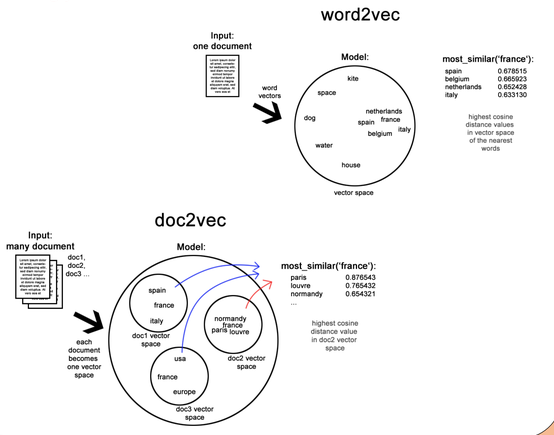
An Empirical Approach to Sentiment Analysis with Doc2Vec

Hieu Pham and Daniel Boley
University of Minnesota – Computer Science and Engineering

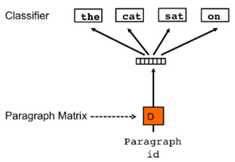
Introduction

The project aims to apply and evaluate Doc2Vec, an improvement of the so-called breakthrough Word2Vec in Natural Language Processing (NLP) tasks. Word2Vec and Doc2Vec are invented by Google Brain in recent works as a way to efficiently represent any word, sentence, or paragraph as a vector in space. In this research project, we focus on Sentiment Analysis task as a way to evaluate Doc2Vec, simply predicting user's rating from restaurant reviews in Yelp dataset. The practical applications of this research is that reviews play a critical role in recommender systems, business analytics, customer satisfaction and so forth. Through this project, we are able to see the complex math and the novelty behind seemingly-trivial NLP tasks.



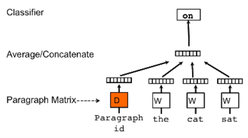
Doc2Vec

Distributed Bag Of Words (DBOW)



DBOW considers the concatenation of paragraph vectors and word vectors without word ordering to predict the next word in a text window. At each iteration of stochastic gradient descent, a text window is sampled, then sample a random word from text window for classification task.

Distributed Memory (DM) model



Paragraph token can be thought as another word, acting as a memory that remembers what is missing or topic of the paragraph. The algorithm has two main stages: training stage to get vectors and softmax weights, and inference stage for unseen data by gradient descending newly added columns on paragraph matrix.

Memory usage = # of vectors * Dimension * 4 bytes

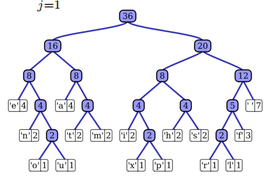
Word2Vec (skip-gram)

Hierarchy Softmax + Huffman Coding

Hierarchy softmax is a computationally efficient approximation of the full softmax. Instead of evaluating W output nodes in neural network, only $\log_2(W)$ nodes are evaluated to obtain probability distribution.

Binary Huffman tree's construction applies the greedy choice of merging two least frequent nodes that makes accessibility of frequent words more efficient and therefore is practical for fast training.

$$p(w = w_O) = \prod_{j=1}^{L(w)-1} \sigma\left([\![n(w,j+1) = \mathrm{ch}(n(w,j))]\!] \cdot {\mathbf{v}'_{n(w,j)}}^{T} \mathbf{h}\right)$$



Huffman Tree - illustration from Wikipedia

Negative sampling

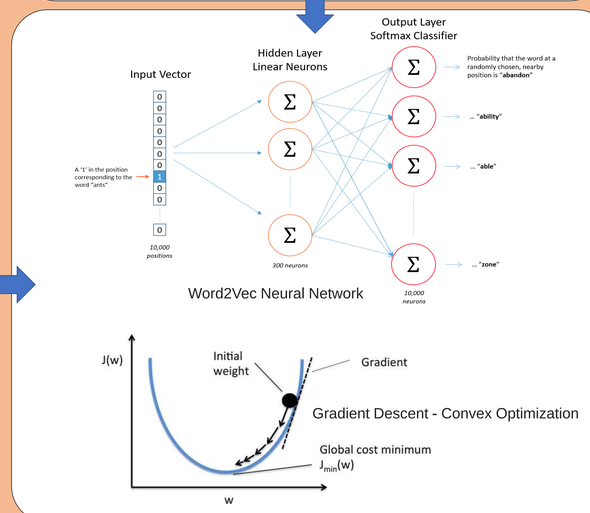
The idea of Word2Vec is to maximize similarity (dot product) of vectors among words appear together in context and minimize for the opposite. Negative sampling technique is a speedup process by randomly choosing contexts instead of considering all of the contexts, making use of a logistic loss function to minimize negative log-likelihood of words in training set. The transformation at the end is a sigmoid function to serve the previously-mentioned purpose for neural network.

$$\log \sigma({v'_{w_O}}^{\top} v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-{v'_{w_i}}^{\top} v_{w_I})\right]$$

Subsampling

To counter the imbalance between the rare and the frequent words, subsampling technique resolves the situation by discarding words with probability:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$



Empirical Results

Google Research

The original papers experienced Word2Vec/Doc2Vec in a handful of NLP tasks, includes Analogical Reasoning, Sentiment Analysis, Text Classification, etc. Word2Vec/Doc2Vec is unsupervised with raw data and a deep learning model. Thus, intuitively, the more data is fed, the better the outcome. This is confirmed in the papers.

In Word2Vec, preprocessing is required before actual training. Semantic meanings are expressive and can be computed with simple mathematics formula. Negative Sampling outperforms Hierarchy Softmax + Huffman Coding. With subsampling, the result is even more positive towards the conclusion.

In Doc2Vec, DM alone usually works well for most tasks (state-of-art performances). Interestingly, the combination of both DM and DBOW is generally more consistent and strongly recommended. Google Research team somehow achieved an impressive result from IMDB movie review dataset (~97%).

IBM Research

Several word-embedding and NLP models are compared against Word2Vec/Doc2Vec, especially a recent skip-thought model. Predictably, Doc2Vec is superior over several datasets: WIKI, AP-NEWS, Google News, STS, StackExchange, etc.

DBOW favors longer windows for context words than DM. Sub-sampling is essential for high performance. DM requires more epochs to reach convergence than DBOW.

Empirical results from IBM paper suggest that DBOW is a better model than DM. The absence of updating embedding words (optional in Gensim) before running DBOW degrades severely the performance of the model.

Unfortunately, the qualitative difference between Word2Vec and Doc2Vec remains unclear up to date.

Yelp dataset

The research project applies Doc2Vec to predict user ratings from Yelp dataset. A rating of 4-5 is considered positive and 1-3 is negative. A wide range of data sizes, options, and techniques are evaluated. Models are tested with holdout-validation and cross-validation methods in data mining. Unlike Word2Vec, Doc2Vec's preprocessing is simple and straightforward. A document is converted to lower cases and trimmed away from punctuations and rare words. Finally, all vectors are graphed against Logistic Regression, Naïve Bayes, and Support Vector Machine.

Doc2Vec, in average, performs well with high accuracy (> 80%). Increment of epochs also helps enhance models' performance. The results are consistent with validation methods. Unexpectedly, a strange behavior occurred to one model of $2x10^5$ documents gives an average result (~50%) when applying the learning rate.

In Word2Vec, to predict unseen paragraph, average vector technique or n-grams are usually applied. In Doc2Vec, the framework associating with DM and DBOW models is highly flexible for prediction of unseen paragraphs.

Training a large corpus with Doc2Vec consumes a significant amount of memories, leading to the thrashing problem. In large corporation, this is not an issue with several data centers and high-performing computing power. In academic settings, this requires several computers and efficient GNU distributions.

Conclusions

The Sentiment Analysis task from this research project serves no other purpose than familiarizing with Word2Vec/Doc2Vec (and related papers), machine learning, data mining, and NLP techniques. A more complex application is emotion classification that is multi-labels and more difficult to observe relationship between semantics and emotional states.

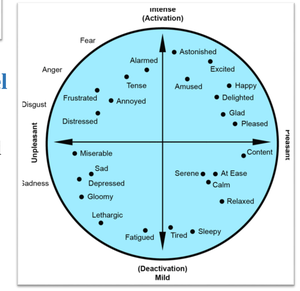
Sentiment Analysis → Emotion Classification

Discrete Emotion Model

- 6 basic emotions/multi-labels
- Proven in Psychology but debatable.
- What about emotionless?
- Experiment with Twitter: github.com/sivu1/word2vec_study/wiki

Circumplex Emotion Model

- Space representation
- Feature extraction task
- Supervised, Semi-supervised or Unsupervised?
- Correlation with Word2Vec and Doc2Vec?
- Nodes in Neural Network?
- What about other models? Lack of reliable datasets?



Acknowledgement

Special thanks to professor Daniel Boley for his technical advice, professor Maria Gini for her support in personal development, and other individuals for their times, knowledge and discussions.

Gensim developed by RaRe Technology in Python is an awesome open-source library for Word2Vec/Doc2Vec applications. The software is well documented and supplied with tutorials.

References

[1] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. 2013.
[2] Le, Quoc V., and Tomas Mikolov. "Distributed Representations of Sentences and Documents." ICML. Vol. 14. 2014.
[3] Dai, Andrew M., Christopher Olah, and Quoc V. Le. "Document embedding with paragraph vectors." arXiv preprint arXiv:1507.07998 (2015).
[4] Lau, Jey Han, and Timothy Baldwin. "An empirical evaluation of doc2vec with practical insights into document embedding generation." arXiv preprint arXiv:1607.05368 (2016).
[5] Canales, Lea, and Patricio Martínez-Barco. "Emotion Detection from text: A Survey." Processing in the 5th Information Systems Research Working Days (JISIC 2014) (2014): 37.