Derivation of Document Vectors from Adaptation of LSTM Language Model

Wei Li and **Brian Mak**
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
{wliax, mak}@cse.ust.hk

Abstract

In many natural language processing tasks, a document is commonly modeled as a bag of words using the term frequency-inverse document frequency (TF-IDF) vector. One major shortcoming of the TF-IDF feature vector is that it ignores word orders that carry syntactic and semantic relationships among the words in a document. This paper proposes a novel distributed vector representation of a document called DV-LSTM. It is derived from the result of adapting a long short-term memory recurrent neural network language model by the document. DV-LSTM is expected to capture some high-level sequential information in a document, which other current document representations fail to do. It was evaluated in document genre classification in the Brown Corpus , the BNC Baby Corpus, and the Penn Treebank Dataset. The results show that DV-LSTM significantly outperforms TF-IDF vector and paragraph vector (PV-DM) in most cases, and their combinations may further improve classification performance.

1 Introduction

In many classification tasks in the area of natural language processing (NLP), it is necessary to transform text documents of variable lengths into vectors of a fixed length so that they can be classified or compared as most classifiers only work on inputs of a fixed length. Perhaps the most popular document vectors is the *term frequency-inverse document frequency* (TF-IDF) feature vector (Robertson and Jones, 1976). Term-frequency-based document vectorization makes two assumptions (Cachopo, 2007; Le and Mikolov, 2014): (a) occurrences of each term are mutually independent, and (b) a document is treated as a "bag of words" and different permutations of the same set of words are considered to be same. These assumptions suffers from a major drawback that it ignores word orders and other sequential information in a document which can be important in some NLP tasks such as genre classification. For example, 'Wall' and 'Street' in the named entity 'Wall Street' are treated as independent words in a TF-IDF vector. Using an n-gram TF-IDF vector may alleviate the problem to some extent, but it is still hard to capture long-distance or high-level abstract sequential patterns. Moreover, (n-gram) TF-IDF vectors cannot capture syntactic or semantic relationship/similarity between words, paragraphs, and documents. Another notable document vectorization is the *paragraph vector* that learns from a distributed memory model (PV-DM), which is a succinct distributed representation of sentences or paragraphs (Le and Mikolov, 2014; Dai et al., 2015; Ai et al., 2016). PV-DM has been shown to perform significantly better than the bag-of-words model in many NLP tasks. Moreover, skip-thought vectors (Kiros et al., 2015) that are derived from recurrent encoder-decoder models also show superior performance against the bag-of-words model.

In this paper, we propose a novel document vectorization method which adapts[1] a long short-term memory recurrent neural network (RNN) language model (LSTM-LM) (Sundermeyer et al., 2012) with a document, and then vectorize the

[1] One may also treat our adaptation method as re-training the initial LSTM-LM with the adapting document.

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 456–461,
Valencia, Spain, April 3-7, 2017.

adapted model parameters to obtain its document vector, labeled as DV-LSTM. Since the recurrent nature of LSTM-LM should capture some high-level and abstract sequential information from its training documents, if the LM adaptation is effective, each adapted LM will contain distinctive sequential information of the adapting document, and the adapted parameters may be used to represent the adapting document distinctively. Our DV-LSTM is similar to the TF-IDF vector and PV-DM in that they all can be derived in an unsupervised manner. Compared with the TF-IDF vector, DV-LSTM is more expressive as it makes use of continuous word embedding and sequential information in a document. Compared with PV-DM, DV-LSTM does not suffer from the limitation due to a sliding context window on the inputs.

2 LSTM Language Modeling

RNN language model (LM) — especially the long short-term memory language model (LSTM-LM) — is the state-of-the-art language models (Mikolov et al., 2010; Mikolov et al., 2011; Bengio et al., 2006). LSTM-LM is chosen to develop our document vectorization for three reasons. Firstly, it can capture comparatively more distant patterns in a document that are not limited by the size of the input context window. Thus, the model parameters of an LSTM-LM can encapsulate the different grammars and styles in its training documents. Secondly, the hidden layer(s) of an LSTM provide a distributed representation of the input words in a continuous space so that the semantic and syntactic relationship among words can be captured. Finally, by controlling the size of the hidden layer(s) and the model parameters to adapt, one may effectively adjust the number of model parameters to adapt according to the size of the adapting document to ensure that the final document vector is derived robustly.

Figure 1 shows the LSTM-LM network for training our document vectors. The input to the model is the current word $\mathbf{w}_t$ represented by its one-hot encoding, which is projected to a distributed representation by a linear identity compression layer and then by a non-linear sigmoid layer. The identity compression layer also helps make the model more compact so as to improve training speed. Let $\mathbf{s}_t$ be the hidden state for the input word $\mathbf{w}_t$. The model is trained to give two kinds of outputs to the word class layer ($\mathbf{v}_t$) as well as to the output word layer ($\mathbf{w}_{t+1}$) (Mikolov et al., 2011). That is, it produces the posterior probability $P(\mathbf{v}_t|\mathbf{s}_t)$ of the word class $\mathbf{v}_t$ given the current state $\mathbf{s}_t$, and the posterior probability $P(\mathbf{w}_{t+1}|\mathbf{v}_t, \mathbf{s}_t)$ of the next word $\mathbf{w}_{t+1}$ given the current word class and LSTM state.

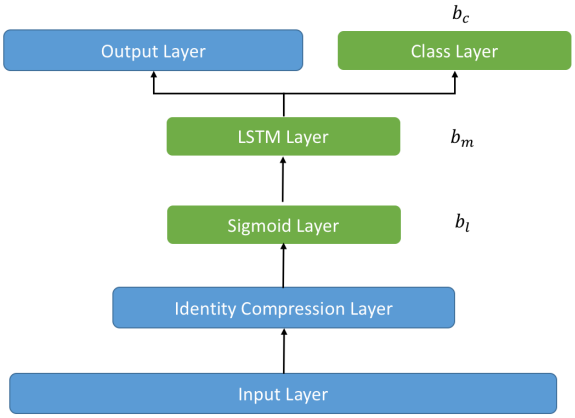

Figure 1: *The LSTM network chosen to derive our document vectors. (The recurrency of LSTM cells is not shown)*

3 Document Vectorization by LSTM-LM Adaptation

We propose to derive document vectors (DVs) from a well-trained parent LSTM language model by adaptation using the following procedure:

STEP 1: Train a parent LM using all the documents in a training corpus.

STEP 2: Adapt the parent LM with each document in the training corpus.

STEP 3: Extract model parameters of interest from the adapted LM, and vectorize them to produce DV-LSTM for the adapting document.

3.1 Derivation of DV-LSTM

In our experiments, the LSTM neural network of Figure 1 has 200 units in the identity compression layer, 100 units in the sigmoid compression layer, 100 LSTM units, 500 word classes and V output units (where V is the vocabulary size). In the derivation of DV-LSTM, only the biases in the sigmoid layer $\mathbf{b}_l \in \mathbb{R}^{100}$, LSTM layer $\mathbf{b}_m \in \mathbb{R}^{400}$, and word-class layer $\mathbf{b}_c \in \mathbb{R}^{500}$ are adapted. The LSTM bias vector $\mathbf{b}_m$ is further comprised of four 100-dimensional bias sub-vectors: input-gate biases $\mathbf{b}_{m_i}$, forget-gate biases $\mathbf{b}_{m_f}$, output-gate biases $\mathbf{b}_{m_o}$, and cell biases $\mathbf{b}_{m_c}$.

The 3 different biases are supposed to capture different and complementary information in a document: $\mathbf{b}_l$ is to capture the abstract and distributed word embeddings; $\mathbf{b}_m$ is to capture the long-span sequential text information in a document; $\mathbf{b}_c$ is to capture the word class statistics. The 3 biases are concatenated to the final 1000-dimensional DV-LSTM document vector as follows:

$$\text{DV-LSTM} = [n(\mathbf{b}'_{ml}), n(\mathbf{b}'_c)]' \,, \qquad (1)$$

where $\mathbf{b}_{ml}$ is given by

$$[n(\mathbf{b}'_{m_i}), n(\mathbf{b}'_{m_f}), n(\mathbf{b}'_{m_o}), n(\mathbf{b}'_{m_c}), n(\mathbf{b}'_l)]' \,. \quad (2)$$

In Eq.(1) and Eq.(2), $n(\cdot)$ is the normalization operator which normalizes a vector to the unit norm.

According to some previous researches in genre classification, it is found that models fitted on some lower-level features (e.g., term-frequency related feature, which is highly correlated to the topic and language) may actually hurt genre classification when they are tested on new documents of the same genre but of different topic or language (Petrenz and Webber, 2011; Petrenz, 2009; Petrenz, 2012).

In our model, $\mathbf{b}_m$ is a high-level abstract feature, which is relatively independent of the topic or language specific term-frequency distribution. $\mathbf{b}_c$ is a lower-level feature that is related to the word clusters. Comparing with n-gram term-frequency features whose good performance depend on a strong topic-genre correlation, $\mathbf{b}_c$ is a relatively moderate lower-level feature. We believe that by combining high-level abstract features and lower-level features, our model may perform better in situations where the term-frequency based pattern is not entirely reliable for classification. Such is the case in the genre classification tasks of this paper, where term-frequency distribution can be confused by different topic-genre correlation.

4 Experimental Evaluation: Text Genre Classification

The proposed document vector DV-LSTM was evaluated on the genre classification of documents in three corpora:

- *Brown Corpus* (Brown) (Francis and Kucera, 1979): It consists of 500 documents with a total of about 1 million words distributed across 15 genres organized hierarchically in three levels. The sub-genres under the *fiction* genre were merged (Wu et al., 2010) so that the total number of genres was reduced to 10.

- *BNC Baby Corpus* (BNCB) (Burnard, 2003): It is a subset of BNC, consisting of 182 documents written in 4 genres: *fiction, newspapers, academic* and *conversation*. Each genre consists of a total of about 1 million words.

- *Penn Treebank Dataset* (PTB): It was artificially extracted from the Penn Treebank Corpus by taking out the documents that have genre tags provided by (Webber, 2009; Plank, 2009). It has 5 genres: *essays, highlights, letters, errata* and *news*. The *errata* genre was removed as there are very few documents of that genre. We also removed short documents with fewer than 200 words from the dataset. At the end, the dataset has a total of 239 documents in 4 genres: 38 *highlights*, 95 *essays*, 42 *letters*, and 64 *news*.

4.1 Text pre-processing and SVM training

The Natural Language Toolkit (NLTK) (Loper and Bird, 2002) was used for tokenization, and the WordNet Lemmatizer (Miller, 1994) was used for text pre-processing. The letters in the documents were also converted to lower cases to improve the TF-IDF baseline performance, and the word classes were determined by Brown clustering (Brown et al., 1992). During the unsupervised training of PV-DMs and DV-LSTMs, documents in a dataset were shuffled to eliminate the possibility that a classifier may simply use the position of documents for genre classification. All data were mean-zeroed before inputting to the classifier.

For each type or combination of document feature vectors, a linear SVM classifier was built from the training dataset using LinearSVC from the scikit-learn toolkit[2]. To improve the reliability of experimental results, documents in each corpus were shuffled ten times, and for each shuffled dataset, a 10-fold cross-validation was conducted. Our DV-LSTM was tested against the TF-IDF feature and the state-of-the-art paragraph vector PV-DM. Results are reported in terms of classification accuracies that are averages from classifications over 10×10-fold cross validations.

[2]Empirically, we did not get better results using nonlinear kernels such as the RBF kernel.

4.2 Training of document vector DV-LSTM

The RWTH Aachen University Neural Network Language Modeling Toolkit (RWTHLM) (Sundermeyer et al., 2015; Sundermeyer et al., 2014) was used for training all LSTM-LMs and adapting them to produce the DV-LSTMs. The length of historical context is the concatenation of the default sentence segmentations in the original corpus up to 500 characters. The parent model was trained with a maximum of 10 epochs, while LM adaptation took at most 15 epochs. The initial learning rates were set to 0.02. The sub-vectors in $\mathbf{b}_m$ were whitened first (mean-zeroed and scaling to the unit variance for each axis) before concatenation.

Table 1: *Values of various hyperparameters being tuned for the derivation of the best PV-DM.*

context window size	{5, 10, 15, 20}
min. word frequency	{0, 5, 10, 20}
negative word samples	{0, 10 , 20}
downsampling threshold	{0, 5E-5}

4.3 Training of paragraph vector PV-DM

A PV-DM was trained for each document in a corpus using the Gensim toolkit (Řehůřek and Sojka, 2010). They were trained for 20 epochs with an initial learning rate of 0.025. PV-DMs with dimensions of 100, 500 and 2000 were investigated, and it was found that PV-DMs of 500 dimensions provide consistently good performance; they are denoted as PV_{500}. The optimal hyperparameters for PV-DM derivation were grid-searched for each task using 1/10 of its corpus data. The hyperparameters and their values tried in the grid search are summarized in Table 1.

Most hyperparameters in Table 1 are also shared by the training of DV-LSTM. However, due to the limitation of the current experiment platform and the cost of grid searches, we do not tune these hyperparameters in training DV-LSTM. Hence the corresponding hyperparameters in DV-LSTM are all set to 0 unless stated explicitly. Thus DV-LSTM is expected to have a disadvantage in the tuning of hyperparameters.

4.4 Summary

Table 2 summarizes the dimension of various feature vectors used in the experiments, where $\mathbf{z}_5^{1000}$ and $\mathbf{z}_5$ represent the TF-IDF feature vectors using the top 1,000 and 10,000 5-grams respectively.

Table 2: *The dimension of various feature vectors.*

Feature	Dimension
PV_{500}	500
$\mathbf{z}_5^{1000}$	1,000
DV-LSTM	1,000
$\mathbf{z}_5$	10,000

4.5 Experimental Results

The genre classification accuracy and the weighted F-score results using different feature vectors over the three corpora are summarized in Table 3 and Table 4.

Table 3: *Genre classification accuracy (%).*

Features	PTB	Brown	BNCB
4-char-gram*	-	64.40	-
5-gram $\mathbf{z}_5$	80.91	65.24	96.27
PV_{500}	81.63	**65.68**	98.35
DV-LSTM-b_m	75.93	60.14	98.50
DV-LSTM-b_c	82.63	63.88	99.45
DV-LSTM	**84.70**	65.20	**100.00**
DV-LSTM-PV_{500}	86.00	**67.00**	**100.00**
DV-LSTM-$\mathbf{z}_5^{1000}$	**86.38**	66.84	**100.00**

Table 4: *Genre classification F-score.*

Features	PTB	Brown	BNCB
1-gram*	-	-	0.913
5-gram*	-	-	0.956
5-POS*	-	-	0.947
5-gram $\mathbf{z}_5$	0.7996	0.6275	0.9623
PV_{500}	0.8154	**0.6455**	0.9820
DV-LSTM-$\mathbf{b}_m$	0.7559	0.5959	0.9841
DV-LSTM-$\mathbf{b}_c$	0.8239	0.6326	0.9941
DV-LSTM	**0.8434**	0.6443	**1.0000**
DV-LSTM-PV_{500}	0.8576	0.6613	**1.0000**
DV-LSTM-$\mathbf{z}_5^{1000}$	**0.8607**	**0.6614**	**1.0000**

Besides individual features, we also investigated the contribution of each bias vector in DV-LSTM and the possibility of feature combinations. The bold results represent the best performance for each task given by a single feature or a set of combined features. Results labeled with * are baseline

results quoted from (Tang and Cao, 2015; Wu et al., 2010).

We have the following observations:

- For both the Brown Corpus and BNCB Corpus, results from our own 5-gram TF-IDF are better than the quoted baselines.
- In general, our DV-LSTM performs better than PV-DM, and PV-DM performs better than the 5-gram TF-IDF. All the bold results are statistically significantly better than the 5-gram TF-IDF results based on the paired sample t-test (Dietterich, 1998) at the 99% confidence level.
- Among the single features, the proposed DV-LSTM performs the best in both PTB and BNCB tasks, and gives comparable performance as PV_{500} in the Brown Corpus.

  One possible reason is that the hyperparameters for training DV-LSTM were not as fine-tuned as those for PV_{500}, giving DV-LSTM a disadvantage. Another plausible reason is that PTB's genres are almost unrelated to the topics and it likely requires more abstract sequential information for their classification. On the other hand, the Brown Corpus has a relatively strong overlapping between topics and genres. Thus, features such as TF-IDF or PV-DM that have good estimates of the term frequencies of topic related words/phrases could perform better.
- Both PV_{500} and our DV-LSTM show superior performance comparing to the traditional n-gram TF-IDF. This is probably attributed to the neural network's capability of learning abstract patterns. Moreover, the paragraph vector and our DV-LSTM are dense representations of documents. They have more utility than the sparse TF-IDF vector, especially when comparing the semantic and syntactic similarity of documents.
- Between the two bias components of our DV-LSTM, it is interesting to see that the LSTM bias vector $\mathbf{b}_m$ (and its results are labeled with DV-LSTM-$\mathbf{b}_m$ in Tables 3 and 4) is outperformed by the class bias vector $\mathbf{b}_c$ (and its results are labeled with DV-LSTM-$\mathbf{b}_c$ in Tables 3 and 4). Nevertheless, it seems that they are complementary to each other, and their combination in DV-LSTM further improves the classification performance.

5 Conclusions and Future Works

This paper proposes a novel distributed representation of a document, which we call "document vector" (DV). Currently, we estimate the DV by adapting the various bias vectors and the word class bias of an LSTM-LM network trained from the corpus of a task. We believe that these parameters capture some word ordering information in a larger context that may supplement the standard frequency-based TF-IDF feature or the paragraph vector PV-DM in solving many NLP tasks. Here, we only confirm its effectiveness in document genre classification. In the future, we would like to investigate the effectiveness of our DV-LSTM in other NLP problems such as topic classification and sentiment detection. Moreover, we would also like to investigate the utility of this model (or its variants) in the cross-lingual problems, as high-level sequential pattern captured by the (deep) hidden layers is expected to be relatively language independent.

6 Acknowledgements

The work described in this paper was supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. HKUST616513, HKUST16206714 and HKUST16215816).

References

Qingyao Ai, Liu Yang, Jiafeng Guo, and W. Bruce Croft. 2016. Analysis of the paragraph vector model for information retrieval. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval*, pages 133–142. ACM.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, T. J. Watson, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–480.

Lou Burnard. 2003. Reference guide for BNC Baby.

Ana Margarida de Jesus Cardoso Cachopo. 2007. *Improving methods for single-label text categorization*. Ph.D. thesis, Universidade Técnica de Lisboa.

Andrew M. Dai, Christopher Olah, and Quoc V. Le. 2015. Document embedding with paragraph vectors. *arXiv preprint arXiv:1507.07998*.

Thomas G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923.

W. Nelson Francis and Henry Kucera. 1979. Brown corpus manual. *Brown University*, 15.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3294–3302. Curran Associates, Inc.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of The 31st International Conference on Machine Learning*, volume 14, pages 1188–1196.

Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of Interspeech*, pages 1045–1048.

Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5528–5531. IEEE.

George A. Miller. 1994. Wordnet: A lexical database for english. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, page 468.

Philipp Petrenz and Bonnie Webber. 2011. Squibs: Stable classification of text genres. *Computational Linguistics*, 37(2):385–394.

Philipp Petrenz. 2009. Assessing approaches to genre classification. Master's thesis, School of Informatics, University of Edinburgh.

Philipp Petrenz. 2012. Cross-lingual genre classification. In *Proceedings of the Student Research Workshop at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 11–21, Avignon, France, April. Association for Computational Linguistics.

Barbara Plank. 2009. PTB/PDTB files belonging to different genres. `http://www.let.rug.nl/~bplank/metadata/genre_files_updated.html`.

Radim Řehůřek 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, Valletta, Malta, May. ELRA. `http://is.muni.cz/publication/884893/en`.

Stephen E. Robertson and K. Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Proceedings of Interspeech*, pages 194–197.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2014. RWTHLM – the RWTH Aachen University neural network language modeling toolkit. In *Proceedings of Interspeech*, pages 2093–2097.

Martin Sundermeyer, Hermann Ney, and Ralf Schlüter. 2015. From feedforward to recurrent LSTM neural networks for language modeling. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 23(3):517–529.

Xiaoyan Tang and Jing Cao. 2015. Automatic genre classification via n-grams of part-of-speech tags. *Procedia-Social and Behavioral Sciences*, 198:474–478.

Bonnie Webber. 2009. Genre distinctions for discourse in the penn treebank. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 674–682, Suntec, Singapore, August. Association for Computational Linguistics.

Zhili Wu, Katja Markert, and Serge Sharoff. 2010. Fine-grained genre classification using structural learning algorithms. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 749–759, Uppsala, Sweden, July. Association for Computational Linguistics.