).

Vapnik, V. and Chervonenkis, A. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. *arXiv*.

Veeriah, V., Hessel, M., Xu, Z., Lewis, R., Rajendran, J., Oh, J., van Hasselt, H., Silver, D., and Singh, S. (2019). Discovery of Useful Questions as Auxiliary Tasks. *Advances in Neural Information Processing Systems*, (NeurIPS):1–12.

Velez, R. and Clune, J. (2017). Diffusion-based neuromodulation can eliminate catastrophic forgetting in simple neural networks. *PLoS ONE*, 12.

Ven, G. M. V. D. and Tolias, A. S. (2018). Three scenarios for continual learning. *NeurIPS Continual Learning Workshop*, pages 1–18.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., Vezhnevets, A. S., Leblond, R., Pohlen, T., Dalibard, V., Budden, D., Sulsky, Y., Molloy, J., Paine, T. L., Gulcehre, C., Wang, Z., Pfaff, T., Wu, Y., Ring, R., Yogatama, D., Wünsch, D., McKinney, K., Smith, O., Schaul, T., Lillicrap, T., Kavukcuoglu, K., Hassabis, D., Apps, C., and Silver, D. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (2016). Matching Networks for One Shot Learning. *Advances in Neural Information Processing Systems*.

Vygotsky, L. (1934). *Thought and Language*. MIT Press, 1986 edition.

Wakefield, E., Novack, M. A., Congdon, E. L., Franconeri, S., and Goldin-Meadow, S. (2018). Gesture helps learners learn, but not merely by guiding their visual attention. *Developmental Science*, (February):1–12.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. (2019a). SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. *arXiv preprint*, 2019(July):1–29.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. (2019b). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. (2016). Learning to reinforcement learn. *arXiv preprint*, pages 1–17.

Weber, K. (2001). Student difficulty in constructing proofs: The need for strategic knowledge. *Educational Studies in Mathematics*, 48(1):101–119.

Wilensky, U. (1991). Abstract Meditations on the Concrete and Concrete Implications for Mathematics Education. In Harel, I. and Papert, S., editors, *Constructionism*. Ablex Publishing, Westport, CT, US.

Wolpert, D. H. (1996). The Lack of a Priori Distinctions between Learning Algorithms. *Neural Computation*, 8(7):1391–1420.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, Ł., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv*, pages 1–23.

Wu, Z., Xiong, Y., Yu, S. X., and Lin, D. (2018). Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Xian, Y., Lampert, C. H., Schiele, B., and Akata, Z. (2018). Zero-Shot Learning - A Comprehensive Evaluation of the Good, the Bad and the Ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–14.

Xu, D., Nair, S., Zhu, Y., Gao, J., Garg, A., Fei-Fei, L., and Savarese, S. (2017). Neural Task Programming: Learning to Generalize Across Hierarchical Tasks.

Xu, Z., van Hasselt, H., and Silver, D. (2018). Meta-Gradient Reinforcement Learning. In *Advances in Neural Information Processing Systems*.

Yamins, D. L. K. and Dicarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, 19(3):356–365.

Yamins, D. L. K., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D., and DiCarlo, J. J. (2014). Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences of the United States of America*, 111(23):8619–8624.

Yao, H., Wei, Y., Huang, J., and Li, Z. (2019). Hierarchically Structured Meta-learning. *Proceedings of the 36th International Conference on Machine Learning*.

Zadrozny, W. (1992). On compositional semantics. *Proceedings of the 14th conference on Computational Linguistics*, 1.

Zaheer, M., Kottur, S., Ravanbhakhsh, S., Poczos, B., Salakhutdinov, R., and Smola, A. J. (2017). Deep Sets. *Advances in Neural Information Processing Systems*.

Zaremba, W. and Sutskever, I. (2014). Learning to Execute. *International Conference on Learning Representations*, pages 1–25.

Zenke, F., Poole, B., and Ganguli, S. (2017). Continual Learning Through Synaptic Intelligence. In *Proceedings of the 34th International Conference on Machine Learning*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint*.

Zhang, C., Ren, M., Urtasun, R., Advanced, U., and Group, T. (2019). Graph HyperNetworks for neural architecture search. In *International Conference on Learning Representations*, number 2018, pages 1–17.

Zhou, Z. and Firestone, C. (2019). Humans can decipher adversarial images. *Nature Communications*, 10(1).

Zintgraf, L. M., Shiarlis, K., Kurin, V., Hofmann, K., and Whiteson, S. (2018). Fast Context Adaptation via Meta-Learning. *Proceedings of the 36th International Conference on Machine Learning*.

Zoph, B. and Le, Q. V. (2016). Neural Architecture Search with Reinforcement Learning. *arXiv preprint*, pages 1–16.