SIGML - Document Embeddings using DBNs

...

Nishit Asnani, 14433

This Project had been done under IIT Kanpur New York Office during Summer 2016.

Why Document Embeddings?

Why Document Embeddings?

- Low dimensional representation of documents that (hopefully) captures their semantics.
- Useful for clustering on the basis of their topics / contents - document classification
- Useful for information retrieval - a similarity measure can help retrieve a set of documents that is similar in content as the query document.

Datasets Used for Training / Testing

- IMDb movie review dataset - 50000 examples of labelled reviews - half positive and half negative. Using document embeddings for Sentiment Classification.
- 20 Newsgroups (Reuters) - about 19000 documents coming from 20 different classes, ranging from atheism to computer graphics to politics.

Popular Approaches

- Naive Bayes Classifier - naive, but could be useful for simple datasets.
- Latent Dirichlet Allocation - probably the most popular approach. Considers a document to be a mixture of topics and a topic to be a distribution over words.
- Other approaches - Latent Semantic Analysis and similar methods.

Towards another method

- Need to get better results.
- Intuition - Deep models might use a better insight into the documents.
- I felt that Generative Models can generally express complex relationships among the observed and target variables.
- Firstly, I tried Paragraph Vectors (Mikolov et. al.) - not a generative model, uses an approach similar to continuous bag of words for word embeddings.
- Later, for shorter run times and better classification, Deep Belief Networks were tried.

Paragraph Vectors

Paragraph Vector Model

- Randomly initialize fixed length vectors for each word in our vocabulary and for each document in the dataset.
- For a document d, word w, concatenate the vectors corresponding to the words just before and just after w, up to a specified context size.
- Concatenate the document vector to the above vector to get a vector the size of twice the context size + 1 times the size of each word vector.
- Perform a training step on the weights and biases connecting this input vector to the vector of the word w. In this way, all the word vectors and document vectors can be learned.

Paragraph Vector Model - Limitations

- Training takes a long while to come down to a decent error rate.
- Accuracy on the datasets tested on is not high.

Deep Belief Network

Restricted Boltzmann Machine (RBM)

- Energy based model - $E(v, h)$ is defined as the energy of the model, v is the set of visible units and h is the set of hidden units.
- Partition Function, Z
- a and b are the visible and hidden biases respectively, and w is the weight matrix relating the visible and hidden units
- $E(v, h) = - \sum a(i)v(i) - \sum b(j)h(j) - \sum v(i)h(j)w(i, j)$
- $p(v) = 1/Z^* \sum_h \exp(-E(v, h))$
- $\partial \log p(v) / \partial w(i, j) = Ex(v(i)h(j))_{data} - Ex(v(i)h(j))_{model}$

RBM

- Expectation over the model can’t be computed exactly, so we rely on Contrastive Divergence learning.
- Hinton et. al. showed that even a single step of Gibbs sampling can lead to a decent learning algorithm.
- Weights are updated as: $\Delta w(i, j) = lr * (Ex(v(i)h(j)_{data} - Ex(v(i)h(j)_{recon})$

The DBN

- A stack of RBMs are used to pre-train a DBN. For an input vector, output from one RBM is used as input for the next one.
- The stack is unfolded and formed into an autoencoder-decoder structure. Finetuning is done by approximating the input vector by the output vector, and the bottleneck layer is used as the embedding.

Extending to an Online Implementation

- A mini batch of data is stored and processed.
- Learning rate is linearly / exponentially reduced, and mini batch size is decided using cross validation.
- Testing is done by assigning all documents in the test set to the label of their nearest neighbor majority class.

Observations

- The DBN, even in the online setting, performed better than the paragraph vector model both in terms of accuracy and running time.
- The offline DBN is an order of magnitude faster than the paragraph vector model.
- This is despite the fact that the input vectors to the DBN are simply binary vectors corresponding to the occurrence of particular words in the document, whereas the paragraph vector takes into account the semantic relationship among words as well as their sequence.
- A replicated softmax model was also tried, which tries to make sense of not just the occurrence, but the frequency of particular words in a document. Unfortunately, this method failed to reach a global optimum, and thus didn’t perform very well.

Thank You