Towards a Model of Prediction-based Syntactic Category Acquisition: First Steps with Word Embeddings

Robert Grimm Giovanni Cassani Walter Daelemans Steven Gillis
University of Antwerp, CLiPS
{name.surname}@uantwerpen.be

Abstract

We present a prototype model, based on a combination of count-based distributional semantics and prediction-based neural word embeddings, which learns about syntactic categories as a function of (1) writing contextual, phonological, and lexical-stress-related information to memory and (2) predicting upcoming context words based on memorized information. The system is a first step towards utilizing recently popular methods from Natural Language Processing for exploring the role of prediction in childrens' acquisition of syntactic categories.[1]

1 Introduction

Evidence is mounting that during language processing, the brain is predicting upcoming elements at different levels of granularity (Huettig, 2015). This could serve at least two purposes: (1) to facilitate understanding in dialogue and (2) to acquire abstract syntactic structure.

With respect to (1), Pickering and Garrod (2007) review evidence suggesting that people predict upcoming elements in their interlocutors' speech streams using the production system. This is thought to facilitate understanding in dialogue. One reason to postulate (2) is that length of memory span for syntactically well-formed sequences is positively correlated with an individual's ability to predict upcoming words (Conway, 2010; see Huettig, 2015, for further arguments).

We thus have evidence that people predict linguistic elements, and there is reason to suspect that this could be linked to the acquisition of syntactic structure. Models of prediction in language processing should therefore aim to demonstrate the emergence of such structure as a function of learning to predict upcoming elements.

Perhaps the most explicit account of such a process can be found in the work of Chang et al. (2006), who use a recurrent neural network in combination with an event semantics, in order to generate sentences with unseen bindings between words and semantic roles – i.e., the types of novel sentential constructions that could be afforded by abstract syntactic structure.

It is noteworthy, given this line of work, that prediction is central to recently popular methods from Natural Language Processing (NLP) for obtaining distributional representations of words (Mikolov et al., 2013; Pennington et al., 2014). Vector representations (often called *word embeddings*) obtained using these methods cluster closely in terms of semantic and syntactic types – an achievement due to engineering efforts, without emphasis on psychological constraints. Thus, if these methods are to be used for modelling aspects of human language processing, they should be modified to reflect such constraints.

Here, we attempt to take a first step into this direction: we modify the skipgram model from the *word2vec* family of models (Mikolov et al., 2013) – which predicts both the left and right context of a word – to predict only the righ context. Word counts from the left context form the basis for prediction and are tuned to maximize the likelihood of correctly predicting words from the right context. Throughout, we measure the organization of word embeddings in terms of syntactic categories – and find that embeddings of the same category cluster more closely after each training stage.

In addition to word frequencies from the left context, we experiment with phonological information and features related to lexical stress as the basis of predicting words from the right context.

[1]The work reported here was implemented in the Theano framework (Bastien et al., 2012; Bergstra et al., 2010). The code is freely available at: https://github.com/RobGrimm/prediction_based (commit ID: 6d60222)

2 Language Model

The model is trained in two consecutive stages: (1) for each word in the vocabulary, create a vector of frequency counts for words from its left context. Concatenate this with phonological and / or lexical stress features, and project the result into a joint dimensionality-reduced space. (2) Use the embeddings obtained in stage 1 to predict words from the right context, and modify the input embeddings via the backpropagation algorithm.

Stage 1 is meant to correspond to a memory component which tracks backward statistical regularities, while stage 2 is meant to correspond to a forward-looking predictive mechanism. On the NLP side, the model is a combination of count-based distributional semantics (stage 1) and prediction-based neural word embeddings (stage 2). While count-based and prediction-based approaches can produce similar results, provided the parameters are tweaked in a certain way, (Levy et al., 2015), it seems intuitively that adding counts is more like memorizing context, whereas an explicitly predictive component is more suited for modelling prediction in language processing.

To the best of our knowledge, there exists no other work which combines counting and predicting to derive word embeddings, nor work which attempts to relate this to language acquisition. The current model, being a result of preliminary explorations, is only loosely based on possible principles of cognitive processing; it may, nevertheless, have the potential to move currently successful methods from NLP closer to language acquisition research.

2.1 Memory Component: Auto Encoder

During the first stage, we use a denoising Auto Encoder to (a) reduce the dimensionality of the feature vectors and (b) project concatenated feature vectors (e.g. contextual and phonological) into a shared space. As a result, we see some first improvements of the vectors' clustering in terms of syntactic categories.

An Auto Encoder is a neural network that learns to transform a given input $x^{(i)}$ into an intermediate representation $h(x^{(i)}) = s(W \cdot x^{(i)} + b_v)$, so that a faithful reconstruction $y^{(i)} = s(W' \cdot h(x^{(i)}) + b_h)$ can be recovered from $h(x^{(i)})$ (Bengio, 2009), where s is a non-linear activation function. We set $s(z) = max(0, min(1, z))$ and $W' = W^T$, i.e. we use a truncated linear rectified activation function and work with tied weights.

The parameters of the network are the weight matrix W, the visible bias b_v and the hidden bias b_h. The Auto Encoder is trained via gradient descent to produce faithful reconstructions of a set of input vectors $\{x^{(1)}, ...x^{(n)}\}$ by minimizing the average, across training examples, of the reconstruction error $||x^{(i)} - y^{(i)}||^2$.

After training, the latent representation $h(x^{(i)})$ is often used for some other task. One strategy to force $h(x^{i)})$ to retain useful features is to train on a partially corrupted version $\widetilde{x}^{(i)}$ of $x^{(i)}$. This is the idea behind the denoising Auto Encoder (dAE), where part of the input vector $\widetilde{x}^{(i)}$ is set to 0.0 with probability v (the *corruption level*). The dAE is then trained to reconstruct the uncorrupted input.

2.2 Predictive Component: Softmax Model

In stage 2, the model learns to predict words from the embeddings obtained in stage 1. This is done by maximizing the probability $p(c|w;\theta)$ of context word c given target word w, for all pairs of target and context words $(w, c) \in D$. To obtain D, we first define an integer $t > 0$ as the context window. Considering each sentence S from the training corpus, for each target word $w_n \in S$ at position $n \leq length(S)$, we sample an integer t_n from the uniform distribution $\{1, ...t\}$. We then add the target-context word pairs $\{(w_n, w_{n+j}) : 0 < j \leq t_n, w_i \in S\}$ to D. Note that we only sample words from the right context, instead of from both left *and* right context. Aside from this difference, stage 2 is comparable to the *word2vec* skipgram model (Mikolov et al., 2013).

Here as there, the probability of a context word given its target word can be defined as:

$$p(c|w;\theta) = \frac{e^{v_c \cdot T_{w*}}}{\sum_{c' \in V} e^{v_{c'} \cdot T_{w*}}} \quad (1)$$

where the embedding T_{w*} of target word w is a row in the embeddings matrix T, v_c is a vector representation for context word c, and V is the vocabulary. (1) is computed by a neural network with a softmax output layer and weight matrix W, such that v_c is a row in W, and the parameters θ are W and T. The training objective is the minimization of the negative sum of log probabilities across all target word – context word pairs.

3 Data

3.1 Corpus and Vocabulary

Training data are based on a concatenation of 18 POS-tagged English corpora[2] from the CHILDES database (MacWhinney, 2000). We only consider utterances from the father and mother, i.e. utterances whose speaker was coded with either *FAT* or *MOT* (child-directed speech, or CDS). The concatenated North American and English corpora contain 1.555.311 and 1.575.548 words of CDS, respectively (3.130.859 words).

The vocabulary consists of the 2000 most frequent nouns, verbs, adjectives, and closed class words (words that are tagged as adverbs, communicators, conjunctions, determiners, infinitival *to*, numerals, particles, prepositions, pronouns, quantifiers, auxiliaries, wh-words, or modifiers), with homophones disambiguated by POS tags. In total, we end up with 1010 nouns, 522 closed class words, 302 verbs, and 166 adjectives.

3.2 Phonology and Lexical Stress

For each word from the vocabulary, we construct a phonological feature vector by first extracting its sequence of phonemes from the CELEX database (Baayen et al., 1995). Each phoneme is then placed on a trisyllabic consonant-vowel-grid, which is transformed into a 114-dimensional binary vector by concatenating the phonemes' vector-representations, as given by Li and MacWhinney (2002) (empty consonant-vowel slots are assigned a vector of zeros). Once done for every word, embeddings of similar-sounding words tend to be close to one another in the embeddings space. Finally, in order to learn more abstract phonological representations, the feature vectors are reduced to 30 dimensions, using a dAE trained for 200 epochs, with a learning rate of 0.1 and a corruption level of 0.1.

For each word, we also extract a lexical stress component from the CELEX database, which we transform into a binary vector of length three. Each index corresponds to one of three possible syllables, such that a one signifies the presence of primary stress and a zero indicates its absence.

[2] *UK corpora*: Belfast, Manchester. *US corpora*: Bates, Bliss, Bloom 1973, Bohannon, Brown, Demetras – Trevor, Demetras – Working, Feldman, Hall, Kuczaj, MacWhinney, New England, Suppes, Tardif, Valian, VanKleeck. See the CHILDES manuals for references: http://childes.psy.cmu.edu/manuals/

3.3 Training Set

Given the vocabulary V, we create the embeddings matrix T of size $|V| \times |V|$, where each row T_{w*} is a word embedding corresponding to a unique target word $w \in V$ and each column T_{*c} corresponds to a unique context word $c \in V$. A cell T_{wc} is then the frequency with which c occurs within a sentence-internal window of $t = 3$ words to the left of w, across all occurrences of w in CDS. Rows are normalized to unit interval.

The model is trained in three conditions, with the rows in T constituting the training set: (1) *context*: T remains unchanged; (2) *context + stress*: each row T_{w*} is concatenated with the lexical stress feature vector of w; (3) *context + phonology*: each row is concatenated with a phonological feature vector.

4 Training Procedure and Evaluation

While the task is to predict words, we are interested in a side effect of the learning process: the induction of representations whose organization in vector space reflects syntactic categories. To measure this, we train a 10-NN classifier on the embeddings after each training epoch, with embeddings labeled by syntactic category, and we stop training as soon as the micro F_1 score does not increase anymore.[3] To avoid premature termination of training due to fluctuations in F_1 scores during stage 1, we keep track of the epoch E at which we got the best score A. If scores stay smaller than or equal to A for 10 epochs, we terminate training and obtain the dimensionality-reduced embeddings for further training in stage 2 from the dAE's state at E. In stage 2, as there are no such fluctuations, it is safe to terminate as soon as there is no increase anymore. This procedure allows for as many training epochs as are necessary for achieving the best results – between 22 and 30 in the first and between 4 and 5 epochs in the second stage.

Performance is compared across stages as well as to a majority vote baseline (each data point is assigned the most common class) and a stratified sampling baseline (class labels are assigned in accordance with the class distribution). The expected pattern is that performance at each training stage is both above baseline and significantly bet-

[3] We track the micro instead of the macro F_1 measure because we think it is important for potential models of language acquisition to correctly categorize a majority of words, even at the expense of minority categories.

progress	category	context				context + stress				context + phonology			
		prec.	rec.	ma. F_1	mi. F_1	prec.	rec.	ma. F_1	mi. F_1	prec.	rec.	ma. F_1	mi. F_1
before stage 1	nouns	89	92	**0.746**	**0.819**	88	92	**0.751**	**0.821**	72	94	**0.566**	**0.734**
	verbs	66	92			70	90			67	77		
	adj.	68	48			68	48			58	4		
	clos. cl.	82	68			82	70			88	52		
after stage 1	nouns	91	93	**0.785**	**0.847**	89	92	**0.778**	**0.840**	81	94	**0.707**	**0.804**
	verbs	73	93			75	91			72	86		
	adj.	74	54			70	53			70	30		
	clos. cl.	84	73			83	73			89	66		
after stage 2	nouns	90	96	**0.810**	**0.865**	88	96	**0.799**	**0.858**	79	97	**0.725**	**0.812**
	verbs	81	87			81	86			82	83		
	adj.	76	60			75	54			75	32		
	clos. cl.	86	77			86	76			90	66		

Table 1: Precision and recall (in percent), together with micro and macro F_1 scores, based on a 10-NN classifier trained on the word embeddings at different stages during the training process.

ter than performance at the previous stage. Significance of differences is computed via approximate randomization testing (Noreen, 1989),[4] a statistical test suitable for comparing evaluation metrics such as F-scores (cf. Yeh, 2000).

Results are based on a dAE with 400 hidden units, trained with a learning rate of 0.01, and a corruption level of 0.1. The softmax model was trained with a learning rate of 0.008, with context words sampled from a sentence-internal window of $t = 3$ words to the right. Both models were optimized via true stochastic gradient descent.

5 Results and Discussion

Table 1 shows precision, recall and F_1 scores based on a 10-NN classifier trained on the word embeddings at three different points in time: (a) before training begins, with scores based on the input embeddings, (b) after stage 1, with embeddings projected into a lower-dimensional space, and (c) after stage 2, with embeddings modified as a result of predicting words from the right context.

F_1 scores at every stage are highly significantly different ($p \leq 0.001$) from both the majority vote baseline (macro F_1 = 0.168, micro F_1 = 0.505) and the stratified sampling baseline (macro F_1 = 0.247, micro F_1 = 0.354). Across conditions, F_1 scores after stage 1 are very significantly different ($p \leq 0.01$) from scores obtained before stage 1. Scores calculated after stage 2 are still significantly different ($p \leq 0.05$) from scores at the previous stage in the *context* and *context + stress* conditions, although there is no such significant difference in the *context + phonology* condition (but $p \approx 0.07$ for the difference between micro F_1 scores). There is no significant difference between the *context* and *context + stress* conditions at any stage, whereas the within-stage differences between F_1 scores in the *context* and *context + phonology* conditions are all highly significant.

We can make at least three observations. (1) The model performs as expected, in that there is a significant increase in performance after every stage – i.e., the induced word representations cluster more closely in terms of syntactic categories as training progresses. (2) The phonological component does not improve on the *context* condition, likely because phonological similarity often conflicts with syntactic similarity – most notably with homophones, but also with words such as the verb *tickle* and the noun *pickle*. (3) The lexical stress features do not seem to help, as there is no significant difference between the *context* and *context + stress* conditions.

6 Conclusions and Future Work

In general, the model demonstrates that it is possible to augment prediction-based word embeddings with a count based component, such that frequency counts serve as the basis of prediction and are further refined as a result of predicting right context. This can serve as a first step towards utilizing prediction-based methods in order to model childrens' acquisition of syntactic categories through a process of (1) tracking backward statistical regularities by writing to memory and

[4]We used an implementation by Vincent Van Asch, available at: http://www.clips.uantwerpen.be/scripts/art

(2) tracking forward regularities via prediction

Apart from that, the model can be used to compare the utility of different types of features. It makes explicit the distinction, identified by Huettig (2015), between *cue of prediction* (what is used as the basis of prediction) and *content of prediction* (what is predicted). Neither of the two possible *cues of prediction* we investigated turned out to be helpful for the induction of syntactic categories.

The initial experiments described in this paper emphasize different *cues of prediction*. In the future, we plan to also predict different kinds of features. Moreover, we plan to replace the psychologically implausible stage-like organization of the model with a more incremental architecture.

Acknowledgments

The present research was supported by a BOF/TOP grant (ID 29072) of the Research Council of the University of Antwerp.

References

Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. *Proceedings of the 18th conference on Computational Linguistics. Volume 2 (947–953).*

Brian MacWhinney. 2000. The CHILDES project: Tools for analyzing talk: Volume I: Transcription format and programs, volume II: The database. *Computational Linguistics*, 26(4):657–657.

Christopher M. Conway, Althea Baurnschmidt, Sean Huang, and David B. Pisoni. 2010. Implicit statistical learning in language processing: word predictability is the key. *Cognition*, 114(3):356–371.

Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses: An Introduction.* Wiley, Hoboken, New Jersey.

Falk Huettig (in press). 2015. Four central questions about prediction in language processing. *Brain Research*, doi:10.1016/j.brainres.2015.02.014.

Franklin Chang, Gary S. Dell, and Kathryn Bock. 2006. Becoming Syntactic. *Psychological Review*, 113(2):234–272.

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian Goodfellow, Arnaud Bergeron, Nicolas Bouchard, David Warde-Farley, Yoshua Bengio. 2012. Theano: new features and speed improvements. *NIPS 2012 deep learning workshop.*

Harald Baayen, Richard Piepenbrock, and Leon Gulikers. 1995. *The CELEX lexical database (CD-ROM release 2).* Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543,* Doha, Qatar, October. Association for Computational Linguistics.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, Yoshua Bengio 2010. Theano: A CPU and GPU Math Compiler in Python. *Proceedings of the Python for Scientific Computing Conference (SciPy) 2010.* Austin, Texas.

Martin J. Pickering and Simon Garrod. 2007. Do people use language production to make predictions during comprehension? *Trends in cognitive sciences*, 11(3):105–110.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3.

Ping Li and Brian MacWhinney. 2002. PatPho: A phonological pattern generator for neural networks. *Behavior Research Methods, Instruments, & Computers*, 34(3):408–415.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv:1301.3781 [Computation and Language (cs.CL)]*

Yoshua Bengio. 2009. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127.