Do LSTMs Learn Compositionally?

Anonymous submission

Abstract

LSTM-based language models exhibit compositionality in their representations, but how this behavior emerges over the course of training has not been explored. Analyzing synthetic data experiments with contextual decomposition, we find that LSTMs learn long-range dependencies compositionally by building them from shorter constituents during training.

1 Introduction

Consider the process of backpropagation through time for a language model. As an example, the language model should learn that an occurrence of "either" increases the later likelihood of "or". To do so, it must backpropagate information from the occurrence of "or" through some intervening constituent, which we will refer to as a **conduit** because the association of either/or is carried through it to affect the representation of "either". Perhaps it encounters a training example that uses a conduit that is predictable by being structured in familiar ways, here italicized: "Either *Socrates is mortal* or not all men are mortal." However, what if the conduit is unpredictable and the structure cannot be interpreted by the model, for example, if the conduit includes unknown tokens, as in: "Either *slithy toves gyre* or mome raths outgrabe"? Which conduit will carry the gradient from "or" to "either" easily?

Formally, as the gradient of the error e_t at timestep t is backpropagated k timesteps through the hidden state h:

$$\frac{\partial e_t}{\partial h_{t-k}} = \frac{\partial e_t}{\partial h_t} \prod_{i=1}^{k} \frac{\partial h_{t-i+1}}{\partial h_{t-i}}$$

The backpropagated message is multiplied repeatedly by the gradients associated with each item in the conduit. If the recurrence derivatives $\frac{\partial h_{i+1}}{\partial h_i}$ are large at some parameter, the correspondingly larger backpropagated gradient $\frac{\partial e_t}{\partial h_{t-k}}$ will accelerate descent in that direction.

When we ask which conduit will carry the gradient message to learn a long-range dependency faster, the answer will depend on the magnitude and distribution of the recurrence gradients. If the language model relies on linguistic structure in the conduit in order to pass the message effectively, then the more predictable conduit will facilitate learning a long-range pattern.

In order to investigate whether long-range dependencies are built from short constituents, we train models on synthetic data which varies the predictability of short sequences. We find that memorizing local patterns allows a language model to learn a long-range dependency faster but ultimately inhibits its ability to fully acquire long-range rules.

2 Related Work

How do neural language models learn? The key to answering this question is to understand the compositionality of LSTM training. To this end, we connect the hierarchical structure of language model representations with the incremental nature of neural learning dynamics.

We have extensive evidence that hierarchical structure is integral to the high performance of fully trained neural language models. LSTMs learn more effectively from natural language data than from similarly distributed random data, implying that they take advantage of linguistic structure (Liu et al., 2018). The representations they produce seem to be hierarchical in nature (Hewitt and Manning, 2019; Blevins et al., 2018; Hupkes et al., 2017). They implicitly exhibit a number of compositionality assumptions linguists tend to make by encoding information about part of speech (Belinkov et al., 2017), morphology (Vania and Lopez, 2017), and verb agreement (Lakretz et al., 2019). But the compositional nature of these representations tells us little about the process by which they are learned.

Humans learn by memorizing short rote phrases and later mastering the ability to construct deep syntactic trees from them (Lieven and Tomasello, 2008). LSTM models, meanwhile, learn by backpropagation through time, leading to different inductive priors compared to a human. We may not therefore expect an LSTM to exhibit similarly compositional learning behavior. However, language models are known to encode hierarchical syntax, so we must consider whether they learn hierarchically as well, by building longer constituents out of shorter ones during training.

Recognizing the role of inductive priors in training is critical. LSTMs have the theoretical capacity to encode a wide range of context-sensitive languages, but in practice their ability to learn such rules from data is limited, implicating the importance of the training process (Weiss et al., 2018). However, we may find that the hierarchical nature of the representation is entirely a result of the data, rather than induced by the biases of the training process. LSTMs by default learn associations from the most recent items in a sequence, but they are still capable of learning to encode grammatical inflection from the first word in a sequence rather than the most recent (Ravfogel et al., 2019). Inductive priors play a critical role in the ability of an LSTM to learn effectively, but they are neither necessary nor sufficient in determining what the model can learn.

We therefore investigate further into LSTM learning dynamics. In general, work in deep learning has supported the assumption that easy examples are learned before hard examples (Arpit et al., 2017). A controversial proposal by Shwartz-Ziv and Tishby (2017) held that learning begins with a memorization phase followed by a compression phase which makes the model more general, a claim that has been extensively debated with evidence for (Noshad and Hero III, 2018) and against (Saxe et al., 2018) it. If the hypothesis holds generally, the transition from memorized to compressed rules is another example of, or potential explanation for, easy-first learning.

In the case of an LSTM, dependency range is one aspect of difficulty that might affect the order of learning. For example, an either/or matching over a short distance can be memorized, but over a long distance requires an encoding of concepts like constituency in order to be applied generally. The learning dynamics of an LSTM cause lower layers to converge faster than higher layers when there are many layers (Raghu et al., 2017), which combined with findings of hierarchy (Blevins et al., 2018; Belinkov et al., 2018) imply that the local connections encoded by lower layers are learned before the more distant connections encoded by higher layers. Even within a single layer, Saphra and Lopez (2019) found that local properties, such as syntactic tags, were learned earlier than the less local property of topic. The transition from local dependencies to more global dependencies is yet another example of how simple patterns are required before complex ones.

However, even if simple local rules are learned first, they might not be used compositionally in constructing longer rules. In fact, simple rules learned early on might inhibit the learning of more complex rules through the phenomenon of gradient starvation (Combes et al., 2018), in which more frequent features reduce the gradient directed at rarer features. Simple local rules could slow down the training process by affecting the recurrence from timestep to timestep to degrade the gradient, or by trapping the model in a local minimum which makes the long-distance rule harder to reach. The compositional view of training is not a given and must be verified.

3 Methods

All experiments use a one layer LSTM, with inputs taken from an embedding layer and outputs processed by a softmax layer. All hidden dimensions are 200. We train with a learning rate set at 1 throughout and gradients clipped at 0.25. We found momentum and weight decay to slow rule learning in this setting, so they are not used.

3.1 Contextual Decomposition

In our running example, we need to determine when our language model has learned that "either" implies an appearance of "or" later in the sequence. It is difficult to directly measure the influence of "either" on the later occurrence of "or", so in order to dissect the sequence and understand the impact of individual elements in the sequence, we employ contextual decomposition (CD).

First introduced by Murdoch et al. (2018), CD is a method of looking at the individual influences of words and phrases in a sequence on the output of a recurrent model. CD converts the output vector from an LSTM layer into a sum of relevant (contributed only by the input word or phrase of interest x_γ; represented as v_γ) and irrelevant (contributed by, or involving interactions with, other words in the sequence; represented as v_β) parts, $v = v_\gamma + v_\beta$. Because the individual contributions of the items in a sequence interact in nonlinear ways, it is difficult to disentangle the impact of a specific word or phrase on the label distribution predicted. However, the dynamics of LSTMs are approximately linear in natural settings, as found by Morcos et al. (2018), who used canonical correlation analysis to find close linear projections between the activations at each timestep in a repeating sequence and the activations at the end of the sequence. It is therefore unsurprising that approximation error is low for the CD approach of linearizing the output of a LSTM layer so it can be viewed as the sum of a relevant component and the contributions of the rest of the sequence.

While Murdoch et al. (2018) were primarily interested in analyzing the importance and classification tendencies of the phrases and words that formed a sequence, we are interested in understanding whether a dependency between two words has been learned at all. Because the decomposed logits can be used as inputs for a softmax, we convert the decomposed elements into probability distributions by $P(x|x_\gamma) = \text{softmax}(v_\gamma)$. This allows us to analyze the effect of x_γ on a later element x while controlling for the influence of the rest of the sequence. We consider the running either-or example dependency to have been effectively learned when the contribution of the opening token ('either') places a high probability on its mate ('or') at the appropriate timestep when 'or' occurs in the data.

4 Experiments

We use synthetic data to test the ability of an LSTM to learn a consistent rule with a long-distance dependency. This controls for the irregularity of natural language as well as for the confounding factor of rule frequency. While LSTMs in natural language model settings learn short-range dependencies first, we must consider the possibility that this pattern is unrelated to any inductive prior. It could be that longer-range dependencies are simply rarer and therefore learned later. Our synthetic data sets instead have a fixed number of occurrences of the long distance relationship, regardless of the conduit length.

We generate data uniformly at random from a vocabulary Σ. However, we insert n instances of the long-distance rule $\alpha\Sigma^k\omega$ (with conduit length k), where we consider an **open symbol** α and a **close symbol** ω, with $\alpha, \omega \notin \Sigma$. Relating to our running example, α stands for "either" and ω stands for "or". We use a corpus of 1m tokens with $|\Sigma| = 1$k types, which leaves a low probability that any conduit sequence longer than 1 token appears elsewhere by chance.

For all analyses, CD yielded an approximation error $\frac{\|(v_\gamma+v_\beta)-v\|}{\|v\|} < 10^{-5}$, when running the true LSTM model to generate v for comparison.

Limitations While we hope to isolate the role of long range dependencies through synthetic data, we must consider the possibility that the natural predictability of language data differs in relevant ways from the synthetic data, in which the conduits are predictable only through pure memorization. Because LSTM models take advantage of linguistic structure, we cannot be confident that predictable natural language exhibits the same cell state dynamics that make a memorized uniformly sampled conduit promote or inhibit long-range rule learning.

4.1 The Effect of Rule Frequency

First, we investigate how the frequency of a rule affects the ability of the model to learn the rule. We vary the conduit length k while keeping n constant. The results in Figure 1 illustrate how a longer conduit length requires more examples before the model can learn the corresponding rule. We consider the probability assigned to the close symbol according to the contributions of the open symbol, excluding interaction from any other token in the sequence. For contrast, we also show the extremely low probability assigned to the close symbol according to the contributions of the conduit taken as an entire phrase. In particular, note the pattern when the rule is extremely rare: The probability of the close symbol as determined by the open symbol is low but steady regardless of the conduit length, while the probability as determined by the conduit declines with conduit length due to the accumulated low probabilities from

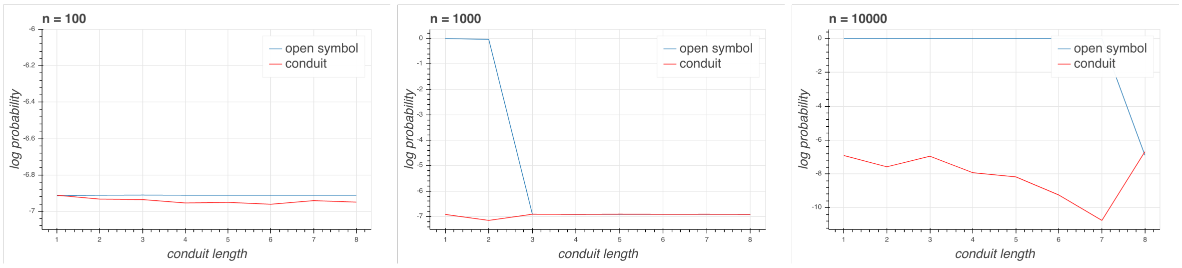

Figure 1: The predicted probability $P(x_t = \omega)$, according to the contributions of open symbol $x_{t-k} = \alpha$ and of the conduit sequence $x_{t-k+1} \dots x_{t-1}$, for various rule occurrence counts n. Shown at 40 epochs.

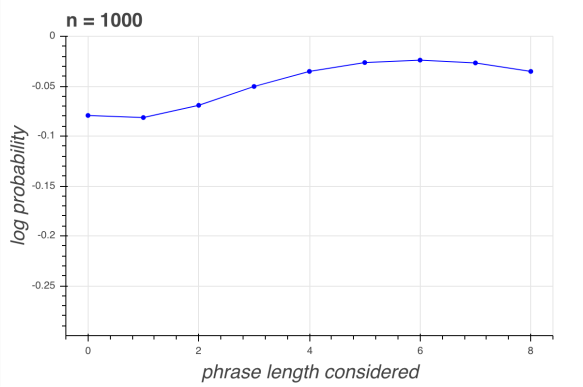

Figure 2: The predicted $P(x_t = \omega | x_{t-k} \dots x_{t-k+i})$ according to CD, varying i as the x-axis and with $x_{t-k} = \alpha$ and $k = 8$. Shown at 50 epochs.

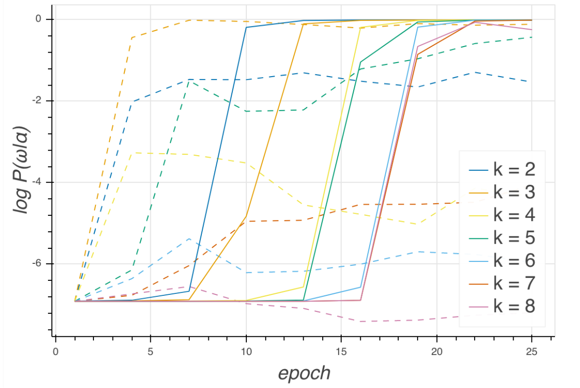

Figure 3: The predicted $P(x_t = \omega | x_{t-k} = \alpha)$, according to CD. Solid lines are in the unpredictable conduit setting, while dashed lines are in the predictable conduit setting.

each element in the sequence.

To understand the impact of the open symbol in context, see Figure 2, which illustrates that the conduit interacts with the open symbol to increase the probability slightly, a sign that the model is counting the intervening symbols rather than registering only the effect of the open symbol.

4.2 The Effect of Conduit Predictability

To understand the effect of conduit predictability, we modify the synthetic data such that the sequence in the conduit appears frequently outside of the long-distance rule. In this experiment, the conduits are actually taken from a randomly generated vocabulary of 100, so that each unique conduit q appears in the training corpus 10 times in the context $\alpha q \omega$. This repetition is necessary in order to fit $n = 1000$ occurrences of the rule in all settings. In the **unpredictable-conduit setting**, q appears only in this context as a conduit, so the conduit remains random and unpredictable. In the **predictable-conduit setting**, we randomly distribute $m = 1000$ occurrences of each conduit q throughout the corpus outside of the rule patterns. In the predictable-conduit setting, each conduit is seen often enough to be memorized.

As we see in Figure 3, copying each conduit 100 times throughout the corpus inhibits learning of the symbol-matching rule over the long run of training, but promotes early learning of the rule. This result implies that long-range dependencies are learned from the structure of their constituents. Therefore the model is delayed during training in representing longer dependencies in part because it depends on constituents being effectively learned first.

5 Conclusions

We confirm that the longer the span of a rule, the more examples are required for an LSTM model to effectively learn the rule. We then find1 that a more predictable conduit between the rule symbols promotes the early learning of the rule, implying that the process by which an LSTM learns long-range rules is compositional. However, the representation learned through the predictable conduit ultimately prevents the model from confidently learning these long-range connections.

References

Devansh Arpit, Stanisaw Jastrzbski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A Closer Look at Memorization in Deep Networks. *arXiv:1706.05394 [cs, stat]*. ArXiv: 1706.05394.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do Neural Machine Translation Models Learn about Morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Yonatan Belinkov, Llus Mrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James R. Glass. 2018. Evaluating Layers of Representation in Neural Machine Translation on Part-of-Speech and Semantic Tagging Tasks. *CoRR*, abs/1801.07772.

Terra Blevins, Omer Levy, and Luke Zettlemoyer. 2018. Deep RNNs Encode Soft Hierarchical Syntax. *arXiv:1805.04218 [cs]*. ArXiv: 1805.04218.

Remi Tachet des Combes, Mohammad Pezeshki, Samira Shabanian, Aaron Courville, and Yoshua Bengio. 2018. On the Learning Dynamics of Deep Neural Networks. *arXiv:1809.06848 [cs, stat]*. ArXiv: 1809.06848.

John Hewitt and Christopher D Manning. 2019. A Structural Probe for Finding Syntax in Word Representations. In *NAACL*.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2017. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *arXiv:1711.10203 [cs]*. ArXiv: 1711.10203.

Yair Lakretz, German Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 2019. The emergence of number and syntax units in LSTM language models. *arXiv:1903.07435 [cs]*. ArXiv: 1903.07435.

Elena Lieven and Michael Tomasello. 2008. *Children's first language acquisition from a usage-based perspective*. Routledge.

Nelson F. Liu, Omer Levy, Roy Schwartz, Chenhao Tan, and Noah A. Smith. 2018. LSTMs Exploit Linguistic Attributes of Data. *arXiv:1805.11653 [cs]*. ArXiv: 1805.11653.

Ari S. Morcos, Maithra Raghu, and Samy Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. *arXiv:1806.05759 [cs, stat]*. ArXiv: 1806.05759.

W. James Murdoch, Peter J. Liu, and Bin Yu. 2018. Beyond Word Importance: Contextual Decomposition to Extract Interactions from LSTMs. In *ICLR*.

Morteza Noshad and Alfred O. Hero III. 2018. Scalable Mutual Information Estimation using Dependence Graphs. *arXiv preprint arXiv:1801.09125*.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. 2017. SVCCA: Singular Vector Canonical Correlation Analysis for Deep Learning Dynamics and Interpretability. *arXiv:1706.05806 [cs, stat]*. ArXiv: 1706.05806.

Shauli Ravfogel, Yoav Goldberg, and Tal Linzen. 2019. Studying the Inductive Biases of RNNs with Synthetic Variations of Natural Languages. *arXiv:1903.06400 [cs]*. ArXiv: 1903.06400.

Naomi Saphra and Adam Lopez. 2019. Understanding Learning Dynamics Of Language Models with SVCCA. In *NAACL*. ArXiv: 1811.00225.

Andrew Michael Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan Daniel Tracey, and David Daniel Cox. 2018. On the Information Bottleneck Theory of Deep Learning. *ICLR*.

Ravid Shwartz-Ziv and Naftali Tishby. 2017. Opening the Black Box of Deep Neural Networks via Information. *arXiv:1703.00810 [cs]*. ArXiv: 1703.00810.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? In *ACL*.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. On the Practical Computational Power of Finite Precision RNNs for Language Recognition. *arXiv:1805.04908 [cs, stat]*. ArXiv: 1805.04908.