CSLT TECHNICAL REPORT-2015XXXX [Monday 11th April, 2016]

Low-Frequency Words Embedding

Chao Xing[1,2], Yiqiao Pan[1,2] and Dong Wang[1,2*]

*Correspondence: wang-dong99@mails.tsinghua.edu.cn
[1]Center for Speech and Language Technology, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China
Full list of author information is available at the end of the article

Abstract

Most of words are low frequency in natural languages, due to the Zipf's law. The low frequency words are often entity names and hence convey significant semantic load, however due to the limited occurrences, it is difficult to build statistical model for them. This technique report presents some studies on low-frequency word treatment conducted in our team. Our study is based on low-dimensional embeddings with neural network framework. Six strategies are presented and compared in this report, and we found that a two-step approach, that embeds high-frequency words at first and then predict low-frequency word embeddings with high-frequency words, is highly effective. Our experiments show that this approach can get state-of-art performance on word similarity and text classification tasks.

Keywords: low frequency words; word vector; word similarity; document classification; sentiment classification

1 Introduction

In recent years, scientists' eyes focused on neural based word representation, and many novel models had been proposed. Treating word as continuous vector space does promote the performance in many NLU(nature language understanding) tasks. Despite these remarkable achievements there is still a acuity problem in word representation. We can easily handle high frequency words in statistics models for that their contexts are plentiful ,and high frequency words' information are enough for the training model. However, low frequency words has rare information, after training , the low frequency words can almost be regarded as randomly to be other words which appear more often. We center our experiments among word similarity, text classification, and sentiment classification which are one of the most important tasks in NLU, to show that the largest bottleneck of performance improvement is low frequency words.

2 Related work

In word representation, traditional method is so-called bag of word. The bag of word method treat word as a one-hot vector, which is a very large sparse vector and there are no connection between words. So that we can use bag-of-words method to represent a document , which means document vectors are extracted from combining one-hot word vectors together. Also,there are some other models which can generate document vectors, such as LDA ,indicating document vector by estimating the words topic representation,or NN and CNN ,indicating document vector by neural network models. Although traditional methods have a series of achievements, word representation suffered from high compute time costing, large storage costing, and

also the performance still not satisfied. Recently word represent method tends to be pointed on continuous, low dimension, and highly abstract. The most popular one is word2vec tool, proposed by Mikolov. et. al, using context to represent a word. There are two ways in word2vec, CBOW(continuous bag of words) and Skip-gram. As we argued before, word2vec has some strategies for balance word frequency. One is they use the method named sub-sampling to randomly drop high frequency words during training process. Another one is they drop the words which frequency are lower than a hyper-parameter mini-count named mini-count strategy . Both of them show efficiency in some tasks. But it still not satisfied us,for that despite some high performance in some particular areas, word2vec model can not generate low frequency words with high performance .

3 Theory

We proposed six strategies in this paper, compared with original word2vec. The basic idea is ignore the low frequency words first while training, and just train the high frequency words themselves. Then generate low frequency word vectors after training process rather than dropping them away. The experiment shows that the representation quality is not always proportional to the count of cut down low frequency words. There are two fundamental method to deal with low frequency words. One is to predict the low frequency words given it's context information, this method is quickly, and can show approximate results to other's. The other is retrain the low frequency by initialize with high frequency words. Low frequency means a word appears rarely for a model to deal with it, so we can use context information to give an dark representation. Or given a similar word's vector derived from knowledge.

3.1 Strategy-1 : Predict low frequency words and context consider low frequency words

In this strategy, our assumption is low frequency words can be reduced by their context. The context should contain nature fixed window size. This assumption fit to word2vec training process. And we can easily get the result.

3.2 Strategy-2 : Predict low frequency words and context do not consider low frequency words

In strategy 2 we consider that low frequency words' context shouldn't contain low frequency words, this is because low frequency word vectors initialized randomly so if considering these context cover low frequency words makes no sense but add some rubbish knowledge in low frequency word vectors.

3.3 Strategy-3 : Retrain low frequency words and fine tune high frequency words

Above strategies are all based on predict structure. And at the same time, one can easily find there are another way to generate low frequency words by retraining word vectors. The method retrain low frequency words and high frequency words. The basic idea is when change low frequency word vectors, the high frequency word vectors should change equally to maintain the whole system be stable.

3.4 Strategy-4 : Retrain low frequency words and do not fine tune high frequency words

Strategy 4 is opposite to strategy 3, in this one we think while high frequency word vectors had already trained well, we should not do any change to high frequency words, and in another words low frequency words are almost random at first, it may ull high frequency to a wrong point in vector space. So consist high frequency word vectors, just update low frequency words seems make sense.

3.5 Strategy-5 : Retrain low frequency words, fine tune high frequency words and initial word vector given predict low frequency words

Above four strategies explain two main ways to illustrate our idea, these two strategies below combine two ways together. The fifth strategy is that initial word vectors in both pre-trained high frequency word vectors and predicted low frequency word vectors. This method will train both high frequency word vectors and low frequency word vectors.

3.6 Strategy-6 : Retrain low frequency words, do not fine tune high frequency words and initial word vector given predict low frequency words

The last strategy is retrain low frequency word vectors initialized by predict low frequency word vectors. In this strategy we consist high frequency word vectors, and just retrain low frequency words,for the assumption that if we change high frequency words, they can be pull to a wrong point in vector space.

4 Experiment

We would like to show the generality of our strategies, so we carried out experiments on four different type word vector. Two of them are trained by C-bow model with one is by softmax method and the other one is by negative sampling method. Meanwhile,other two word vectors are trained by skip model with one is softmax method and the other one is negative sampling method.

C-bow skip

Softmax method is to control freedom. Word2vec model has many random process,so during the training process, we just use hierarchical softmax method in which given a same corpus the results are same. While use hierarchical softmax method we should build a basic Huffman tree and ignore smaller frequency than a given hyper-parameter words during training.

Negative sampling method can largely accelerate computation speed but with relatively random essence compared with softmax method. In initial process all word vectors are random initialized, in negative sampling process negative words are random sampled , and in sub-sampling process high frequency word are random dropped by a hyper-parameter.

In our experiment we set the basic Huffman tree build by mini-count 5. And we carry out our experiments centering among two tasks, which is word similarity, and text classification. Simultaneously, the six strategies has been tested on the two tasks with nine different kinds of word vectors with four different types of word vectors.

4.1 Experiment set

We train word vectors based on fine kind of corpora on the two different tasks.

For the task of similarity, there are two standard corpora which are Text8 and Wiki News 2011. Text8 proposed by google is a 100M corpus contains 71291 words which frequency are larger than 5 . And Wiki News 2011 is a set of news with a capacity of 1G.

Meanwhile, for text classification task, corpora are reuters52, reuters8 and 20news which are all common corpora . 20news dataset is a collection of approximately 20,000 newsgroup documents, with 11M for model training and 7M for document classification. Reuters52 is a totally 4M corpus which contains 3M text for model training, and 1M for document classification test. Also, Reuters8 has 2M text for training and 1M for test.

4.2 Word Similarity

Word similarity is a common nature language understanding task. In this part, we test word vector representation quality by three judge sets RG65, SIM353, SIM999.RG65, SIM353, SIM999 are proposed by linguists. For example, linguists judge and score the similarity between two words through their professional knowledge and experience.SIM353 has 353 word pairs with human similarity ratings . In the same way, SIM999 lists 999 pairs words.

In this table we show different mini-count's coefficient and Spielman coefficient.We set the basic Huffman tree build by mini-count 5, and list the results for parameter mini-count 5, 15 and 25. The first line shows initial results .For the line 2 and line 3, "con" means "consider" and line 3 means when predict low frequency words we also consider low frequency words, and they are strategy 1 and strategy 2. Line 4 and line 5 mean different retrain strategies. And line 6 and line 7 which has been explained above for strategy 3 and strategy 4. The next two lines are mainly about strategy 5 and strategy 6. The different parts are the initial word vectors in the retrain processing which can be the baseline, predicted word vectors or con-predicted word vectors.

Coefficient	min count 5						min count 15				
	rg65	sim353	sim999	scws	mc30	men3000	rg65	sim353	sim999	scws	mc30
Ini-Train	0.5897	0.5929	0.2280	0.6126	0.5903	0.5658	0.5315	0.6004	0.2281	0.6086	0.595
Predict	0.5897	0.5929	0.2280	0.6126	0.5903	0.5658	0.5386	0.6031	0.2351	0.6164	0.553
Con-Predict	0.5897	0.5929	0.2280	0.6126	0.5903	0.5658	0.5386	0.6031	0.2351	0.6164	0.553
Ini-Retrain-With	0.5820	0.5937	0.2366	0.6101	0.6112	0.5645	0.5751	0.6066	0.2297	0.6151	0.607
Ini-Retrain-Without	0.5915	0.5932	0.2278	0.6126	0.5930	0.5658	0.5760	0.6080	0.2349	0.6164	0.589
Pre-Retrain-With	0.5997	0.5949	0.2321	0.6107	0.6240	0.5623	0.5793	0.6027	0.2380	0.6144	0.613
Pre-Retrain-Without	0.5914	0.5931	0.2278	0.6127	0.5923	0.5658	0.5816	0.6068	0.2355	0.6162	0.579
Con-Pre-Retrain-With	0.6132	0.5976	0.2388	0.6118	0.6327	0.5646	0.5913	0.6106	0.2431	0.6160	0.636
Con-Pre-Retrain-Without	0.5903	0.5928	0.2278	0.6126	0.5893	0.5658	0.5847	0.6077	0.2360	0.6165	0.582

Table 1 ../cbow-tree/text8

4.3 Text Classification

Text classification is a quite significant task in the field of nature language understanding. In this part, we generate document vectors just from the average pooling strategy. And we test the classification accuracy by classifier of Gaussian Naive Bayes (Gaussian NB), Bernoulli NB,KNeighborsClassifier,SVM classifier with liner kernel and LogisticRegression.

Coefficient	min count 5						min count 15				
	rg65	sim353	sim999	scws	mc30	men3000	rg65	sim353	sim999	scws	mc30
Ini-Train	0.5596	0.6246	0.3013	0.6417	0.6163	0.6108	0.5530	0.6332	0.2737	0.6304	0.609
Predict	0.5596	0.6246	0.3013	0.6417	0.6163	0.6108	0.5528	0.6352	0.2900	0.6373	0.579
Con-Predict	0.5596	0.6246	0.3013	0.6417	0.6163	0.6108	0.5528	0.6352	0.2900	0.6373	0.579
Ini-Retrain-With	0.5816	0.6318	0.2995	0.6376	0.6520	0.6167	0.5761	0.6530	0.3025	0.6376	0.621
Ini-Retrain-Without	0.5585	0.6243	0.3013	0.6416	0.6132	0.6108	0.5288	0.6381	0.2895	0.6372	0.601
Pre-Retrain-With	0.5673	0.6399	0.3012	0.6373	0.6271	0.6208	0.6010	0.6516	0.3037	0.6374	0.636
Pre-Retrain-Without	0.5584	0.6239	0.3013	0.6415	0.6094	0.6108	0.5535	0.6360	0.2938	0.6385	0.579
Con-Pre-Retrain-With	0.5806	0.6317	0.3022	0.6379	0.6356	0.6184	0.5871	0.6487	0.3030	0.6374	0.609
Con-Pre-Retrain-Without	0.5590	0.6243	0.3014	0.6416	0.6132	0.6108	0.5528	0.6365	0.2938	0.6387	0.577

Table 2 ../cbow-random/text8

Coefficient	min count 5						min count 15				
	rg65	sim353	sim999	scws	mc30	men3000	rg65	sim353	sim999	scws	mc30
Ini-Train	0.5528	0.5761	0.3391	nan	0.6025	0.6121	0.5100	0.5750	0.3400	nan	0.606
Predict	0.5528	0.5761	0.3391	0.6303	0.6025	0.6121	0.5707	0.5771	0.3400	0.6273	0.606
Con-Predict	0.5528	0.5761	0.3391	0.6303	0.6025	0.6121	0.5707	0.5771	0.3400	0.6273	0.606
Ini-Retrain-With	0.5792	0.5725	0.3345	0.6291	0.6341	0.6108	0.5830	0.5747	0.3359	0.6294	0.622
Ini-Retrain-Without	0.5528	0.5761	0.3391	0.6302	0.6025	0.6121	0.5604	0.5776	0.3400	0.6287	0.606
Pre-Retrain-With	0.5687	0.5691	0.3341	0.6290	0.6213	0.6109	0.5856	0.5705	0.3346	0.6286	0.618
Pre-Retrain-Without	0.5528	0.5761	0.3391	0.6308	0.6025	0.6121	0.5682	0.5779	0.3400	0.6290	0.606
Con-Pre-Retrain-With	0.5677	0.5719	0.3353	0.6291	0.6263	0.6101	0.5824	0.5727	0.3336	0.6288	0.635
Con-Pre-Retrain-Without	0.5528	0.5761	0.3391	0.6306	0.6025	0.6121	0.5739	0.5774	0.3400	0.6289	0.606

Table 3 ../cbow-random/wiki

Coefficient	min count 5						min count 15				
	rg65	sim353	sim999	scws	mc30	men3000	rg65	sim353	sim999	scws	mc30
Ini-Train	0.6554	0.6705	0.2580	0.6440	0.6484	0.6453	0.5898	0.6502	0.2270	0.6262	0.635
Predict	0.6554	0.6705	0.2580	0.6440	0.6484	0.6453	0.6856	0.6647	0.2559	0.6412	0.654
Con-Predict	0.6554	0.6705	0.2580	0.6440	0.6484	0.6453	0.6856	0.6647	0.2559	0.6412	0.654
Ini-Retrain-With	0.6590	0.6647	0.2612	0.6424	0.6436	0.6480	0.6527	0.6632	0.2624	0.6449	0.630
Ini-Retrain-Without	0.6524	0.6698	0.2571	0.6441	0.6420	0.6451	0.6374	0.6576	0.2521	0.6402	0.574
Pre-Retrain-With	0.6646	0.6628	0.2595	0.6421	0.6508	0.6467	0.6566	0.6613	0.2617	0.6456	0.637
Pre-Retrain-Without	0.6526	0.6698	0.2572	0.6441	0.6422	0.6451	0.6472	0.6592	0.2500	0.6414	0.591
Con-Pre-Retrain-With	0.6563	0.6649	0.2599	0.6423	0.6450	0.6475	0.6312	0.6526	0.2606	0.6450	0.600
Con-Pre-Retrain-Without	0.6505	0.6693	0.2571	0.6440	0.6375	0.6451	0.6421	0.6597	0.2494	0.6415	0.597

Table 4 ../skip-tree/text8

Coefficient	min count 5						min count 15				
	rg65	sim353	sim999	scws	mc30	men3000	rg65	sim353	sim999	scws	mc30
Ini-Train	0.6040	0.6564	0.2786	0.6313	0.6341	0.6561	0.4595	0.6301	0.2214	0.6002	0.574
Predict	0.6040	0.6564	0.2786	0.6313	0.6341	0.6561	0.6368	0.6654	0.2710	0.6278	0.706
Con-Predict	0.6040	0.6564	0.2786	0.6313	0.6341	0.6561	0.6368	0.6654	0.2710	0.6278	0.706
Ini-Retrain-With	0.5995	0.6524	0.2792	0.6311	0.6149	0.6537	0.6080	0.6671	0.2744	0.6340	0.631
Ini-Retrain-Without	0.6059	0.6569	0.2787	0.6314	0.6387	0.6561	0.6066	0.6620	0.2637	0.6323	0.656
Pre-Retrain-With	0.5939	0.6549	0.2747	0.6318	0.6214	0.6552	0.5974	0.6645	0.2741	0.6335	0.635
Pre-Retrain-Without	0.6059	0.6570	0.2787	0.6314	0.6389	0.6561	0.6037	0.6610	0.2636	0.6322	0.649
Con-Pre-Retrain-With	0.5943	0.6570	0.2746	0.6317	0.6186	0.6545	0.6113	0.6651	0.2727	0.6333	0.630
Con-Pre-Retrain-Without	0.6060	0.6570	0.2787	0.6314	0.6390	0.6561	0.6042	0.6613	0.2634	0.6322	0.651

Table 5 ../skip-random/text8

Coefficient	min count 5						min count 15				
	rg65	sim353	sim999	scws	mc30	men3000	rg65	sim353	sim999	scws	mc30
Ini-Train	0.6400	0.6155	0.3606	nan	0.6455	0.6642	0.5645	0.6187	0.3568	nan	0.625
Predict	0.6400	0.6155	0.3606	0.6239	0.6455	0.6642	0.6173	0.6208	0.3568	0.6202	0.625
Con-Predict	0.6400	0.6155	0.3606	0.6239	0.6455	0.6642	0.6173	0.6208	0.3568	0.6202	0.625
Ini-Retrain-With	0.6293	0.6146	0.3597	0.6246	0.6361	0.6645	0.6290	0.6218	0.3574	0.6268	0.630
Ini-Retrain-Without	0.6400	0.6155	0.3606	0.6251	0.6455	0.6642	0.6167	0.6207	0.3568	0.6245	0.625
Pre-Retrain-With	0.6254	0.6149	0.3597	0.6254	0.6319	0.6650	0.6261	0.6207	0.3568	0.6263	0.624
Pre-Retrain-Without	0.6400	0.6155	0.3606	0.6250	0.6455	0.6642	0.6164	0.6205	0.3568	0.6242	0.625
Con-Pre-Retrain-With	0.6335	0.6149	0.3597	0.6250	0.6391	0.6643	0.6274	0.6206	0.3563	0.6262	0.625
Con-Pre-Retrain-Without	0.6400	0.6155	0.3606	0.6251	0.6455	0.6642	0.6165	0.6205	0.3568	0.6242	0.625

Table 6 ../skip-random/wiki

Gaussian Naive Bayes Bernoulli Naive Bayes KNeighborsClassifier Support Vector Machine classifier LogisticRegression

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.6158	0.6528	0.6904	0.7232	0.7311	0.6207	0.6520	0.6830	0.7233	0.7294	0.6012
Predict	0.6158	0.6528	0.6904	0.7232	0.7311	0.6206	0.6581	0.6876	0.7268	0.7335	0.6036
Con-Predict	0.6158	0.6528	0.6904	0.7232	0.7311	0.6206	0.6581	0.6876	0.7268	0.7335	0.6036
Ini-Retrain-With	0.6029	0.6453	0.6824	0.7169	0.7260	0.6360	0.6621	0.6954	0.7286	0.7345	0.6404
Ini-Retrain-Without	0.6153	0.6513	0.6912	0.7224	0.7305	0.6304	0.6615	0.6949	0.7302	0.7383	0.6323
Pre-Retrain-With	0.6039	0.6420	0.6821	0.7147	0.7216	0.6350	0.6566	0.6962	0.7306	0.7384	0.6362
Pre-Retrain-Without	0.6157	0.6509	0.6905	0.7228	0.7307	0.6250	0.6652	0.6946	0.7305	0.7384	0.6277
Con-Pre-Retrain-With	0.6072	0.6403	0.6838	0.7152	0.7236	0.6304	0.6622	0.6963	0.7307	0.7391	0.6376
Con-Pre-Retrain-Without	0.6158	0.6520	0.6910	0.7226	0.7310	0.6251	0.6641	0.6966	0.7322	0.7374	0.6266

Table 7 ../cbow-tree/20ng

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.8096	0.8400	0.8715	0.9058	0.8984	0.8092	0.8590	0.8754	0.9167	0.9065	0.8076
Predict	0.8096	0.8400	0.8715	0.9058	0.8984	0.8088	0.8610	0.8750	0.9182	0.9089	0.8115
Con-Predict	0.8096	0.8400	0.8715	0.9058	0.8984	0.8088	0.8610	0.8750	0.9182	0.9089	0.8115
Ini-Retrain-With	0.8088	0.8466	0.8707	0.9058	0.8984	0.8244	0.8512	0.8695	0.9128	0.9046	0.8275
Ini-Retrain-Without	0.8096	0.8407	0.8715	0.9034	0.8999	0.8189	0.8645	0.8680	0.9116	0.9081	0.8181
Pre-Retrain-With	0.8084	0.8450	0.8727	0.9046	0.8952	0.8259	0.8532	0.8754	0.9159	0.9065	0.8252
Pre-Retrain-Without	0.8088	0.8407	0.8715	0.9034	0.8991	0.8181	0.8610	0.8695	0.9132	0.9073	0.8158
Con-Pre-Retrain-With	0.8088	0.8524	0.8762	0.9046	0.8980	0.8224	0.8548	0.8715	0.9128	0.9054	0.8275
Con-Pre-Retrain-Without	0.8096	0.8400	0.8719	0.9046	0.8995	0.8162	0.8629	0.8699	0.9128	0.9081	0.8178

Table 8 ../cbow-tree/r52

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.8954	0.9100	0.9402	0.9438	0.9507	0.8931	0.9237	0.9429	0.9575	0.9616	0.8803
Predict	0.8954	0.9100	0.9402	0.9438	0.9507	0.8995	0.9214	0.9452	0.9589	0.9621	0.8867
Con-Predict	0.8954	0.9100	0.9402	0.9438	0.9507	0.8995	0.9214	0.9452	0.9589	0.9621	0.8867
Ini-Retrain-With	0.8972	0.9054	0.9420	0.9424	0.9447	0.9036	0.9214	0.9374	0.9520	0.9575	0.9063
Ini-Retrain-Without	0.8954	0.9118	0.9392	0.9438	0.9502	0.9004	0.9283	0.9383	0.9552	0.9639	0.8945
Pre-Retrain-With	0.8977	0.9036	0.9388	0.9443	0.9470	0.9068	0.9237	0.9397	0.9548	0.9575	0.9100
Pre-Retrain-Without	0.8954	0.9123	0.9402	0.9429	0.9502	0.8995	0.9278	0.9374	0.9561	0.9625	0.8963
Con-Pre-Retrain-With	0.8990	0.9196	0.9370	0.9434	0.9466	0.9045	0.9214	0.9374	0.9561	0.9589	0.9091
Con-Pre-Retrain-Without	0.8954	0.9100	0.9397	0.9438	0.9502	0.8995	0.9269	0.9370	0.9557	0.9621	0.8963

Table 9 ../cbow-tree/r8

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.6316	0.6014	0.7030	0.7376	0.7419	0.6298	0.5979	0.6912	0.7395	0.7460	0.6206
Predict	0.6316	0.6014	0.7030	0.7376	0.7419	0.6323	0.5996	0.6966	0.7430	0.7508	0.6219
Con-Predict	0.6316	0.6014	0.7030	0.7378	0.7419	0.6323	0.5996	0.6966	0.7428	0.7508	0.6219
Ini-Retrain-With	0.6537	0.6362	0.7078	0.7350	0.7407	0.6550	0.6323	0.7040	0.7380	0.7439	0.6513
Ini-Retrain-Without	0.6311	0.6018	0.7028	0.7366	0.7416	0.6335	0.5975	0.6933	0.7415	0.7473	0.6266
Pre-Retrain-With	0.6494	0.6351	0.7068	0.7345	0.7419	0.6566	0.6389	0.7116	0.7380	0.7451	0.6534
Pre-Retrain-Without	0.6311	0.6020	0.7026	0.7368	0.7414	0.6347	0.6022	0.6955	0.7446	0.7499	0.6287
Con-Pre-Retrain-With	0.6526	0.6331	0.7083	0.7355	0.7410	0.6563	0.6367	0.7145	0.7394	0.7471	0.6536
Con-Pre-Retrain-Without	0.6311	0.6016	0.7024	0.7368	0.7414	0.6347	0.6007	0.6951	0.7444	0.7496	0.6281

Table 10 ../cbow-random/20ng

5 Conclusions

We studied a number of strategies to deal with low-frequency words in neural-based word embedding. Due to the limited statistics, it is difficult to embed low-frequency words in an appropriate position, and the low-quality of the embeddings of low-frequency words may impact the quality of high-frequency words as well. Motivated

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.8454	0.8357	0.9003	0.9194	0.9213	0.8407	0.8485	0.8937	0.9178	0.9233	0.8462
Predict	0.8454	0.8357	0.9003	0.9194	0.9213	0.8442	0.8477	0.8960	0.9198	0.9245	0.8474
Con-Predict	0.8454	0.8357	0.9003	0.9198	0.9213	0.8442	0.8477	0.8960	0.9194	0.9245	0.8474
Ini-Retrain-With	0.8512	0.8536	0.8995	0.9186	0.9248	0.8512	0.8614	0.9003	0.9143	0.9206	0.8563
Ini-Retrain-Without	0.8454	0.8364	0.9007	0.9190	0.9210	0.8411	0.8474	0.8941	0.9182	0.9237	0.8477
Pre-Retrain-With	0.8524	0.8555	0.8972	0.9155	0.9210	0.8540	0.8563	0.8991	0.9147	0.9217	0.8583
Pre-Retrain-Without	0.8454	0.8364	0.9003	0.9186	0.9210	0.8435	0.8458	0.8968	0.9202	0.9256	0.8474
Con-Pre-Retrain-With	0.8532	0.8544	0.8976	0.9174	0.9217	0.8540	0.8575	0.8980	0.9147	0.9206	0.8571
Con-Pre-Retrain-Without	0.8454	0.8364	0.9007	0.9194	0.9210	0.8435	0.8450	0.8976	0.9206	0.9252	0.8477

Table 11 ../cbow-random/r52

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.9169	0.9223	0.9603	0.9584	0.9653	0.9091	0.9114	0.9616	0.9625	0.9662	0.9063
Predict	0.9169	0.9223	0.9603	0.9584	0.9653	0.9105	0.9095	0.9653	0.9635	0.9653	0.9086
Con-Predict	0.9169	0.9223	0.9603	0.9584	0.9653	0.9105	0.9095	0.9653	0.9635	0.9653	0.9086
Ini-Retrain-With	0.9205	0.9233	0.9557	0.9548	0.9625	0.9233	0.9182	0.9598	0.9598	0.9630	0.9178
Ini-Retrain-Without	0.9173	0.9214	0.9598	0.9589	0.9653	0.9086	0.9109	0.9662	0.9625	0.9657	0.9100
Pre-Retrain-With	0.9205	0.9233	0.9543	0.9539	0.9616	0.9201	0.9196	0.9561	0.9580	0.9662	0.9191
Pre-Retrain-Without	0.9169	0.9219	0.9603	0.9584	0.9653	0.9114	0.9118	0.9648	0.9644	0.9662	0.9095
Con-Pre-Retrain-With	0.9201	0.9196	0.9534	0.9552	0.9616	0.9201	0.9159	0.9561	0.9612	0.9653	0.9196
Con-Pre-Retrain-Without	0.9169	0.9219	0.9603	0.9580	0.9653	0.9109	0.9109	0.9653	0.9635	0.9653	0.9100

Table 12 ../cbow-random/r8

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.5862	0.6824	0.7342	0.7972	0.8003	0.5861	0.6591	0.7184	0.7800	0.7816	0.5731
Predict	0.5862	0.6824	0.7342	0.7972	0.8003	0.5806	0.6595	0.7222	0.7824	0.7853	0.5736
Con-Predict	0.5862	0.6824	0.7342	0.7972	0.8003	0.5806	0.6595	0.7222	0.7824	0.7853	0.5736
Ini-Retrain-With	0.5780	0.6720	0.7305	0.7922	0.7949	0.6103	0.6910	0.7326	0.7957	0.8002	0.6219
Ini-Retrain-Without	0.5858	0.6828	0.7327	0.7990	0.8009	0.5982	0.6761	0.7361	0.7910	0.7917	0.6032
Pre-Retrain-With	0.5773	0.6694	0.7301	0.7938	0.7956	0.6068	0.6869	0.7333	0.7932	0.7974	0.6168
Pre-Retrain-Without	0.5857	0.6834	0.7325	0.7984	0.8007	0.5951	0.6753	0.7339	0.7884	0.7926	0.5995
Con-Pre-Retrain-With	0.5765	0.6735	0.7266	0.7920	0.7956	0.6126	0.6849	0.7334	0.7933	0.7994	0.6165
Con-Pre-Retrain-Without	0.5859	0.6820	0.7333	0.7986	0.8010	0.5947	0.6745	0.7337	0.7883	0.7930	0.6007

Table 13 ../skip-tree/20ng

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.8322	0.8621	0.8949	0.9330	0.8902	0.8197	0.8645	0.8941	0.9299	0.8859	0.8143
Predict	0.8322	0.8621	0.8949	0.9330	0.8902	0.8205	0.8610	0.8941	0.9311	0.8890	0.8181
Con-Predict	0.8322	0.8621	0.8949	0.9330	0.8902	0.8205	0.8610	0.8941	0.9311	0.8890	0.8181
Ini-Retrain-With	0.8329	0.8629	0.8968	0.9307	0.8898	0.8372	0.8668	0.8968	0.9354	0.8902	0.8361
Ini-Retrain-Without	0.8322	0.8625	0.8952	0.9319	0.8902	0.8267	0.8641	0.8906	0.9342	0.8937	0.8294
Pre-Retrain-With	0.8333	0.8606	0.8960	0.9311	0.8886	0.8368	0.8711	0.8960	0.9361	0.8906	0.8392
Pre-Retrain-Without	0.8322	0.8625	0.8945	0.9326	0.8906	0.8252	0.8633	0.8890	0.9338	0.8933	0.8287
Con-Pre-Retrain-With	0.8353	0.8637	0.8941	0.9334	0.8906	0.8361	0.8684	0.8960	0.9357	0.8898	0.8400
Con-Pre-Retrain-Without	0.8322	0.8629	0.8949	0.9326	0.8906	0.8263	0.8653	0.8894	0.9346	0.8929	0.8283

Table 14 ../skip-tree/r52

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.9114	0.9242	0.9557	0.9694	0.9621	0.9004	0.9187	0.9566	0.9712	0.9616	0.8977
Predict	0.9114	0.9242	0.9557	0.9694	0.9621	0.9063	0.9127	0.9534	0.9726	0.9630	0.9009
Con-Predict	0.9114	0.9242	0.9557	0.9694	0.9621	0.9063	0.9127	0.9534	0.9726	0.9630	0.9009
Ini-Retrain-With	0.9100	0.9251	0.9511	0.9717	0.9612	0.9141	0.9260	0.9520	0.9730	0.9607	0.9127
Ini-Retrain-Without	0.9109	0.9251	0.9557	0.9694	0.9625	0.9091	0.9196	0.9520	0.9726	0.9653	0.9159
Pre-Retrain-With	0.9091	0.9182	0.9529	0.9712	0.9612	0.9155	0.9246	0.9529	0.9726	0.9607	0.9173
Pre-Retrain-Without	0.9109	0.9260	0.9557	0.9694	0.9625	0.9100	0.9201	0.9507	0.9726	0.9653	0.9159
Con-Pre-Retrain-With	0.9091	0.9242	0.9552	0.9703	0.9616	0.9150	0.9260	0.9525	0.9730	0.9607	0.9146
Con-Pre-Retrain-Without	0.9109	0.9265	0.9557	0.9694	0.9625	0.9091	0.9191	0.9525	0.9726	0.9653	0.9155

Table 15 ../skip-tree/r8

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.6138	0.5680	0.7147	0.7728	0.7751	0.6400	0.5664	0.7201	0.7714	0.7771	0.6223
Predict	0.6138	0.5680	0.7147	0.7728	0.7751	0.6363	0.5696	0.7256	0.7750	0.7803	0.6210
Con-Predict	0.6138	0.5680	0.7147	0.7728	0.7751	0.6363	0.5696	0.7256	0.7750	0.7803	0.6210
Ini-Retrain-With	0.6245	0.6071	0.7206	0.7774	0.7812	0.6538	0.6205	0.7303	0.7844	0.7864	0.6589
Ini-Retrain-Without	0.6133	0.5676	0.7143	0.7735	0.7750	0.6501	0.5798	0.7345	0.7827	0.7852	0.6435
Pre-Retrain-With	0.6298	0.6200	0.7201	0.7784	0.7804	0.6516	0.6144	0.7286	0.7847	0.7883	0.6574
Pre-Retrain-Without	0.6133	0.5673	0.7145	0.7734	0.7751	0.6473	0.5773	0.7343	0.7817	0.7852	0.6415
Con-Pre-Retrain-With	0.6266	0.6125	0.7212	0.7782	0.7799	0.6498	0.6121	0.7276	0.7824	0.7863	0.6542
Con-Pre-Retrain-Without	0.6133	0.5677	0.7140	0.7735	0.7750	0.6473	0.5797	0.7333	0.7821	0.7852	0.6405

Table 16 ../skip-random/20ng

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.8474	0.8458	0.8867	0.9338	0.9019	0.8283	0.8302	0.8917	0.9287	0.8964	0.8267
Predict	0.8474	0.8458	0.8867	0.9338	0.9019	0.8302	0.8290	0.8921	0.9303	0.8984	0.8318
Con-Predict	0.8474	0.8458	0.8867	0.9338	0.9019	0.8302	0.8290	0.8921	0.9303	0.8984	0.8318
Ini-Retrain-With	0.8501	0.8384	0.8863	0.9319	0.9003	0.8435	0.8583	0.8894	0.9295	0.9011	0.8520
Ini-Retrain-Without	0.8481	0.8458	0.8879	0.9338	0.9019	0.8396	0.8279	0.8871	0.9326	0.8984	0.8400
Pre-Retrain-With	0.8497	0.8431	0.8894	0.9326	0.9015	0.8427	0.8509	0.8890	0.9319	0.9042	0.8509
Pre-Retrain-Without	0.8481	0.8454	0.8882	0.9338	0.9019	0.8376	0.8290	0.8859	0.9326	0.8991	0.8403
Con-Pre-Retrain-With	0.8501	0.8466	0.8890	0.9319	0.9003	0.8419	0.8516	0.8879	0.9319	0.9019	0.8524
Con-Pre-Retrain-Without	0.8481	0.8454	0.8879	0.9338	0.9019	0.8380	0.8287	0.8863	0.9326	0.8991	0.8403

Table 17 ../skip-random/r52

Coefficient	min count 5					min count 15					
	GNB	BNB	KNN	SVM	LOR	GNB	BNB	KNN	SVM	LOR	GNB
Ini-Train	0.9137	0.9237	0.9479	0.9708	0.9685	0.9091	0.9159	0.9561	0.9703	0.9630	0.9073
Predict	0.9137	0.9237	0.9479	0.9708	0.9685	0.9123	0.9077	0.9571	0.9703	0.9639	0.9109
Con-Predict	0.9137	0.9237	0.9479	0.9708	0.9685	0.9123	0.9077	0.9571	0.9703	0.9639	0.9109
Ini-Retrain-With	0.9141	0.9219	0.9479	0.9717	0.9662	0.9159	0.9223	0.9479	0.9712	0.9648	0.9159
Ini-Retrain-Without	0.9132	0.9233	0.9479	0.9708	0.9685	0.9146	0.9169	0.9529	0.9712	0.9667	0.9164
Pre-Retrain-With	0.9132	0.9214	0.9456	0.9717	0.9648	0.9164	0.9201	0.9493	0.9717	0.9648	0.9164
Pre-Retrain-Without	0.9132	0.9228	0.9479	0.9703	0.9680	0.9150	0.9191	0.9529	0.9712	0.9662	0.9169
Con-Pre-Retrain-With	0.9127	0.9242	0.9470	0.9712	0.9657	0.9164	0.9187	0.9493	0.9721	0.9648	0.9164
Con-Pre-Retrain-Without	0.9132	0.9233	0.9475	0.9708	0.9685	0.9150	0.9182	0.9529	0.9712	0.9662	0.9169

Table 18 ../skip-random/r8

by this idea, we proposed a two-step embedding approach that firstly embed high-frequency words and then predict embeddings of low-frequency words from the high-frequency words. Our experiments on word similarity and text classification demonstrated the effectiveness of the proposed approach.

Acknowledgement

Author details
[1]Center for Speech and Language Technology, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [2]Center for Speech and Language Technologies, Division of Technical Innovation and Development, Tsinghua National Laboratory for Information Science and Technology, ROOM 1-303, BLDG FIT, 100084 Beijing, China.

References