Word2vec has revolutionized word embedding in Natural Language Processing

Summary: Word2Vec, a statistical method for efficiently learning a standalone word embedding from a text corpus, was developed at Google [1] in 2013 to make the neural-network-based training of the embedding more efficient.

Texts are an important piece of information which is readily available on the internet. Using it as training data set for computers, we can build state of the art applications such as spell correction, language translator, spam filtering, recommendation systems, virtual assistants. All such applications are based on Natural Language Processing (NLP) models.

Computers can't handle texts but can easily handle mathematics. Word embedding is a collection of models that maps words or phrases to vectors in space. These vector representations are of significant importance. Lot of research has been made to train these vectors efficiently based on word frequency and grouping similar words (called distributed representation of words)[2] but no significant improvements were seen. This was due to limited availability of high quality training data. Word2vec is a prediction based algorithm to efficiently represent these vectors and therefore to maximize the accuracy of results. It extends the concept of grouping of similar words.

There are two types of language models : (i) Count based (ii) Continuous space based. Count based are N-gram models using Markov assumptions which are inefficient when training data is slightly altered. For example "I am watching a movie" is independent of "A movie is watched by me on television". Continuous space based models are widely used which represent words as continuous vectors. Popular continuous space based models are Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA), neural network language models (NNLM). It has been shown that NNLM perform better than LSA while LDA is computationally expensive on large datasets. Feedforward NNLM evaluate word feature vectors and probability function values side by side. Its disadvantage is that it is time consuming. Recurrent NNLM (RNNLM) makes recurrent use of previous computations. RNNLM cannot handle large context problem as the learning rate becomes slow and predictions are done mostly on recent seen words. It is shown that neural networks trained on distributed representation of words perform better than LDA and LSA[3][4].

Word2vec is a technique to effectively handle the input data to reduce complexity of model. It is not a deep neural network but instead it is a method to represent word into vectors that can be understood by deep neural networks. It gives probabilities to words and design the structure in such a way that the words which are nearby in a sentence are nearby in vector space too. In this way the deep neural networks identify similarities between words. These similarities are both semantic-wise and syntactic-wise. Word2vec consists of two types of models: (i) Continuous Bag of Words (CBOW) model (ii) Continuous Skip Gram model. CBOW makes the content as subject and outputs the target word whereas Skip Gram makes a word as subject and outputs the target context.

CBOW predicts the probability of a word given words or group of words. It uses continuous distribution of existing standard bag of words model. Skip Gram is a continuous evaluating model which learns one value, sends back the errors and again the model learns a new value. The process is repeated until all the conditions are satisfied. In Skip Gram model, the main equation is

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

where $w_1, w_2, ..., w_T$ is a sequence of training words and c is the size of training set. The function $\log p(w_{t+j}|w_t)$ is calculated using softmax function where softmax function can be estimated by hierarchical softmax in which number of computations become less.

Skip Gram is capable of classifying similar words. For example Switzerland is similar to France. It can also perform syntactic word relations. For example, it can compute the vector operations as vec(read) - vec(reading) + vec(ask) = vec(asking). It can also perform semantic word relations. For example, it can compute the vector operations as vec(female) - vec(male) + vec(boy) = vec(girl).

Microsoft Research Sentence Completion Challenge has been introduced as a task for benchmarking of NLP models and techniques. Combination of RNNLM models currently holds state of the art performance of 55.4 percent accuracy. When Word2vec with Skip Gram model was applied in combination with RNNLM, benchmark accuracy of 58.9 percent was achieved. Accuracy can be further increased by increasing space dimensions and training data simultaneously.

Existing NLP models were complex and time consuming. Word2vec is an efficient algorithm which solves model complexity and time sustainability problem simultaneously. When word vectors are trained using Word2vec and applied on simple model network then results show it can outperform popular neural networks. Combination of Word2vec trained vectors with neural networks, specifically RNNLM, yields higher efficiency. Currently, the paper does not consider phrases and multi-words (example: South America) as input words. It also doesn't consider morphology of words which can lead to inaccuracies. These topics are candidates for future research.

References

[1] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. ICLR Workshop, 2013.

[2] David E Rumelhart, Geoffrey E Hintont, and Ronald J Williams. *Learning representations by backpropagating errors.* Nature, 323(6088):533–536, 1986.

[3] T. Mikolov, W.T. Yih, G. Zweig. Linguistic Regularities in Continuous Space Word Representations. NAACL HLT 2013.

[4] A. Zhila, W.T. Yih, C. Meek, G. Zweig, T. Mikolov. Combining Heterogeneous Models for Measuring Relational Similarity. NAACL HLT 2013.