ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

Visualizing Stacked Autoencoder Language Learning

Trevor Barron and Matthew Whitehead *

Colorado College - Department of Mathematics and Computer Science
14 E. Cache La Poudre St., Colorado Springs, CO 80903 - USA

Abstract. Visualizing the features of unsupervised deep networks is an important part of understanding what a network has learned. In this paper, we present a method for visualizing a deep autoencoder's hidden layers when trained on natural language data. Our method provides researchers insight into the semantic language features the network has extracted from the dataset. It can also show a big picture view of what a network has learned and how the various features the network has extracted relate to one another in semantic hierarchies. We hope that these visualizations will aid human understanding of deep networks and can help guide future experiments.

1 Introduction

In this paper, we present a visualization procedure for deep machine learning networks of stacked autoencoders (a neural network consisting of multiple layers of sparse autoencoders in which the outputs of each layer are connected to the inputs of the successive layer) trained on natural language data. After processing large amounts of text data, the networks learn different language features with representations stored within their hidden layers. We take these trained networks and then generate images that help show each of the learned language features. Our method is analogous to the methods used in computer vision research, but since we are working with text data, we use a different visualization technique that incorporates word cloud images and t-SNE plots. We hope that our visualization technique will help NLP researchers analyze and understand their networks more easily. Our software is publicly available: `https://github.com/tpbarron/CaffeAutoencoderViz`

Research in computer vision with deep networks often includes images showing the visual features particular hidden nodes have learned to recognize. These images are generated by calculating the pixel values that maximally activate a particular hidden node. For example, [1] shows maximally activated input images for several hidden layers of a deep network trained on images of human faces. The detected features on different levels of the network correspond with different levels of visual features. Low-level features include edges and simple curves, while higher-level features are more specific to human faces and are comprised of combinations of the lower-level features.

*We gratefully acknowledge NVIDIA Corporation for the donation of the Titan X GPU used to support this research.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

Techniques from machine learning have been used extensively for language modeling, including the generation of word embeddings in semantic feature spaces. Bengio et al. showed early work in using neural models for learning distributed word embeddings [2]. Mikolov et al. showed an effective method for generating word embeddings and also discussed how embeddings can be combined in semantically and syntactically interesting ways [3].

2 Visualizing Language Learning with Word Clouds

The goal of our work is to visualize what stacked autoencoders have learned when processing natural language data. In particular, we are interested in analyzing networks that are learning word embedding features that provide a distributed, vector representation of coarse word semantics. These word embeddings are learned by training a stacked autoencoder on a large corpus of text documents with individual documents modeled using bag-of-words representations. After training, word embeddings can be obtained by capturing hidden node activations when the network is given a sparse vector representing a single word as input. Since there are multiple hidden layers in typical stacked autoencoder networks, word embeddings values from different hidden layers can be obtained that yield different granularity levels of coarse word semantics.

Once the network is trained, then it can be examined to determine the input vector that produces the highest output signal for a particular hidden node. A large activation signifies an input feature that has been identified by the node as significant. The maximum activation input vector can be produced for each hidden node in the network by choosing input values that satisfy a certain constraint, typically a bound on the overall magnitude of the input vector. Erhan et al. [4] describe one method for solving this optimization problem by determining the maximum activation for any hidden node through the use of gradient ascent.

Once maximum activation vectors have been obtained for all the hidden nodes in a network, we propose visualizing the detected features using word clouds. A *word cloud* is widely-used language visualization technique where words from a given text are arranged in an image such that a word's size indicates its relative importance to other words: larger words have more importance and the smaller words have less importance. We use word clouds to visualize each hidden node's set of maximum activation words. Hidden node word clouds consist of words from the vocabulary scaled in size relative to the magnitude of their contribution to the node's activation. Words corresponding with high values in the maximum activation vector are in a larger font in the word cloud than words with lower values. The relative size of two words is not significant across two word clouds but only within a single word cloud.

3 Experiments

We ran two experiments using implementations written in Torch [5] and Caffe [6] to show how our visualizations are helpful for seeing how networks have learned

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

from text data. The first had a very small vocabulary making it easy to see clearly what each node has conceptually come to represent. The second used a much larger vocabulary and dataset and only a small part of the visualization is viewed at one time.

3.1 Four Semantic Categories

We first ran a small test with just 41 words hand-picked to fit into one of four semantic categories: days of the week, months of the year, NBA basketball teams, and NFL football teams. We used one GB of data from Wikipedia [7] for our dataset of documents. We built a small stacked autoencoder network with four nodes (one for each category) in its first hidden layer and two in its second hidden layer to force the network to find an optimal compression of the four categories down to two. We trained the first hidden layer for 20,000 iterations and the second hidden layer for 15,000 iterations. We then used a world cloud [8] library to generate visualizations for each of the network's hidden nodes.

Fig. 1: Word clouds for all the hidden nodes in the day/month/basketball/football network after training completes.

Figure 1 shows all the hidden node word clouds for the network. The lower word clouds show that the four word groups are completely separated by each of the four nodes in the first hidden layer. The higher word clouds show that the network learned to effectively group both sports categories in one node and both time-related categories in the second hidden layer.

3.2 Large Wordlist Experiment

Our second experiment used a wordlist of size 10,000 gathered from a complete copy of Wikipedia. The words were the 10,000 most common words after removing stopwords. We implemented a stacked autoencoder network with 200 hidden nodes in the first hidden layer and 50 hidden nodes in the second hidden layer.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

We randomly sampled 45,574 Wikipedia articles and then trained the network for 500,000 iterations.

Since the network is so large, visualizing it all at one time is unhelpful. Instead, we visualize parts of the network individually by selecting a particular node from the second hidden layer and then producing a smaller combined word cloud along with the nodes from the first hidden layer that maximally contribute to the second layer node.

Fig. 2: A combined word cloud showing a single second layer hidden node word cloud on top and five of its maximally contributing first layer hidden node word clouds beneath.

Fig. 3: A second example of a combined word cloud constructed using another second hidden layer node along with its five maximally contributing first layer nodes.

In Figure 2, the top word cloud is from the second hidden layer so it detects higher-level features that combine several of the lower-level features. For example, the left-most first layer node has words that relate to directions (*north,*

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

south) and places (*united states, british, world*). We can see that *world, south, north,* and *west* are also prominent in the second layer node. The right-most first layer node contains many number words and we see that the word *three* is in the second layer node as well. With such a large wordlist and dataset, the semantic separations are less distinct than in our small experiment. Figure 3 shows another example word cloud based on a different second layer hidden node. We can see similar relationships between the first layer node themes (*family, film, national, etc.*) and the second layer node.

Using a t-SNE visualization that maps word activations to a lower-dimensional space for easier viewing, we can see the same kind of learned semantic hierarchies that the word cloud images show.

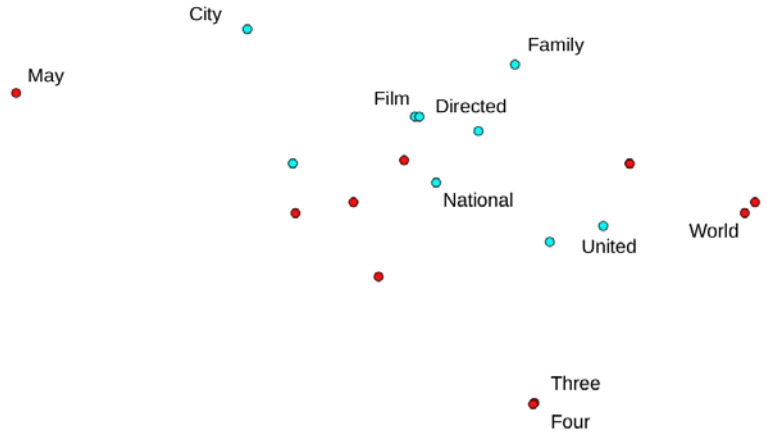

Fig. 4: t-SNE visualization of word first layer hidden node activations colored according to which second layer hidden node they are most closely aligned.

Figure 4 shows a t-SNE embedding of first hidden layer activations for words from Figure 2 (darker red points) and Figure 3 (lighter teal points). We can see that the two groups are relatively well separated which corresponds with the two second layer hidden nodes finding mostly distinct higher-level features to detect.

Figure 5 shows a t-SNE embedding of a single second layer hidden node. The color of each point represents the groupings of the first hidden layer nodes within the second layer node. One can see that even within a second layer node words are grouped conceptually according to their input node. For example, in Figure 5 we see that this node has learned both temporal, directional, and some sort of locational semantics but even these are split according to semantic closeness.

4 Conclusion

In this paper we have presented a technique for visualizing what stacked autoencoders have learned when trained on natural language data. These visualizations

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

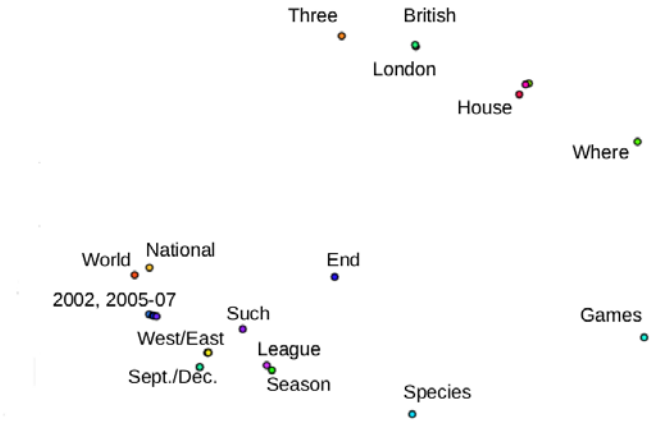

Fig. 5: t-SNE showing a single second layer hidden node where each color signifies input from a unique first layer hidden node.

help show how a network has learned to represent word meanings internally and these insights could guide future experiments, including determining effective network structures and finding appropriate training stopping points.

References

[1] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 609–616, New York, NY, USA, 2009. ACM.

[2] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. In Todd K. Leen, Thomas G. Dietterich, and Volke Tresp, editors, *NIPS*, pages 932–938. MIT Press, 2000.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen 0010, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013. `http://arxiv.org/abs/1310.4546`.

[4] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Visualizing higher layer features of a deep network. *Technical Report 1341, DIRO, Universite de Montreal*, 2009.

[5] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning, 2011. `http://torch.ch`.

[6] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[7] Matt Mahoney. Large text compression benchmark, 2011. `http://mattmahoney.net/dc/textdata.html`.

[8] Andreas Mueller. Python wordcloud library, 2015. `https://github.com/amueller/word_cloud`.