Text classification with sparse composite document vectors

Dheeraj Mekala*  Vivek Gupta*  Harish Karnick
IIT Kanpur  Microsoft Research  IIT Kanpur
dheerajm@iitk.ac.in  t-vigu@microsoft.com  hk@iitk.ac.in

Abstract

We present a modified feature formation technique - sparse document vector (SDV) derived from - graded-weighted Bag of Word Vectors (BoWV) by (Vivek Gupta, 2016) for faster and better composite document feature representation. We propose a very simple feature construction algorithm that potentially overcomes many weaknesses in current distributional vector representations and other composite document representation methods widely used for text representation. Through extensive experiments on multi-class classification on 20newsgroup dataset and multi-label text classification on Reuters-21578, we achieve better performance results and also significant reduction in training and prediction time compared to composite document representation methods BoWV (Vivek Gupta, 2016) and TWE (Liu et al., 2015b). We also significantly perform better than current state of art method NTSG (Liu et al., 2015a) on standard performance metric for 20NewsGroup.

1 Introduction

Text classification and text clustering are widely used in various information retrieval and natural language processing tasks (Moulinier et al., 1996). Various machine learning algorithms are used to perform these tasks. They require text data to be represented as a fixed dimension $D$ floating point vector. Some of the most common techniques used are described in the next section along with their problems.

2 Related Work

Le. and Mikolov et. al. (Le and Mikolov, 2014) proposed two models for Distributional representation of document as fixed dimensional vectors called Distributed Memory Model Paragraph Vectors (PV-DM) (Le and Mikolov, 2014) and Distributed BoWs paragraph vectors (PV-DBoW) (Le and Mikolov, 2014). However, they do not perform very well in document categorization due to multiple reasons as stated in (Vivek Gupta, 2016).

2.1 Composite Document Representation

Recently much efforts has gone into composing a document vector from word vectors. Some of the important methods are described below:

2.1.1 Weighted Average Embedding

Mukerjee et al. (Pranjal Singh, 2015) proposed a weighted average composite document vectors in which vectors for words appearing in the document are added after weighting them with idf (Robertson, 2004) values from the training set. This method tries to capture the relative importance of words by varying weights.

This method assumes that all words within a document have the same semantic topic. Intuitively a paragraph often has words coming from multiple semantically different topics.

2.1.2 Topical Word Embedding

Liu et al. (Liu et al., 2015b) proposed three novel composite document representations called Topical word embedding (TWE-1, TWE-2 and TWE-3). For each word in the vocabulary, word-topic assignments are obtained through Latent Dirichlet Allocation (Blei et al., 2003). TWE-1 learns word and topic embedding by considering each topic as pseudo word and builds the topical word embedding for each word-topic assignment. TWE-2 learns topical word embedding of each word-topic assign-
Figure 1: Skip-Gram and TWE models. Blue circles indicate word embeddings and green circles indicate topic embeddings. Since TWE-2 does not reserve stand-alone word / topic embeddings, we simply represent topical word embeddings in TWE-2 using blue circles.

ment directly by considering each word-topic pair as a pseudo word. For each word and each topic, TWE-3 builds distinct embeddings for the topic and word separately and for each word-topic assignment, the corresponding word embedding and topic embedding are concatenated to form a topical word embedding which is used for further learning. Architecture for same is shown in Figure 1. TWE-1 outperforms other embedding methods on 20 newsgroup data-set.

(Liu et al., 2015a) proposed an architecture called Neural tensor skip-gram model (NTSG(SOA)-1, NTSG(SOA)-2, NTSG(SOA)-3, NTSG(SOA)-4), to learn multi-prototype word embeddings and uses a tensor layer to model the interaction of words and topics. NTSG(SOA) outperforms other embedding methods like TWE-1 on 20 newsgroup data-set.

In TWE-1, the interaction between a word and the topic to which it is assigned is not considered. In TWE-2, the interaction between a word and its assigned topic is considered. As each word is differentiated into different topics, there are sparsity issues. TWE-3 stands between both TWE-1 and TWE-2 with respect to differentiation and sparsity. In TWE-3, the word embeddings are influenced by the corresponding topic embeddings, making words in the same topic less discriminative. Since, TWE uses LDA it suffers from computational issues like large training and prediction time and requires more storage.

2.1.3 Graded weighted Bag of Word Vectors

(Vivek Gupta, 2016) proposed a method to form a composite document vector using word vectors and tf-idf values called Graded Weighted Bag of Words Vector (BoWV). In BoWV, each document is represented by a vector of dimension $D = K \times d + K$, where $K$ represents number of clusters, $d$ is the dimension of the word vectors. The basic idea is, semantically different words belong to different clusters and their word vectors should not be averaged, hence concatenated. Further, BoWV concatenates inverse cluster frequency of each cluster (icf) which is computed by idf values of all words in a cluster to capture the importance of words across documents. (Vivek Gupta, 2016) shows BoWV outperforms paragraph vector models on a hierarchical product classification task.

BoWV is a non-sparse high dimensional continuous vector and thus suffers computational issues like large training and prediction time and takes considerable storage. This makes them impractical for many large datasets.

3 Sparse Document Vectors

In this section we would present the main contributions of the paper. We modify BoWV to incorporate sparsity, leading to faster computation time and better results. We also modify the weighting scheme by directly multiplying idf values with word vectors leading to off-store computation and significant improvement in time and space complexity. We call our modified document representation as Sparse Document Vector (sdv). Compared to BoWV (Vivek Gupta, 2016), sdv has the following changes and major advantages:

1. We replace hard clustering (K-means) with soft clustering algorithm (GMM) so that each word can belong to multiple topics, thus handling polysemy. Lots of work had been done earlier to handle polysemy for word embeddings (Huang et al., 2012) but not on its effects in document representation.

2. We introduce sparsity in the feature vector by zeroing attribute values that are $< 5\%$ of average attribute values, which leads to reduction in feature vector size, training and prediction time (Linear SVM) by significant factors as shown in Figure 2.

3. Instead of separately concatenating cluster frequency as in BoWV, we weight each word vector of a word by its inverse document frequency while forming cluster vectors. This...
leads to significant reduction in feature formation time due to off-loop pre-computation.

The full details are in Algorithm 1.

Algorithm 1: Sparse Document Vector

Data: Documents $D_n, n = 1 \ldots N$
Result: Document vectors $sdv_{D_n}, n = 1 \ldots N$
1 Obtain word vector $(w_i)$, for each word $w_i$;
2 Calculate idf values, $idf(w_i), i = 1..|V|$
3 Cluster word vectors $\vec{w}$ using GMM clustering into K clusters;
4 Obtain soft assignment $P(c_k|w_i)$ for word $w_i$ and cluster $c_k$:
5 for each word $w_i$ in vocabulary $V$ do
6   for each cluster $c_k$ do
7       $w\vec{c}_{ik} = \vec{w} \times P(c_k|w_i)$;
8   end
9   $w\vec{v}_i = idf(w_i) \bigoplus_{k=1}^{K} w\vec{c}_{ik}$ ;
10  end
11 for $n \in (1..N)$ do
12  Initialize document vector $dv_{D_n} = \vec{0}$;
13  for word $w_i$ in $D_n$ do
14    $dv_{D_n} += w\vec{v}_i$;
15  end
16  $sdv_{D_n} = make\-sparse(dv_{D_n})$;
17 end

Word vectors are clustered using soft clustering algorithms (e.g. GMM), thus each word belongs to every cluster with some probability. For each word $w_i$, we create $K$ word-cluster-vectors ($w\vec{c}_{ik}$) by multiplying word vector with the probability of word belonging to each cluster. For each word $w_i$, we concatenate all word-cluster-vectors ($w\vec{c}_{ik}$) and weight it with idf of $w_i$ to form a word-topics vector ($w\vec{v}_i$).

We use idf values from training corpus directly for the test corpus for weighting. Finally, for each word $w_i$ appearing in the document $D_n$, we sum word-topics vector $w\vec{v}_i$ to obtain the document vector $dv_{D_n}$. We make document vector $dv_{D_n}$ sparse as explained in point 2 above resulting in $sdv_{D_n}$.

During implementation, for each word $w_i$ in the vocabulary $V$, we can pre-compute the word top-
ics vector $w\vec{v}_i$ which results in significant reduction in computation time as described in Algorithm 1 (line 5-10).

4 Experiments

We perform multiple experiments to show the effectiveness of our representation for multi-class and multi-label text classification. We run the multi-class experiments on 20NewsGroup dataset (Lang.) and multi-label classification experiments on Reuters-21578 dataset (Lewis.).

Experimental conditions are as follows: pre-processing: remove stop words; word vector dimension: 200; GMM components: 60 with same spherical co-variance matrix for all components; classifier multi-class: logistic regression, OneVsRest setting; platform: Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz, 40 working cores, 128GB RAM with Linux Ubuntu 14.4, use multiple cores for one-vs-rest classifier, Word2vec, and Doc2Vec training in all other cases only a single core.

4.0.4 Baseline

We consider the following baselines, bag-of-words (BOW) model, Paragraph vector models (Mikolov), Topical word embeddings (TWE) (Liu et al., 2015b). Average word-vector model, where we build document vector by averaging all word vectors in a document and tf-idf weighted average word-vector model, where we build document vector by tf-idf weighted averaging all word vectors in a document. The dimension of word vectors in both Average word-vector and tf-idf weighted word-vector model is 200. The dimension of document vector in paragraph vector models is 400. We use the source code provided by (Liu et al., 2015b) with same experimental parameter settings for TWE models, the number of topics are set to 80 and the dimension of word and
topic embeddings is 400. We use gensim (Rehůřek and Sojka, 2010) for implementing Doc2Vec and Word2Vec. For all baselines, we use LinearSVM for multi-class classification and Logistic regression in OneVsRest approach for multi-label classification. During training we tune all parameters for baselines and the performance reported is with the best parameters. We used default hyperparameters i.e. negative sampling = 10 an number of epoch = 25 for training word vectors. Code is available here 1.

4.1 Dataset Description

**Multi Class**: The training:test samples breakup on 20NewsGroup dataset is 11314 : 7532.

**Multi Label**: The training:test samples breakup on Reuters-21578 dataset is 13734 : 5887. We utilize the script provided by Eustache 2 for preprocessing the Reuters-21578 dataset. Dataset contains 5 main categories and 445 categories in total. Each article is assigned to several categories, thus a multi-label classification.

5 Results

5.1 Multi-class classification

We evaluate performance using standard metrics like accuracy, macro-averaging precision, recall and F-measure for comparison. Table 1 shows results with popular and current state-of-art (NTSG) document representation on the 20News-group dataset.

| Model       | Acc  | Prec | Rec  | F-mes |
|-------------|------|------|------|-------|
| SDV(SGNS)   | 84.6 | 84.1 | 83.7 | 83.8  |
| BoWV(SGNS)  | 83.8 | 83.4 | 82.9 | 82.9  |
| SDV(CBoW)   | 82.6 | 82.1 | 81.8 | 81.8  |
| BoWV(CBoW)  | 81.6 | 81.1 | 81.1 | 80.9  |
| NTSG-1      | 82.6 | 82.5 | 81.9 | 81.2  |
| NTSG-2      | 82.5 | 83.7 | 82.8 | 82.4  |
| NTSG-3      | 81.9 | 83.0 | 81.7 | 81.1  |
| TWE-1       | 81.5 | 81.2 | 80.6 | 80.6  |
| BOW         | 79.7 | 79.5 | 79.0 | 79.0  |
| PV-DBOW     | 75.4 | 74.9 | 74.3 | 74.3  |
| PV-DM       | 72.4 | 72.1 | 71.5 | 71.5  |
| Ave-Vec     | 71.8 | 71.2 | 70.5 | 70.0  |

Table 1: Performance on multi-class classification

Compared to TWE, SDV manages to reduce training and prediction times significantly. Table 2 shows a comparison of training and prediction times between BoWV, SDV and TWE models. Table 3 shows the space comparison between BoWV and SDV models. The pictographic comparison between training and prediction times are shown in Figure 3 and 4.

We observe that SDV outperforms paragraph vector, average word-vector, TWE, NTSG tf-idf weighted average word-vector, bag-of-words models by reasonable margins. As topic model learning and document vector formation in NTSG(SOA) is similar to TWE, SDV is faster than NTSG(SOA). Compared to BoWV, there is significant reduction in training and prediction times and also in document feature space (by 80%).

| Time (sec) | BoWV | TWE | SDV |
|------------|------|-----|-----|
| Clusters formation | 90   | 660 | 90  |
| DocVec formation | 1170 | 180 | 60  |
| Total training | 1320 | 858 | 210 |
| Total prediction | 780  | 120 | 42  |

Table 2: Time Comparison

| Space Occupied | BoWV | SDV |
|----------------|------|-----|
| Document Vectors | 1.1 GB | 236 MB |

Table 3: Space Complexity

5.2 Multi-label classification

We evaluate performance using Precision@K, nDCG@k, Coverage error, Label ranking average precision score (LRAPS), weighted F-measure. The first two are taken from Extreme learning repository (Bhatia et al., 2011) and the next four are defined in Scikit-Learn multilabel loss function class (Pedregosa et al., 2011).
As we split data randomly into train and test sets, weighted F-measure is an appropriate metric for multi-label classification as it considers label sample biases as well. Table 4 and 5 shows the evaluation results on multi-label text classification on Reuters-21578 dataset.

| Model        | Prec@1 | Prec@5 | nDCG@1 | nDCG@5 |
|--------------|--------|--------|--------|--------|
| SDV(SGNS)    | 94.20  | 36.98  | 49.55  |        |
| BoWV(SGNS)   | 93.76  | 36.70  | 49.23  |        |
| SDV(CBoW)    | 93.00  | 36.40  | 48.90  |        |
| BoWV(CBoW)   | 92.90  | 36.14  | 48.55  |        |
| TWE-1        | 90.91  | 35.49  | 47.54  |        |
| PV-DM        | 87.54  | 33.24  | 44.21  |        |
| PV-DBOW      | 88.78  | 34.51  | 46.42  |        |
| AvgVec       | 89.09  | 34.73  | 46.48  |        |
| tfidf AvgVec | 89.33  | 35.04  | 46.83  |        |

Table 4: Performance on various metrics I

| Model        | Coverage Error | LRAPS | Weighted F1-Score |
|--------------|----------------|-------|------------------|
| SDV(SGNS)    | 6.48           | 93.30 | 81.75            |
| BoWV(SGNS)   | 7.56           | 92.70 | 81.07            |
| SDV(CBoW)    | 7.49           | 92.00 | 80.15            |
| BoWV(CBoW)   | 8.16           | 91.46 | 79.16            |
| TWE-1        | 9.03           | 89.25 | 74.76            |
| PV-DM        | 13.15          | 86.21 | 70.24            |
| PV-DBOW      | 11.28          | 87.43 | 73.68            |
| AvgVec       | 9.67           | 87.28 | 71.91            |
| tfidf AvgVec | 9.42           | 87.90 | 71.97            |

Table 5: Performance on various metrics II

6 Time Complexity Analysis

Let, $W = \text{size of vocabulary}$, $N = \text{number of documents}$, $T = \text{number of topics or clusters}$, $C = \text{window size}$, $M = \text{corpus length}$, $K_W = \text{word vector length}$, $K_T = \text{topic vector length}$, $I = \text{number of iterations in topic modeling}$, $D = \text{feature vector dimension}$.

LDA time complexity: $O(W^2NT)$
GMM time complexity: $O((NT^2D))$.
Since $W^2 >> TD$, thus better time complexity. Detailed time complexity comparison is shown in Table 6.

| Model  | Parameters | Comp. Complexity |
|--------|------------|------------------|
| SGNS   | WK         | CM(1 + logW)     |
| TWE-1  | (W + T)K   | IM + 2CM(1 + logW) |
| TWE-2  | WT K       | IM + CM(1 + logWT) |
| TWE-3  | WK_W + TK_T| IM + CM(1 + logWT) |
| SVD    | WK         | NK^2D + CM(1 + logW) |

Table 6: Complexity Analysis

7 Conclusion

In this paper we modified BoWV and reduced overall feature vector computation time by $8.5\times$, prediction time by $20\times$ and space by $4\times$ on 20newsgroup multi-class dataset. We experimentally demonstrated that in multi-class classification on 20newsgroup dataset, our model outperforms TWE-1 and NTSG(SOA) on standard performance metrics by significant margins. Compared to TWE-1 and NTSG(SOA), our model is faster by $6\times$ during feature vector computation and by $3\times$ during test class prediction on 20newsgroup. In multi-label classification on Reuters, our model outperforms on every metric by reasonable margins. Overall, we improved BoWV by making it simple and efficient and obtained better results on standard multi-class and multi-label datasets.

We presented a SOA method for document classification with composite document vectors.

References

Bhatia et al.] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. In Advances in Neural Information Processing Systems 28.

[Blei et al.2003] David M. Blei, Andrew Y. Ng, and
Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

[Huang et al.2012] Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 873–882. Association for Computational Linguistics.

[Lang] K. Lang. 20 newsgroups data set.

[Le and Mikolov2014] Quoc V Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*.

[Lewis] David D. Lewis. Reuters-21578.

[Liu et al.2015a] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2015a. Learning context-sensitive word embeddings with neural tensor skip-gram model. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[Liu et al.2015b] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015b. Topical word embeddings.

[Moulinier et al.1996] Isabelle Moulinier, Gailius Raskinis, and J Ganascia. 1996. Text categorization: a symbolic approach. In *proceedings of the fifth annual symposium on document analysis and information retrieval*, pages 87–99.

[Pedregosa et al.2011] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Pranjal Singh2015] Amitabha Mukerjee Pranjal Singh. 2015. Words are not equal: Graded weighting model for building composite document vectors. In *Proceedings of the twelfth International Conference on Natural Language Processing (ICON-2015)*. BSP Books Pvt. Ltd.

[ˇRehůrek and Sojka2010] Radim ˇRehůrek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50. ELRA, May.

[Robertson2004] Stephen Robertson. 2004. Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation*, 60(5):503–520.

[Vivek Gupta2016] Ashendra Bansal Pradhuman Jhala Vivek Gupta, Harish Karnick. 2016. Product classification in e-commerce using distributional semantics. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 