Generating Sentences from Semantic Vector Space Representations

Mohit Iyyer[1], **Jordan Boyd-Graber**[2], **Hal Daumé III**[1]
[1]Computer Science & UMIACS
University of Maryland
College Park, MD
{miyyer,hal}@umiacs.umd.edu

[2]Computer Science
University of Colorado
Boulder, CO
jordan.boyd.graber@colorado.edu

1 Introduction

Distributed vector space models have recently shown success at capturing the semantic meanings of words [2, 15, 14], phrases and sentences [18, 16, 12], and even full documents [13, 3]. However, there has not been much work in the reverse direction: given a single vector that represents some meaning, can we generate grammatically correct text that retains that meaning?

The first work of this kind in a monolingual setting[1] successfully generates two and three-word phrases with predetermined syntactic structures by decoupling the task into three phases: *synthesis*, *decomposition*, and *search* [4]. During the synthesis phase, a vector is constructed from some input text. This vector is decomposed into multiple output vectors that are then matched to words in the vocabulary using a nearest-neighbor search.

We depart from this formulation by learning a joint synthesis-decomposition function that is capable of generating grammatical sentences with arbitrary syntactic structures. Our model is an unfolding and untied recursive autoencoder (RAE) with connections between sibling nodes. We show promising qualitative results and conclude with future directions.

2 Unfolding Recursive Autoencoders

The unfolding recursive autoencoder was first introduced in Socher et al. [20] for a paraphrase detection task. We structure our network around dependency parse trees because dependency-tree recursive neural networks have been shown to be more invariant to syntactic transformations than their constituency-tree counterparts [19, 10]. As we will show later, dependency trees are also ideal for generation because the most meaningful words in a sentence (e.g., verb, subject, object) are close to the root node.

2.1 Model Structure

We start by associating each word w in our vocabulary with a vector representation[2] $x_w \in \mathbb{R}^d$. These vectors are stored as the columns of a $d \times V$ dimensional word embedding matrix L, where V is the vocabulary size.

The input to our model is a collection of dependency parse trees where each node n in the parse tree for a particular sentence is associated with a word w, a word vector x_w, and a hidden vector $h_n \in \mathbb{R}^d$ of the same dimension as the word vectors. Unlike in constituency

[1]Recently proposed MT models for rescoring candidate translations [11, 21, 1] can conceivably also be used to generate language.

[2]We use GloVe [17] to initialize these vectors.

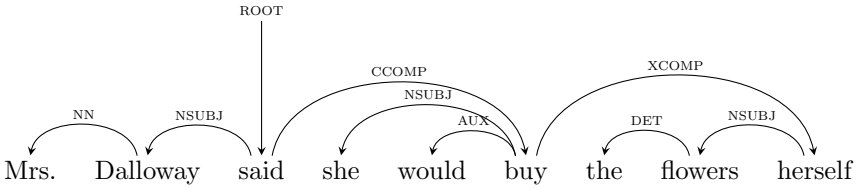

Figure 1: Dependency parse tree of the opening sentence of Virginia Woolf's *Mrs. Dalloway*.

trees where all words reside at the leaf level, internal nodes of dependency trees are associated with words. Thus, the DT-RAE has to combine the current node's word vector with its children's hidden vectors to form h_n. This process continues recursively up to the root, whose hidden vector h_{root} represents the entire sentence. During the decomposition phase, we unfold the tree from the root, which gives us a reconstructed version of the sentence. Our training objective minimizes the error between the original and the reconstructed sentence.

2.2 From Sentence to Vector: the Synthesis Phase

We associate a separate $d \times d$ matrix W_r with each dependency relation r in our dataset and learn these matrices during training. Syntactically untying these matrices allows the model to take advantage of relation identity as well as tree structure. We include an additional $d \times d$ matrix, W_v, to incorporate the word vector x_w at a node n into the hidden vector h_n.

Given a parse tree, we first compute all leaf representations. For example, the hidden representation $h_{\text{mrs.}}$ for the parse tree given in Figure 1 is

$$h_{\text{mrs.}} = f(W_v \cdot x_{\text{mrs.}} + b_1), \tag{1}$$

where f is a non-linear activation function such as tanh and b_1 is a bias term. After finishing with the leaves, we move to interior nodes whose children have already been processed. Continuing from *mrs.* to its parent, *dalloway*, we compute

$$h_{\text{dalloway}} = f(W_{\text{NN}} \cdot h_{\text{mrs.}} + W_v \cdot x_{\text{dalloway}} + b_1). \tag{2}$$

We repeat this process up to the root, which is

$$h_{\text{said}} = f(W_{\text{NSUBJ}} \cdot h_{\text{dalloway}} + W_{\text{CCOMP}} \cdot h_{\text{buy}} + W_v \cdot x_{\text{said}} + b_1). \tag{3}$$

The composition equation for any node n with children $K(n)$ and word vector x_w is $h_n =$

$$f(W_v \cdot x_w + b_1 + \sum_{k \in K(n)} W_{R(n,k)} \cdot h_k), \tag{4}$$

where $R(n, k)$ is the dependency relation between node n and child node k.

2.3 From Vector to Sentence: the Decomposition Phase

In the traditional RAE, the error for each node in the network is computed by reconstructing the hidden layers of its immediate children and then taking the Euclidean distance between the original and reconstruction. The objective function of the unfolding RAE is more powerful: for every node in a tree, we first *unfold* that node's hidden layer down to the leaf level. In our model, we only unfold the root node (instead of unfolding all internal nodes) to improve training speed. Given the root representation h_{root} of a sentence, we compute a sequence of word embeddings that are then compared to the original sequence through Euclidean distance.

To be more specific, we associate a $d \times d$ decomposition matrix D_r with each dependency relation r in our dataset. Going back to our example, we unfold from the root representation to compute reconstructions u_n for every node n in the tree:

$$u_{\text{dalloway}} = f(D_{\text{NSUBJ}} \cdot h_{\text{said}} + b_2), \quad u_{\text{mrs.}} = f(D_{\text{NN}} \cdot u_{\text{dalloway}} + b_2). \tag{5}$$

Finally, given a reconstructed hidden vector, we apply D_v, the decomposition analogue of W_v, to extract the reconstructed word embedding x'_n:

$$x'_{\text{mrs.}} = f(D_v \cdot u_{\text{mrs.}} + b_2). \tag{6}$$

The error J for a dataset of sentences $s \in S$ where each parse tree contains nodes N_s is

$$J = \sum_{s \in S} \frac{1}{|N_s|} \sum_{n \in N_s} \left\| x_n - x'_n \right\|^2, \tag{7}$$

The described model is already capable of producing decent reconstructions of full sentences. However, it suffers from a serious problem: if there exist two sibling nodes c_1 and c_2 that share the same dependency relation r to their parent, then their unfolded representations u_1 and u_2 will be identical. For example, take the phrase *sleepy brown cat*, where *sleepy* and *brown* are both adjective modifiers of *cat*. Then,

$$u_{\text{sleepy}} = u_{\text{brown}} = f(D_{\text{AMOD}} \cdot u_{\text{cat}} + b_2). \tag{8}$$

How do we solve this problem? One simple solution is to untie our composition and decomposition matrices by position as well as dependency relation. This means that in our *sleepy brown cat* example, *sleepy* is related to *cat* through the AMOD_1 relation, while *brown* is connected by the AMOD_2 relation. While simple, this approach runs into data sparsity issues for less common relations and thus requires much more training data to learn good parameters.

We instead alter the structure of our decomposition model by introducing another $d \times d$ matrix, W_{sib}, that conditions the reconstructed hidden layer u of a child node on its nearest left sibling as well as the parent node[3]. For our *sleepy brown cat* example, we have

$$u_{\text{sleepy}} = f(D_{\text{AMOD}} \cdot u_{\text{cat}} + b_2), \quad u_{\text{brown}} = f(W_{sib} \cdot u_{\text{sleepy}} + D_{\text{AMOD}} \cdot u_{\text{cat}} + b_2). \tag{9}$$

This modification allows information to flow left-to-right as well as top-to-bottom (see Figure 2) and fixes the issues of identical sibling reconstructions. There are many possible ways to improve this model's representational capacity: we could untie sibling connections based on parts-of-speech, for example, and add some weighting parameter α that controls how much siblings influence reconstructions. In our current model, every node that has an identically-related sibling to the left is connected to that sibling by W_{sib}.

The model parameters $(W_{r \in R}, D_{r \in R}, W_v, D_v, W_{sib}, L, b_1, b_2)$ are optimized using AdaGrad [5], and the gradient of the objective function is computed using backpropagation through structure [9].

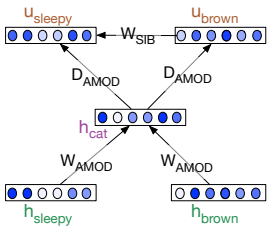

Figure 2: Example DT-RAE with sibling relationship

2.4 Generating Sentences

How do we use this model to generate sentences? Given a sentence, we feed it through the synthesis phase, leaving us with a sentence-level representation at the root node. During decomposition, we pass this vector back through the original tree, which yields a reconstructed vector at every node. By searching for the closest word vector in L to each of these reconstructed vectors, where "closest" is defined in terms of Euclidean distance, we can recreate the original sentence.

Reconstructing a given input sentence is not particularly interesting or useful, although this is the task optimized by our training objective. If we instead allow our output to be of arbitrary syntactic structure, our task becomes paraphrase generation, which is much less trivial. We move in this direction by decomposing the sentence-level representation computed in the synthesis phase through a tree that is randomly chosen from the training data.

[3]The probabilistic version of this technique has been used to improve dependency parsing accuracy [6, 7].

O	name this 1922 novel about leopold bloom written by james_joyce
R	name this 1906 novel about gottlieb flecknoe inspired by james_joyce
P	what is this william_golding novel by its written writer
O	ralph_waldo_emerson dismissed this poet as the jingle man and james_russell_lowell called him three-fifths genius and two-fifths sheer fudge
R	henry_david_thoreau rejected this author like the tsar boat and imbalance created known good writing and his own death
P	henry_david_thoreau rejected him through their stories to go money well inspired stories to write as her writing
O	this is the basis of a comedy of manners first performed in 1892
R	another is the subject of this trilogy of romance most performed in 1874
P	subject of drama from him about romance
O	in a third novel a sailor abandons the patna and meets marlow who in another novel meets kurtz in the congo
R	during the short book the lady seduces the family and meets cousin he in a novel dies sister from the mr.
P	during book of its author young lady seduces the family to marry old suicide while i marries himself in marriage
O	thus she leaves her husband and child for aleksei vronsky but all ends sadly when she leaps in front of a train
R	however she leaves her sister and daughter from former fianc and she ends unfortunately when narrator drives into life of a house
P	leaves the sister of man in this novel

Table 1: Five examples of original sentences from our dataset (**O**), reconstructed versions of those sentences with the same tree structure as the original (**R**), and finally generated paraphrases with different tree structure (**P**).

3 Experiments

Table 1 shows examples of both reconstructions as well as generated paraphrases. We train the model using 100,000 sentences from a combination of Wikipedia and the trivia question dataset of Iyyer et al. [10]. We chose this dataset because it has a very rich vocabulary (31,504 words) that includes numerous named entities, dates, and numbers, and we were curious to see how the model would handle rare words. The output trees during paraphrase generation are constrained such that the number of words in the output must be less than or equal to the number of words in the input, and we set d to 100 for training.

4 Discussion & Future Work

The qualitative results show that while our model is able to reconstruct sentences fairly well, named entities, numbers, and dates are rarely reconstructed correctly (e.g., *1922* becomes *1906*). One potential solution is to modify generated sentences to include such words, which is consistent with the interpreting note-taking method used by simultaneous translators to make sure they do not omit important details during translation [8].

Moving on to generated paraphrases, we see a clear difference in grammaticality as well as meaning retention compared to reconstructions. However, the model has promise: parts-of-speech are reasonably ordered, and at least some of the original meanings are retained. The dependency-tree representation gives us a great starting point since the input verb is associated with the root of the input tree and thus also with the root of the output tree. As we get farther and farther from the root, though, the generated words become more nonsensical (e.g., *marry old suicide*).

We are currently working to improve the quality of generated paraphrases by increasing model complexity specifically within the sibling connections. One especially interesting future direction is to move beyond paraphrases by forcing the model to formulate a response to the input rather than simply copying its meaning.

Acknowledgments

This work was supported by NSF Grant IIS-1320538. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsor.

References

[1] Cho, K., van Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of Emperical Methods in Natural Language Processing*.

[2] Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the International Conference of Machine Learning*.

[3] Denil, M., Demiraj, A., Kalchbrenner, N., Blunsom, P., and de Freitas, N. (2014). Modelling, visualising and summarising documents with a single convolutional neural network. *arXiv preprint arXiv:1406.3830*.

[4] Dinu, G. and Baroni, M. (2014). How to make words with vectors: Phrase generation in distributional semantics. In *Proceedings of the Association for Computational Linguistics*.

[5] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 999999:2121–2159.

[6] Eisner, J. (1996). Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of International Conference on Computational Linguistics*.

[7] Finkel, J. R., Grenager, T., and Manning, C. D. (2007). The infinite tree. In *Proceedings of the Association for Computational Linguistics*.

[8] Gillies, A. (2005). *Note-taking for Consecutive Interpreting: A Short Course*. St. Jerome Pub.

[9] Goller, C. and Kuchler, A. (1996). Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1.

[10] Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., and Daumé III, H. (2014). A neural network for factoid question answering over paragraphs. In *Proceedings of Emperical Methods in Natural Language Processing*.

[11] Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *Proceedings of Emperical Methods in Natural Language Processing*.

[12] Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of Emperical Methods in Natural Language Processing*.

[13] Le, Q. V. and Mikolov, T. (2014). Distributed representations of sentences and documents.

[14] Levy, O. and Goldberg, Y. (2014). Dependency-based word embeddings.

[15] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[16] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Proceedings of Advances in Neural Information Processing Systems*.

[17] Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of Emperical Methods in Natural Language Processing*.

[18] Socher, R., Huang, E. H., Pennington, J., Ng, A. Y., and Manning, C. D. (2011a). Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *Proceedings of Advances in Neural Information Processing Systems*.

[19] Socher, R., Le, Q. V., Manning, C. D., and Ng, A. Y. (2014). Grounded compositional semantics for finding and describing images with sentences. *TACL*.

[20] Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., and Manning, C. D. (2011b). Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proceedings of Emperical Methods in Natural Language Processing*.

[21] Zhang, J., Liu, S., Li, M., Zhou, M., and Zong, C. (2014). Mind the gap: Machine translation by minimizing the semantic gap in embedding space. In *Association for the Advancement of Artificial Intelligence*.