A Study on the CBOW Model's Overfitting and Stability

Qun Luo
Beijing University of Posts and Telecommunications
10 Xitucheng road
Beijing, China
86-15210830278
luoqun9191@163.com

Weiran Xu
Beijing University of Posts and Telecommunications
10 Xitucheng road
Beijing, China
86-13041219475
xuweiran@bupt.edu.cn

Jun Guo
Beijing University of Posts and Telecommunications
10 Xitucheng road
Beijing, China
86-010-62283295
guojun@bupt.edu.cn

ABSTRACT

Word vectors are distributed representations of word features. Continuous Bag-of-Words Model(CBOW) is a state-of-the-art model for learning word vectors, yet can be ameliorated for learning better word vectors because we find that CBOW is vulnerable to be overfitted and unstable. We use two methods to solve these two problems so that CBOW can learn better word vectors. In this study, we add the regularized structure risk summation to the objective function of the CBOW model and propose inverse word frequency encoding for the CBOW model. Our proposed methods substantially improve the quality of word vectors, boosting r from 0.638 to 0.696 for word relatedness and total accuracy from 30.80% to 38.43% for word pairs relationship relatedness regarding to 52 million training words with 200 dimensionality.

Categories and Subject Descriptors

I.2.7 [**Natural Language Processing]**: Language models

General Terms

Algorithms, Experimentation, Theory, Verification

Keywords

Word representations; Continuous Bag-of-Words Model; regularization; inverse word frequency; semantic; syntactic

1. INTRODUCTION

Computing relatedness between "France" and "Italy" is described as word relatedness. Computing relatedness between word pairs "big-bigger" and "small-smaller" is described as word pairs relationship relatedness. There are many researchers focusing on semantic and semantic-syntactic relatedness and they have achieved remarkable success [7, 15] .

It has become a research hotspot to solve these problems using distributed representation methods in recent years. Distributed representation is first proposed in 1986 [9]. Distributed representation can be used in many NLP areas such as POS tagging, chunking, named entity recognition, semantic and syntactic relatedness, sentiment analysis, and so on [4, 12, 20]. In recent years, distributed representation of words as continuous vectors has made a great progress in these NLP areas [2, 3, 10, 14, 18]. It has been proved that distributed representations based on neural networks significantly outperform N-gram models [13, 15, 17, 19]. The state-of-the-art model of distributed representation is CBOW and Skip-gram proposed in 2013 [15]. Since neural networks are apt to be overfitted and unstable, we suppose that the CBOW model need to be ameliorated for learning better word representation. To prevent overfitting problem, we add regularized structure risk summation to the objective function of CBOW model. We also use inverse word frequency for Huffman encoding to make the CBOW model more stable and improve the quality of word vectors. Empirical results have proved our methods superior to original CBOW model in both word relatedness and word pairs relatedness with relatively more stability.

There are two main contributions in this paper. (1)We use regularized structure risk summation to prevent overfitting problem in CBOW model, which improves the quality of word vectors significantly; (2)We use inverse word frequency for Huffman encoding, which makes the model more stable and improves the quality of word vectors when training data is large enough.

2. Continuous Bag-of-Words Model

Neural network language model(NNLM) is very popular in recent years. The neural probabilistic language model which consists of input, projection, hidden and output layers, was first proposed in 2003 [2]. The CBOW model proposed by Mikolov is similar to the feed forward NNLM where the non-linear hidden layer is removed and the projection layer is shared for all words [15]. The CBOW model uses n words from the history and n words from the future to predict the current word. The CBOW model counts word frequency of each word in the training corpus to perform Huffman encoding, so every word is represented by a code composed by "0" and "1". As Huffman encoding proceeds, the order of every bit of code is recorded. [15] showed the architecture of CBOW model with n = 2 (n is the length of history words and length of future words). Figure 1 shows that CBOW uses words w(t-2), w(t-1), w(t+1), w(t+2) to predict w(t). The CBOW model uses the input word vectors to predict code[t], code[t] is the Huffman code of w(t). For example, when we predict ith bit of code[t], the objective function is as follows:

$$\begin{aligned}&\underset{x,P}{\operatorname{argmin}}(-\log(likelihood))\\&\qquad=\underset{x,P}{\operatorname{argmin}}(-code[t][i]\log(1-F)\\&\qquad-(1-code[t][i])\log(F)) \qquad (1)\end{aligned}$$

Web-KR'14, November 3, 2014, Shanghai, China.
Copyright © 2014 ACM 978-1-4503-1606-4/14/11…$15.00.
http://dx.doi.org/10.1145/2663792.2663793

where f = $e^{-\sum_{t^*} x(t^*)P[T_i]}$; F = f / (1+f); likelihood = $(1-F)^{code[t][i]}(F)^{1-code[t][i]}$, x(t*) is the word vector of word w(t*), t*∈(t-2,t-1,t+1,t+2), P is the matrix of projection layer, n is the length of code[t], T1,T2...Tn is the order of each bit of code[t], Ti is the order of i^{th} bit of code[t](P[Ti] is used when predicting i^{th} bit of code[t]).

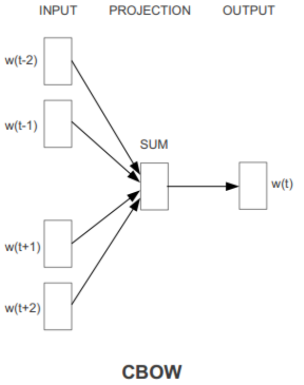

Figure 1. Architecture of CBOW model.

3. Ameliorations on CBOW model

Neural networks have been widely used in language modeling although they are easy to be overfitted and unstable in some cases as the predictors. Since CBOW is similar in principle to the neural network based models, it suffers from the same problems. As the information content of a training sample is ordinarily not sufficient by itself to reconstruct the unknown input–output mapping uniquely, a learning machine has a high possibility of overfitting [8]. To solve the overfitting problem, the method of regularization has been commonly used. Similar to the neural networks, we suppose that the CBOW model is unstable too, which means that small changes in the training data may result in significantly different models [5].

3.1 Regularized structure risk summation

The CBOW model is designed to learn the input word vectors x and projection layer P, as shown in Formula (1). Since both x and P are the parameters which can be overfitted by the training data, the CBOW model has an even higher possibility to be unstable. To overcome this drawback, we propose to employ the x's regularizer and P's regularizer in the objective function simultaneously. Thus, we redesign the objective function as follows:

$$\begin{aligned}\underset{x,P}{\operatorname{argmin}} \Big(&-code[t][i]\log(1-F)\\ &-(1-code[t][i])\log(F)\\ &+c_1\sum x_{ij}^2 + c_2\sum P_{ij}^2\Big)\end{aligned} \qquad (2)$$

where c_1 and c_2 are hyperparameters as penalty factors. The regularizers, which we call regularized structure risk summation, are intended to smooth the solution to word vectors x and projection layer P. As a result, through an appropriate choice of the regularization parameters(c_1 and c_2), the new objective function provides a tradeoff between the "fidelity" of the training data and the "smoothness" of solution, which could overcome the problem of overfitting [8].

3.2 Feature selection for encoding

The number of parameters about a word is shown in Formula (3).

$$(N+1)*D \qquad (3)$$

where N is the length of the word's code, D is the dimensionality of word vector.

The CBOW model, uses word frequency for Huffman encoding. This means that the higher the word's frequency, the shorter the length of its code. If the length of a word's code is shorter, there are less parameters about this word need to be learned because this word uses less projection layers. As a result, the word that has more training data needs to learn less parameters and the word that has less training data needs to learn more parameters. We suppose that this is contrary to the principles of machine learning. To verify this consideration, we propose a method to perform inverse word frequency Huffman encoding. Namely, we use inverse word frequency as feature for Huffman encoding. Specially, we first count each word's frequency, and record the max frequency, after that we use the number of max frequency to minus the word's frequency for Huffman encoding of a given word. Using the inverse word frequency Huffman encoding, a word that has less training data will learn less parameters, so that we expect this will make the model more stable and improve the quality of word vectors.

4. Experiments and Analysis

4.1 Datasets and Evaluation Procedure

We use two datasets as training data in our experiments. The first one [1] is downloaded from Mikolov's word2vec tools which consists of 17 million words. The second one is from Wikipedia. To assess quality of our word vectors, we use two methods. The first method is word relatedness. We use the WordSimilarity-353 collection[2] [6], which contains 353 word pairs and each pair has a single relatedness score averaged by 13-16 human judgements. Spearman rank-order correlation coefficient was used to compare computed relatedness scores with human judgements [1, 7, 11]. The second method is word pairs relationship relatedness. Mikolov observes that there can be many different types of similarities between words, for example, word "big" is similar to "bigger" in the same sense as "small" is similar to "smaller" [16]. We use both questions and assessing program(compute-accuracy) created by Mikolov, and there are 8869 semantic questions of five types and 10675 syntactic questions of nine types[3]. We evaluate the overall accuracy for all question types which is same with Mikolov.

We let c_1 equal c_2 in our experiments. To find the optimal value of c_1 and c_2, we tried three different values(0.05, 0.005, 0.0005) in experiments on small datasets. We found that 0.005 is the best of the three, so 0.005 was used in the experiments. Besides regularization, we also suppose that the length of window is important to CBOW model. In the Mikolov's implementation of CBOW, the default window length is a random number between 1 and 5. We do some experiments with the window length a fixed number(5 in our experiments).

4.2 Experimental results of regularization

We denote F-CBOW as CBOW with fixed window length (equals 5), FR-CBOW as F-CBOW with regularization, TW as training words, Sem as semantic accuracy, Syn as syntactic accuracy, Tot as total accuracy. We compare our word vectors with Mikolov's word vectors both in word relatedness and word pairs relationship

[1] http://mattmahoney.net/dc/text8.zip

[2] http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/

[3] https://code.google.com/p/word2vec/

relatedness. The dimensionality of word vectors is 200. All methods are trained for one epoch on the same data. All the accuracy in this subsection is reported on the most frequent 30k words.

Table 1. Word relatedness correlation with humans

method	TW	Correlation with humans
CBOW	17M	0.606
F-CBOW	17M	0.637
FR-CBOW	17M	**0.665**
CBOW	52M	0.638
F-CBOW	52M	0.646
FR-CBOW	52M	**0.696**

Table 2. Word pairs relationship relatedness accuracy

method	TW	Sem %	Syn %	Tot %
CBOW	17M	13.89	19.63	17.74
F-CBOW	17M	16.52	21.96	20.14
FR-CBOW	17M	**21.94**	**24.51**	**23.65**
CBOW	52M	15.30	38.82	30.80
F-CBOW	52M	22.35	38.82	33.20
FR-CBOW	52M	**30.37**	**42.59**	**38.43**

Table 3. Part of Mikolov's published results on total accuracy

TW dimensionality	49M	391M	783M
50	15.7	22.5	23.2
100	23.1	23.4	32.2
300	29.2	43.7	45.9
600	30.1	46.6	50.4

Table 4. FR-CBOW's results on total accuracy

TW dimensionality	52M	413M
200	38.43	52.46
300	41.37	54.52
400	41.22	55.86

According to Table 1 and Table 2, we can see that the model with regularization performs best in estimating both word relatedness and word pairs relationship relatedness. Because the training data is usually not enough to learn the word vectors, the model may be overfitted by the training data. Adding regularization can partly solve the problem of overfitting.

Mikolov published his results on total accuracy of word pairs relationship using CBOW[4]. Part of results is shown in Table 3. Comparing Table3 with Table 4, remarkable improvement has been made(from total accuracy = 45.9% to 54.52%) with the same size of dimensionality(300) and less training words(413M vs 783M).

To compare the stability of different models, we set the length of window a random number(same with Mikolov's word2vec) so that the training data will change little every time. We observe how the Spearman rank-order correlation coefficient changes in different times. We denote R-CBOW as CBOW with regularization. Every model was trained ten times on the same dataset(17M training words with 200 dimensionality). As shown in Figure 2, both CBOW and R-CBOW are unstable as the window length is a random number for a given word. R-CBOW is relatively more stable as the standard deviation is smaller(CBOW is 0.0166, R-CBOW is 0.0136).

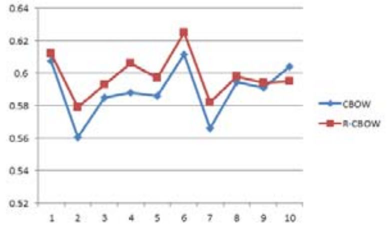

Figure 2. Changes on correlation.

Table 5. Limited accuracy and Full accuracy on F-CBOW and FI-CBOW

method	TW	Limited accuracy	Full accuracy
F-CBOW	17M	**20.14**	11.13
FI-CBOW	17M	17.61	**11.21**
F-CBOW	52M	**33.20**	20.29
FI-CBOW	52M	32.32	**20.52**
F-CBOW	145M	36.47	24.98
FI-CBOW	145M	**37.38**	**25.94**
F-CBOW	413M	45.31	32.69
FI-CBOW	413M	**46.03**	**33.71**

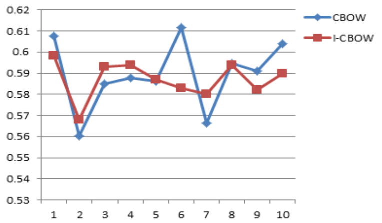

Figure 3. Changes on correlation.

4.3 Experimental results of inverse word frequency

We denote FI-CBOW as F-CBOW with inverse word frequency. We compare F-CBOW and FI-CBOW in word pairs relationship relatedness. Limited accuracy means the most frequent 30k words used. Full accuracy means all vocabulary used. The results of word pairs relationship accuracy are shown in Table 5(dimensionality is 200).

As shown in Table 5, when the training data is small, FI-CBOW has lower limited accuracy but higher full accuracy. As the training data grows, both limited accuracy and full accuracy of FI-CBOW are higher than those of F-CBOW. As discussed in subsection 3.2, when using inverse word frequency, the higher the word's frequency, the longer the length of its code, so inverse word frequency is not good for frequent words. The lower the word's frequency, the shorter of its code, thus, inverse word frequency is better for infrequent words. As a result, when the training data is large enough, it is better to use inverse word frequency for Huffman encoding so that both frequent and infrequent words have enough training data to learn their parameters. The experiments suggest it is more reasonable to learn

[4] Efficient Estimation of Word Representations in Vector Space(on this paper's table 2)

more parameters for more frequent words and less parameters for less frequent words.

We denote I-CBOW as CBOW with inverse word frequency. We use the same method discussed in subsection 4.2 to analyze the stability of CBOW and I-CBOW. Each model is also trained ten times on the same dataset (17M training words with 200 dimensionality). We analyze the changes on Spearman rank-order correlation coefficient of ten times. As can been seen in Figure 3, Huffman encoding with inverse word frequency could make the model more stable.

5. Conclusion

We present two simple and effective methods to elevate the performance of the CBOW model. Our empirical results show that regularization can improve the quality of word vectors significantly and inverse word frequency can improve quality of word vectors relatively. As inverse word frequency is better for infrequent words, the improvement could be more obvious if questions collection contains more infrequent words and the training data is large enough. Since similar words have similar vector space, word vectors can be used in knowledge extraction and knowledge base construction.

Maybe there are many features can be used for Huffman encoding, or we can use other encoding methods to make the model stable and improve the quality of word vectors. We suppose both feature selection and encoding methods are important to the model, thus, our future works will focus on these problems in order to propose better models for learning word vectors.

6. ACKNOWLEDGMENTS

This work was supported by 111 Project of China under Grant No. B08004, key project of ministry of science and technology of China under Grant No. 2011ZX03002-005-01, National Natural Science Foundation of China (61273217) and the Ph.D. Programs Foundation of Ministry of Education of China (20130005110004).

7. REFERENCES

[1] Agirre E, Alfonseca E, Hall K, et al. A study on similarity and relatedness using distributional and WordNet-based approaches[C]//Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2009: 19-27.

[2] Bengio Y, Ducharme R, Vincent P, et al. A Neural Probabilistic Language Model[J]. Journal of Machine Learning Research, 2003, 3: 1137-1155.

[3] Collobert R, Weston J. A unified architecture for natural language processing: Deep neural networks with multitask learning[C]//Proceedings of the 25th international conference on Machine learning. ACM, 2008: 160-167.

[4] Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch[J]. The Journal of Machine Learning Research, 2011, 12: 2493-2537.

[5] Cunningham P, Carney J, Jacob S. Stability problems with artificial neural networks and the ensemble solution[J]. Artificial Intelligence in Medicine, 2000, 20(3): 217-225.

[6] Finkelstein L, Gabrilovich E, Matias Y, et al. Placing search in context: The concept revisited[C]//Proceedings of the 10th international conference on World Wide Web. ACM, 2001: 406-414.

[7] Gabrilovich E, Markovitch S. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis[C]//IJCAI. 2007, 7: 1606-1611.

[8] Haykin S. Neural networks and learning machines[M]. Upper Saddle River: Pearson Education, chapter 7, pages 314-366, 2009.

[9] Hinton G. Learning distributed representations of concepts[C]//Proceedings of the eighth annual conference of the cognitive science society. 1986: 1-12.

[10] Huang E H, Socher R, Manning C D, et al. Improving word representations via global context and multiple word prototypes[C]//Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. Association for Computational Linguistics, 2012: 873-882.

[11] Luong M T, Socher R, Manning C D. Better word representations with recursive neural networks for morphology[J]. CoNLL-2013, 2013, 104.

[12] Maas A L, Daly R E, Pham P T, et al. Learning word vectors for sentiment analysis[C]//Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011: 142-150.

[13] Mikolov T, Deoras A, Kombrink S, et al. Empirical Evaluation and Combination of Advanced Language Modeling Techniques[C]//INTERSPEECH. 2011: 605-608.

[14] Mikolov T. Statistical Language Models based on Neural Networks. PhD thesis, Brno University of Technology, 2012.

[15] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013a.

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

[17] Mnih A, Hinton G. Three new graphical models for statistical language modelling[C]//Proceedings of the 24th international conference on Machine learning. ACM, 2007: 641-648.

[18] Mnih A, Hinton G. A Scalable Hierarchical Distributed Language Model[C]//NIPS. 2008: 1081-1088.

[19] Schwenk H. Continuous space language models[J]. Computer Speech & Language, 2007, 21(3): 492-518.

[20] Turian J, Ratinov L, Bengio Y. Word representations: a simple and general method for semi-supervised learning[C]//Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, 2010: 384-394