Limitations in learning an interpreted language with recurrent models

Denis Paperno
Loria, Centre national de la recherche scientifique (CNRS)
Campus Scientifique, BP 239, 54506 Vandoeuvre-lès-Nancy, France
paperno@ucla.edu

Abstract

In this submission I report work in progress on learning simplified interpreted languages by means of recurrent models. The data is constructed to reflect core properties of natural language as modeled in formal syntax and semantics. Preliminary results suggest that LSTM networks do generalise to compositional interpretation, albeit only in the most favorable learning setting.

Motivation. Despite showing impressive performance on certain tasks, neural networks are still far from showing natural language understanding at a human level, cf. Paperno et al. (2016). In a sense, it is not even clear what kind of neural architecture is capable of learning natural language semantics in all its complexity, with recurrent and convolutional models being currently tried on various tasks.

One can hope to make progress towards the challenging goal of natural language understanding by taking into account what is known about language structure and language processing in humans. With this in mind, it is possible to formulate certain preliminary desiderata for an adequate natural language understanding model.

First, language processing in humans is known to be sequential; people process and interpret linguistic input on the fly, without any lookahead and without waiting for the linguistic structure to be completed. This property, which has serious potential consequences for the cognitive architecture (Christiansen and Chater, 2016), gives a certain degree of cognitive plausibility to unidirectional recurrent models compared to other neural architectures, at last in their current implementations.

Second, natural language can exploit recursive structures: natural language syntax consists of constructions, represented in formal grammars as rewrite rules, which can recursively embed other constructions of the same kind. For example, noun phrases can in principle consist of a single proper noun (e.g. *Ann*) but can also, among other possibilities, be built from other noun phrases recursively via the possessive construction, as in *Ann's child*, *Ann's child's friend*, *Ann's child's friend's parent* etc. The possessive construction can be described by the rewrite rule NP $\longrightarrow$ NP*'s* N.

Third, the recursive syntactic structure drives compositional semantic interpretation. The meaning of the noun phrase *Ann's child's friend* is not merely the sum of the meanings of the individual words (in which case it would have been semantically equivalent to *Ann's friend's child*). Rather, to interpret a complex expression correctly, one has to follow the syntactic structure, first identifying the meaning of the smaller constituent (*Ann's friend*), and then computing the meaning of the whole on its basis.

Fourth, semantic compositionality can be formalized as function application, with one constituent in a complex structure corresponding to an argument of a function that another constituent encodes. For instance, in *Ann's child*, we can think of *Ann* as denoting an individual and *child* as denoting a function from individuals to individuals. In formal semantics, function argument application as a semantic compositionality mechanism extends to a wide range of syntactic constructions.

Finally, natural language interpretation, while being sensitive to syntactic structure, is robust to syntactic variation. For example, humans are equally capable of learning to interpret and using left-branching structures such as NP $\longrightarrow$ NP*'s* N (*Ann's child*) and right-branching structures such as NP $\longrightarrow$ *the* N*of* NP (*the child of Ann*).

The task. To summarize, in order to mimic human language capacities, an artificial system has to be able to learn interpreted languages with compositionally interpreted recursive structures, while

Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pages 384–386
Brussels, Belgium, November 1, 2018.

being adaptive to surface variation in the syntactic patterns. To test whether neural systems can fit the bill, we define toy interpreted languages based on a fragment of English. The vocabulary includes four names (*Ann*, *Bill*, *Dick*, *George*), interpreted as individual identifiers, four function-denoting nouns (*child*, *parent*, *friend*, *enemy*), and grammatical elements (*of*, *'s*, *the*). Our languages contain either left-branching (`NP` $\longrightarrow$ `NP` *'s* `N`, *Ann's child*) or right-branching structures (*the child of Ann*, `NP` $\longrightarrow$ *the* `N` *of* `NP`).

The interpretation is defined model-theoretically. We randomly generate a model where each proper name corresponds to a distinct individual and each function denoted by a common noun is total. In such a model, each well-formed expression of the language is interpreted as an individual identifier. The denotation of any expression can be calculated by recursive application of functions to arguments, guided by the syntactic structure of the expression.

The task given to the neural systems is to identify the individual that corresponds to each expression; e.g. *Ann's child's friend* is the same person as *George*. Since there is just a finite number of individuals in our models, this boils down formally to a string classification task, assigning each expression to one of the set of individuals in the model.

Systems and data. We tested two standard systems: a vanilla recurrent neural network (RNN) and a long short-term memory network (LSTM) on the task. Both systems were implemented in PyTorch and used hidden layers of 256 units.

We used all expressions of the language up to complexity n as experimental data; development and testing data was randomly selected among examples of maximal complexity. Examples of smaller complexity, i.e. 1 and 2, were always included in the training partition since they are necessary to learn the interpretation of lexical items. We also set a curriculum whereby the system was at first given training examples of minimal complexity, with more complex examples added gradually in the process of training.

Results and discussion. We found the RNN system to struggle already at a basic level; it never achieved perfect accuracy even for minimally complex structures (e.g. *Ann's child*), so assessing its recursive compositionality abilities is out of question. Accuracies across LSTM experimental setups are summarized in Table 1.

branching	**3**	**4**	**5**	**6**	**7**
right branching	0	.17	.21	.23	.26
left branching	**1**	**1**	**1**	**1**	**1**
left, slow curriculum	.17	.33	.96	**1**	**1**
left, no curriculum	.17	.21	.19	.21	.26

Table 1: System accuracy as a function of the language, curriculum and data complexity. Random baseline is .25.

rec.in train	0.0	0.2	0.4	0.6	0.8
average accuracy	0	.65	.67	.92	**.98**

Table 2: Percentage of complexity 3 data included in training data vs. average test accuracy over 10 runs.

We find that LSTM does learn to do compositional interpretation in our task, but only in the best scenario. First, and unsurprisingly, a curriculum is essential for the LSTM to generalize to unseen compositional examples. Informally, the system has to learn to interpret words first, and recursive semantic composition has to be learned later.

Second, although the recurrent architecture seems naturally adapted for processing complex left-branching structures, the system has to be trained on a lot of examples of composition before it generalizes; cf. Table 2. Unlike (presumably) in humans, recursive compositionality does not come for free and has to be learned from extensive data. This observation goes in line with other findings in related literature (Liska et al., 2018; Hupkes et al., 2018; Lake and Baroni, 2017).

Third, the LSTM only generalized correctly in the case of left-branching structures; for right branching, the accuracy of recursive composition in the end stays just above the chance level (25%). This means that the system only learned to apply composition following the linear sequence of the input and failed when the order of compositionality as determined by the syntactic structure runs opposite to the linear order.

The last two observations suggest that learning recursive structure remains a challenge for LSTM networks, which excel only in sequential, left-to-right processing. If recursion, as has been claimed, is a core distinguishing property of human language and cognition (Hauser et al., 2002; Chomsky, 2014), we may need to ensure that learning systems designed for language incorporate proper biases towards recursive processing.

Acknowledgments

The research has been supported by CNRS PEPS ReSeRVe grant. I also thank Germán Kruszewski for useful input on the topic.

References

Noam Chomsky. 2014. Minimal recursion: exploring the prospects. In *Recursion: Complexity in cognition*, pages 1–15. Springer.

Morten H. Christiansen and Nick Chater. 2016. The now-or-never bottleneck: A fundamental constraint on language. *Behavioral and Brain Sciences*, 39.

Marc D. Hauser, Noam Chomsky, and W. Tecumseh Fitch. 2002. The faculty of language: What is it, who has it, and how did it evolve? *science*, 298(5598):1569–1579.

Dieuwke Hupkes, Anand Singh, Kris Korrel, Germán Kruszewski, and Elia Bruni. 2018. Learning compositionally through attentive guidance. *CoRR*, abs/1805.09657.

Brenden M. Lake and Marco Baroni. 2017. Still not systematic after all these years: On the compositional skills of sequence-to-sequence recurrent networks. *CoRR*, abs/1711.00350.

Adam Liska, Germán Kruszewski, and Marco Baroni. 2018. Memorize or generalize? searching for a compositional RNN in a haystack. *CoRR*, abs/1802.06467.

Denis Paperno, German Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda Torrent, and Raquel Fernandez. 2016. The lambada dataset: word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin: Association for Computational Linguistics*, pages 1525–1534. ACL (Association for Computational Linguistics).