LEARNING HIERARCHICAL SYNTACTIC TRANSFORMATIONS WITH ENCODER-DECODER NETWORKS

Max Nelson
Department of Linguistics
University of Massachusetts Amherst
Amherst, MA, USA
manelson@umass.edu

ABSTRACT

It is hypothesized that in first language acquisition the space of possible grammars considered by children is restricted to only those which are expressed over hierarchical positions (Chomsky, 1980). This work builds on previous work probing the extent to which encoder-decoder networks are able to learn and generalize over hierarchical structure when trained on syntactic transformations (Frank & Mathis, 2007; McCoy et al., 2018). The primary contribution is training of the networks on multiple artificial languages which differ in the extent to which the target transformation can be expressed as a function of the linear or hierarchical position of words. Results suggest that GRU encoder-decoders reliably behave consistently with hierarchical generalization, while SRNNs and LSTMs do not. Contrary to earlier claims, no network behaves as if it has learned a linear generalization even in a language in which all training data are consistent with one.

1 INTRODUCTION

The data to which children are exposed are largely consistent with syntactic rules expressed over the linear positions of words. However there is little evidence that children entertain linear hypotheses for their syntactic grammar (Crain & Nakayama, 1987), leading to the proposal that the children's hypothesis space is restricted to rules expressed over hierarchical structure (Chomsky, 1980). This proposal has been challenged in a number of ways, including work suggesting that an ideal learner will choose a hierarchical rule even when hierarchical and linear hypotheses are considered (Perfors et al., 2011). This work explores the hypothesis space considered by encoder-decoder networks trained on a syntactic transformation task, the *active↔passive* transformation, loosely analogous to the task of human language learners determining which structures are related. This is done by training networks on the transformation in a series of artificial languages in which it can be expressed as a rule over linear position or over hierarchical position that is or is not cued by distributional lexical information.

There is a growing body of work training neural networks on syntactic transformations to identify their inherent learning biases and relating those to biases observed in humans Buz & Frank (2008). Frank and Mathis (2007) train simple recurrent networks on a subset of possible English *declarative→interrogative* transformations which are expressible as either a generalization over linear positions of the words or the underlying syntactic structure. Specifically, they are trained on questions whose main verb is a modal, and the question is formed by fronting this modal, e.g. *The boy <u>can</u> cough→<u>Can</u> the boy cough*. Networks are tested on held out transformations in which there is an embedded modal preceding the matrix modal. Identifying the verb to be fronted requires identifying the hierarchical, rather than linear, positions of the verbs. e.g. *A boy who must love lizards <u>can</u> cough→<u>Can</u> a boy who must love lizards cough? *A boy who <u>must</u> love lizards can cough→<u>Must</u> a boy who love lizards can cough?*. They find that models largely fail at this kind of generalization, however they also do not make errors that indicate they have learned a linear generalization.

McCoy et al. (2018) update the methodology of Frank and Mathis by training modern encoder-decoder networks on similar *declarative→interrogative* transformations. Their findings are mixed – by strict accuracy no model generalizes to held out mappings. However by a more relaxed metric,

Table 1: Sample input and target outputs for the *passive↔active* task.

Input	Keyword	Target Output	Type
1. The dolphin likes the elephant	PASSIVE	The elephant is liked by the dolphin	Transformation
2. The dolphin likes the elephant	ACTIVE	The dolphin likes the elephant	Identity
3. The elephant is liked by the dolphin	PASSIVE	The elephant is liked by the dolphin	Identity
4. The elephant is liked by the dolphin	ACTIVE	The dolphin likes the elephant	Transformation

in which success is determined simply by placing the correct verb in the first position, they find that GRUs appear to make the hierarchical generalization while LSTMs and simple RNNs do not.

2 METHODS

The current work represents a follow up to McCoy et al. (2018). Simple RNN (Elman, 1990), GRU (Cho et al., 2014), and LSTM (Hochreiter & Schmidhuber, 1997) encoder-decoder networks (Sutskever et al., 2014) will be trained on an *active↔passive* transformation task in a series of artificial languages designed to determine both whether models are able to learn and generalize over hierarchical syntactic structure, and, if so, whether they make linear or hierarchical generalizations in the case where both are valid explanations of their training data.

2.1 ACTIVE↔PASSIVE TASK

Networks are trained on identity, *active→passive*, and *passive→active* mappings in a series of artificial languages. A passive can be formed from the active by applying the following rules: **(1)** Move the object noun phrase (NP) to the position currently occupied by the subject noun phrase **(2)** Append *by* to the left edge of the subject NP, move the whole constituent to the position originally occupied by the object NP **(3)** Replace the main verb with its passive counterpart. This requires identifying the subject DP, object DP, and main verb and parsing them into constituents. There may be DPs and verbs which are not in these roles (see §2.2).

Input sentences are followed by a keyword, ACTIVE or PASSIVE. If the keyword matches the voice of the sentence, the task of the network is to map the sentence to itself. If the keyword does not match the voice of the sentence, the task of the network is to transform the sentence into the voice indicated by the keyword.

All nouns can be modified by zero or more adjectives, which introduces variability in the linear relationships between constituents and in the overall length of the sentence without changing the hierarchical structure. During training, adjectives are present only in identity maps, meaning that networks never encounter *active→passive* or *passive→active* transformations with adjectives. Networks are tested both on a test set which exclusively contains non-identity mappings with adjectives, the novel set, and a test set which has the same composition as the training data, the familiar set.

2.2 LANGUAGES

Networks are trained and tested on three artificial languages differing in the extent to which the *passive↔active* mapping is expressible in terms of linear relationships between constituents. Languages are defined by probabilistic context-free grammars (PCFGs) which are used to generate test and training data. All languages have the same number of unique nouns (18), verbs (16), and adjectives (5). The absolute number of unique words varies slightly across languages because different languages require different sets of function words.

The first language is the **linear language**. In this language *passive↔active* transformation without adjectives can be written in terms of linear position in the input string. All sentences are simple transitives. Actives follow the template: DET-1 NOUN-1 VERB DET-2 NOUN-2 and passives the template: DET-2 NOUN-2 *is* VERB *by* DET-1 NOUN-1. In the absence of adjectives, the relationship between actives and corresponding passives can be expressed over the linear position of the words. The linear language represents a conceptual replication of McCoy et al. (2018). Networks are trained on data which can be explained by a generalization over linear position and then tested on data in which only a hierarchical generalization will yield correct predictions.

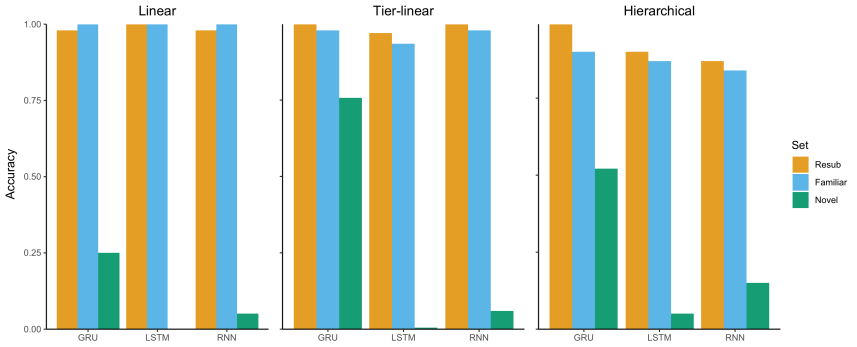

Figure 1: Resubstitution accuracy (orange), accuracy on the held out set with the familiar structures (blue), and accuracy on the held out set with novel structures (green) by language and network type

The second language, the **tier-linear language**, allows for the recursive embedding of object relative clauses (RCs) modifying nouns. All nouns have a probability of 0.1 of being modified by a RC, with no limit on embedding depth. All RCs are marked by overt complementizers (*that*). Crucially, there are two disjoint sets of verbs and nouns, one for matrix clauses and one for embedded clauses. Because of recursively embedded RCs, DPs no longer have fixed length and parsing them into constituents cannot be done based on linear position. However distributional lexical information, namely the fact that certain nouns and verbs never occur in matrix position while others always do, eliminates the problem of having to identify which nouns and verbs are in the matrix clause. One possible way to express the *active*↔*passive* transformation in this language is linear over the tier of words that can occur in the matrix clause.

Finally, the **hierarchical language** is designed to be adversarial for a model which makes linear generalizations. In this language there are embedded clauses, as in the tier-linear language, but there are not distinct sets of lexical items for words in embedded and matrix clauses. There are also prepositional phrases modifying verbs, and adverbial clauses taking the form of entire sentences preceded by an adverb at the left edge of the sentence, i.e. *After the seal ate the nudibranch, the squid hugged the anemone*. Ditransitive verbs are also included and are passivized by placing the indirect object in a right aligned dative phrase. These changes remove any fixed relationship between any of {subject, object, matrix verb} and one another or the edges of the sentence. In this language, even generalizing to novel sequences which have familiar structures requires hierarchical generalization, as there is no other way to characterize the mapping.

2.3 Networks and Training

Encoder-decoder networks with RNN, GRU, and LSTM recurrent layers are tested. All networks have global weighted attention (Bahdanau et al., 2014). Inputs take the form of a series of words, represented by a series of one-hot vectors multiplied through a set of embedding weights. During training strict teacher-forcing is used (Williams & Zipser, 1989). Cross-entropy loss is optimized using Adam (Kingma & Ba, 2014).

During hyperparameter search networks are trained on 15,000 mappings, tested every 5 epochs on a dev set of 2,000 held out mappings. The number of epochs is set as either the last epoch before two consecutive epochs of decreasing accuracy on the dev set or 100. After hyperparameters are set, a new set of 15,000 training mappings is generated, as well as two new test sets: (1) the familiar set, generated in the same fashion as the original dev set, (2) the novel set consisting of 1,200 *active*↔*passive* mappings with adjectives.

3 Results

The main results are shown in Figure 1. Overall no model generalizes well to novel structures (transformations with adjectives), with the possible exception of the GRU in the tier-linear language. Overall novel accuracy is lowest on the linear language.

Certain types of errors on the novel set are consistent and informative across languages. Lexical errors, in which lexical items are replaced by other lexical items of the same class, and adjective errors, in which adjectives are omitted in the input or modify the wrong noun, are both common and not inconsistent with learned hierarchical generalization. While they do suggest a generalization may have been poorly applied, they also generate outputs which are parsable under the context-free grammar that describes the target output language. Common, but less informative, are task errors, in which the output is not in the correct voice.

Table 2: Proportion of errors classifiable as lexical, adjective, or task errors by network and language

	Linear			Tier			Hierarchical		
	Lexical	Adjective	Task	Lexical	Adjective	Task	Lexical	Adjective	Task
SRNN	0.879	0.004	0.063	0.495	0.004	0.0017	0.237	0.027	0.002
GRU	0.973	0.025	0.0	0.443	0.367	0.003	0.417	0.259	0.002
LSTM	0.458	0.004	0.480	0.303	0.048	0.018	0.245	0.058	0.248

In the linear language there are a class of errors which would denote the application of the linear generalization. For example, an input active with the template DET-1 NOUN-1 VERB DET-2 ADJ NOUN-2 may be mapped incorrectly to the passive sequence DET-2 ADJ *is* VERB *by* DET-1 ADJ with NOUN-2 absent in the output because training inputs are exactly 7 words. No linear errors were identified in the predictions of the GRU and RNN and 5% of LSTM predictions were classified as linear errors. While generalization accuracy is low on the linear language, outputs made by the SRNN and GRU are structurally valid outputs and there is little evidence of linear generalization.

To evaluate the extent to which models in the tier language make use of lexical cues rather than syntactic position, models trained on the tier language were also tested on a test set in which the noun tiers were flipped. Here the subject and object DP indicated by syntactic structure are strongly suggested to not be the object and subject by distributional cues. All models achieved an accuracy of 0.0% on this set. However outputs in this task almost categorically have identical structure to the target output, with the exception that matrix nouns are replaced with nouns from the original matrix class and vice versa (GRU-91%, LSTM-90%, RNN-80%).

In the hierarchical language there is no way to express the generalization in the training data other than over syntactic structure, but this does not translate to increased generalization accuracy on the novel set. The SRNN and LSTM fail to generalize and do not make errors consistent with learned syntactic structure. The GRU however achieves 52% accuracy and of the remaining errors, 68% suggest that at least the correct hierarchical structure has been learned.

4 Discussion and Conclusions

McCoy et al. (2018) also report low accuracy in generalizing to a pattern that requires hierarchical generalization. They interpret this to suggest that networks are biased towards making linear generalizations. The results of our linear task corroborate their finding, but our analysis of model predictions does not support the claim that models are making linear generalizations. While the errors may not conclusively suggest that they have learned a generalization over hierarchical structure, it is clear that the RNN and GRU in the linear language were not making a linear generalization.

The findings in the tier-linear language suggest that all models make use of distributional lexical information more than hierarchical structure, a result also found in studies of RNN language models learning syntactic agreement (Bernardy & Lappin, 2017; Gulordava et al., 2018). Generalization accuracy and error analysis across all languages also lend support to another finding of McCoy et al. (2018), that GRUs are more prone to make hierarchical generalizations than RNNs and LSTMs. Relatively high generalization to the familiar set of all models in the hierarchical language indicates all models are capable of learning hierarchical generalizations, but the drop in performance on the novel set, particularly of the LSTM and RNN, suggest that the generalization is brittle.

As with children acquiring language, there is little evidence through error analysis that neural networks are learning linear rules, even when the data to which they are exposed is controlled so as to be expressible as a rule over linear position. This leaves open to future work the question of whether the hypothesis space considered by the networks excludes linear rules, as is expected in children, or whether there is something intrinsic to the task which makes linear generalization unlikely.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Jean-Phillipe Bernardy and Shalom Lappin. Using deep neural networks to learn syntactic agreement. *Linguistic Issues in Language Technology*, 15, 2017.

Esteban Buz and Robert Frank. Evaluating systematicity in neural networks through transformation combination. 2008.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

Noam Chomsky. Rules and representations. *Behavioral and brain sciences*, 3(1):1–15, 1980.

Stephen Crain and Mineharu Nakayama. Structure dependence in grammar formation. *Language*, pp. 522–543, 1987.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Robert Frank and Donald Mathis. Transformational networks. *Models of Human Language Acquisition*, pp. 22, 2007.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. Colorless green recurrent networks dream hierarchically. *arXiv preprint arXiv:1803.11138*, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

R. Thomas McCoy, Robert Frank, and Tal Linzen. Revisiting the poverty of the stimulus: hierarchical generalization without a hierarchical bias in recurrent neural networks. *CoRR*, abs/1802.09091, 2018. URL `http://arxiv.org/abs/1802.09091`.

Amy Perfors, Joshua B Tenenbaum, and Terry Regier. The learnability of abstract syntactic principles. *Cognition*, 118(3):306–338, 2011.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014. URL `http://arxiv.org/abs/1409.3215`.

Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.