On the Replicability of Combining Word Embeddings and Retrieval Models

Luca Papariello(✉), Alexandros Bampoulidis, and Mihai Lupu

Research Studio Data Science, RSA FG, Vienna, Austria
{luca.papariello,alexandros.bampoulidis,mihai.lupu}@researchstudio.at

Abstract. We replicate recent experiments attempting to demonstrate an attractive hypothesis about the use of the Fisher kernel framework and mixture models for aggregating word embeddings towards document representations and the use of these representations in document classification, clustering, and retrieval. Specifically, the hypothesis was that the use of a mixture model of von Mises-Fisher (VMF) distributions instead of Gaussian distributions would be beneficial because of the focus on cosine distances of both VMF and the vector space model traditionally used in information retrieval. Previous experiments had validated this hypothesis. Our replication was not able to validate it, despite a large parameter scan space.

1 Introduction

The last 5 years have seen proof that neural network-based word embedding models provide term representations that are a useful information source for a variety of tasks in natural language processing. In information retrieval (IR), "traditional" models remain a high baseline to beat, particularly when considering efficiency in addition to effectiveness [6]. Combining the word embedding models with the traditional IR models is therefore very attractive and several papers have attempted to improve the baseline by adding in, in a more or less ad-hoc fashion, word-embedding information. Onal et al. [10] summarized the various developments of the last half-decade in the field of neural IR and group the methods in two categories: *aggregate* and *learn*. The first one, also known as *compositional distributional semantics*, starts from term representations and uses some function to combine them into a document representation (a simple example is a weighted sum). The second method uses the word embedding as a first layer of another neural network to output a document representation.

The advantage of the first type of methods is that they often distill down to a linear combination (perhaps via a kernel), from which an explanation about the representation of the document is easier to induce than from the neural network layers built on top of a word embedding. Recently, the issue of explainability in IR and recommendation is generating a renewed interest [15].

In this sense, Zhang et al. [14] introduced a new model for combining high-dimensional vectors, using a mixture model of von Mises-Fisher (VMF) instead

J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12036, pp. 50–57, 2020.
https://doi.org/10.1007/978-3-030-45442-5_7

of Gaussian distributions previously suggested by Clinchant and Perronnin [3]. This is an attractive hypothesis because the Gaussian Mixture Model (GMM) works on Euclidean distance, while the mixture of von Mises-Fisher (moVMF) model works on cosine distances—the typical distance function in IR.

In the following sections, we set up to replicate the experiments described by Zhang et al. [14]. They are grouped in three sets: classification, clustering, and information retrieval, and compare "standard" embedding methods with the novel moVMF representation.

2 Experimental Setup

In general, we follow the experimental setup of the original paper and, for lack of space, we do not repeat here many details, if they are clearly explained there.

2.1 Datasets

All experiments are conducted on publicly available datasets and are briefly described here below.

Classification. Two subsets of the movie review dataset: (i) the subjectivity dataset (subj) [11]; and (ii) the sentence polarity dataset (sent) [12].

Clustering. The 20 Newsgroups dataset[1] was used in the original paper, but the concrete version was not specified. We selected the "bydate" version, because it is, according to its creators, the most commonly used in the literature. It is also the version directly load-able in scikit-learn[2], making it therefore more likely that the authors had used this version.

Retrieval. The TREC Robust04 collection [13].

2.2 Models

The methods used to generate vectors for terms and documents are:

TF-IDF. The basic term frequency - inverse document frequency method [5]. *Implemented in the scikit-learn library*[3].

LSI. Latent Semantic Indexing [4].

LDA. Latent Dirichlet Allocation [2].

cBoW. Word2vec [9] in the Continuous Bag-of-Word (cBow) architecture.

PV-DBOW/DM. Paragraph vector (PV) is a document embedding algorithm that builds on Word2vec. We use here both its implementations: Distributed Bag-of-Words (PV-DBOW) and Distributed Memory (PV-DM) [7].

[1] http://qwone.com/~jason/20Newsgroups/.

[2] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html.

[3] https://scikit-learn.org/stable/.

The LSI, LDA, cBoW, and PV implementations are available in the gensim library[4].

Fisher Kernel (FK). The FK framework offers the option to aggregate word embeddings to obtain fixed-length representations of documents. We use Fisher vectors (FV) based on (i) a Gaussian mixture model (FV-GMM) and (ii) a mixture of von Mises-Fisher distributions (FV-moVMF) [1].

We first fit (i) a GMM and (ii) a moVMF model on previously learnt continuous word embeddings. The fixed-length representation of a document X containing T words w_i—expressed as $X = \{E_{w_1}, \ldots, E_{w_T}\}$, where E_{w_i} is the word vector representation of word w_i—is then given by $\mathcal{G}^X = [\mathcal{G}_1^X, \ldots, \mathcal{G}_K^X]$, where K is the number of mixture components. The vectors $\mathcal{G}_i^X$, having the dimension (d) of the word vectors E_{w_i}, are explicitly given by [3,14]:

$$\text{(i)} \quad \mathcal{G}_i^X = \frac{1}{\sqrt{\omega_i}} \sum_{t=1}^{T} \gamma_t(i) \frac{x_t - \mu_i}{\sigma_i}, \quad \text{and} \quad \text{(ii)} \quad \mathcal{G}_i^X = \sum_{t=1}^{T} \frac{\gamma_t(i) x_t d}{\omega_i \kappa_i}, \qquad (1)$$

where ω_i are the mixture weights, $\gamma_t(i) = p(i|x_t)$ is the soft assignment of x_t to (i) Gaussian and (ii) VMF distribution i, and $\sigma_i^2 = \text{diag}(\Sigma_i)$, with Σ_i the covariance matrix of Gaussian i. In (i), σ_i refers to the mean vector; in (ii) it indicates the mean direction and κ_i is the concentration parameter.

We implement the FK-based algorithms by ourselves, with the help of the scikit-learn library for fitting a mixture of Gaussian models and of the Spherecluster package[5] for fitting a mixture of von Mises-Fisher distributions to our data. The implementation details of each algorithm are described in what follows.

3 Experimental Results

Each of the following experiments is conceptually divided in three phases. First, text processing (e.g. tokenisation); second, creating a fixed-length vector representation for every document; finally, the third phase is determined by the goal to be achieved, i.e. classification, clustering, and retrieval.

For the first phase the same pre-processing is applied to all datasets. In the original paper, this phase was only briefly described as tokenisation and stopword removal. It is not given what tokeniser, linguistic filters (stemming, lemmatisation, etc.), or stop word list were used. Knowing that the gensim library was used, we took all standard parameters (see provided code[6]). Gensim however does not come with a pre-defined stopword list, and therefore, based on our own experience, we used the one provided in the NLTK library[7] for English.

[4] https://radimrehurek.com/gensim/.
[5] https://github.com/jasonlaska/spherecluster.
[6] https://rsagit.researchstudio.at/lpapariello/ecir_2020.git.
[7] https://www.nltk.org.

For the second phase, transforming terms and documents to vectors, Zhang et al. [14] specify that all trained models are 50 dimensional. We have additionally experimented with dimensionality 20 (used by Clinchant and Perronnin [3] for clustering) and 100, as we hypothesized that 50 might be too low. The TF-IDF model is 5000 dimensional (i.e. only the top 5000 terms based on their tf-idf value are used), while the Fischer-Kernel models are $15 \times d$ dimensional, where $d = \{20, 50, 100\}$, as just explained. In what follows, d refers to the dimensionality of LSI, LDA, cBow, and PV models.

The cBoW and PV models are trained using a default window size of 5, keeping both low and high-frequency terms, again following the setup of the original experiment. The LDA model is trained using a chunk size of 1000 documents and for a number of iterations over the corpus ranging from 20 to 100. For the FK methods, both fitting procedures (GMM and moVMF) are independently initialised 10 times and the best fitting model is kept.

For the third phase, parameters are explained in the following sections.

3.1 Classification

Logistic regression is used for classification in Zhang et al., and therefore also used here. The results of our experiments, for $d = 50$ and 100-dimensional feature vectors, are summarised in Table 1. For all the methods, we perform a parameter scan of the (inverse) regularisation strength of the logistic regression classifier, as shown in Fig. 1(a) and (b). Additionally, the learning algorithms are trained for a different number of epochs and the resulting classification accuracy assessed, cf. Fig. 1(c) and (d).

Table 1. Results of classification experiments on the *subj* and *sent* datasets. Shown are the mean accuracy and standard deviation, under 10-fold cross-validation, for optimally chosen hyperparameters (i.e. top values in Fig. 1).

Model	50-dim.		100-dim.	
	Subj	Sent	Subj	Sent
TF-IDF	89.3 ± 0.7	75.9 ± 1.3	89.3 ± 0.7	75.9 ± 1.3
LSI	82.5 ± 1.0	63.2 ± 0.9	84.3 ± 1.0	65.4 ± 1.6
LDA	62.2 ± 1.7	56.5 ± 1.6	62.1 ± 1.0	57.1 ± 1.5
cBow	89.0 ± 1.1	70.1 ± 1.3	89.2 ± 1.6	71.4 ± 1.1
PV-DBOW	88.7 ± 1.1	65.8 ± 1.4	89.4 ± 0.8	68.3 ± 1.3
PV-DM	84.0 ± 1.5	68.6 ± 1.3	88.0 ± 0.9	70.3 ± 1.1
FV-GMM	89.1 ± 0.9	68.5 ± 1.5	88.7 ± 0.9	72.4 ± 1.1
FV-moVMF	89.5 ± 0.9	70.1 ± 0.9	88.7 ± 1.4	71.2 ± 1.5

Figure 1(a) indicates that cBow, FV-GMM, FV-moVMF, and the simple TF-IDF, when properly tuned, exhibit a very similar accuracy on *subj*—the given

confidence intervals do not indeed allow us to identify a single, best model. Surprisingly, TF-IDF outperforms all the others on the *sent* dataset (Fig. 1(b)). Increasing the dimensionality of the feature vectors, from $d = 50$ to 100, has the effect of reducing the gap between TF-IDF and the rest of the models on the *sent* dataset (see Table 1).

3.2 Clustering

For clustering experiments, the obtained feature vectors are passed to the k-means algorithm. The results of our experiments, measured in terms of Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI), are summarised in Table 2. We used both $d = 20$ and 50-dimensional feature vectors. Note that the evaluation of the clustering algorithms is based on the knowledge of the ground truth class assignments, available in the 20 Newsgroups dataset.

As opposed to classification, clustering experiments show a generous imbalance in performance and firmly speak in favour of PV-DBOW. Interestingly, TF-IDF, FV-GMM, and FV-moVMF, all providing high-dimensional document representations, have a low clustering effectiveness.

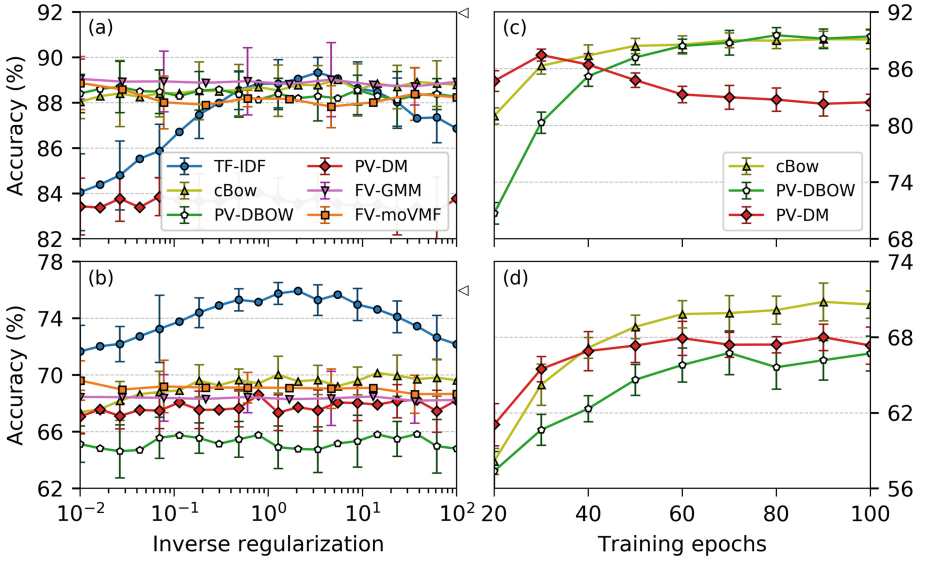

Fig. 1. Results of classification experiments, for 50-dimensional feature vectors, on the *subj* dataset [top panels, (a) and (c)] and *sent* dataset [bottom panels, (b) and (d)]. LSI and LDA achieve low accuracy (see Table 1) and are omitted here for visibility. The left panels [(a) and (b)] show the effect of (inverse) regularisation of the logistic regression classifier on the accuracy, while the right panels [(c) and (d)] display the effect of training for the learning algorithms. The two symbols on the right axis in panels (a) and (b) indicate the best (FV-moVMF) results reported in [14].

Table 2. Results of clustering experiments (mean performance and standard deviation over 10 runs) in terms of Adjusted Rand Index (ARI) and Normalised Mutual Information (NMI).

Model	20-dim.		50-dim.	
	ARI	NMI	ARI	NMI
TF-IDF	0.4 ± 0.1	5.6 ± 0.3	0.4 ± 0.1	5.6 ± 0.3
LSI	0.6 ± 0.1	5.3 ± 0.8	0.5 ± 0.2	5.8 ± 0.3
LDA	23.0 ± 0.7	39.5 ± 0.2	20.8 ± 1.1	40.2 ± 0.8
cBow	31.2 ± 0.4	50.4 ± 0.4	31.5 ± 0.3	51.2 ± 0.3
PV-DBOW	47.6 ± 1.2	63.4 ± 0.7	53.3 ± 1.3	66.1 ± 0.6
PV-DM	16.8 ± 0.6	44.8 ± 0.3	30.1 ± 0.9	53.2 ± 0.5
FV-GMM	1.4 ± 0.1	9.2 ± 0.8	1.0 ± 0.1	9.8 ± 1.3
FV-moVMF	2.1 ± 0.2	13.3 ± 1.3	1.6 ± 0.2	14.9 ± 1.6

3.3 Document Retrieval

For these experiments, we extracted from every document of the test collection all the raw text, and preprocessed it as described in the beginning of this section. The documents were indexed and retrieved for BM25 with the Lucene 8.2 search engine. We experimented with three topic processing ways: (1) title only, (2) description only, and (3) title and description. The third way produces the best results and closest to the ones reported by Zhang et al. [14], and hence are the only ones reported here.

An important aspect of BM25 is the fact that the variation of its parameters k_1 and b could bring significant improvement in performance, as reported by Lipani et al. [8]. Therefore, we performed a parameter scan for $k_1 \in [0, 3]$ and $b \in [0, 1]$ with a 0.05 step size for both parameters. For every TREC topic, the scores of the top 1000 documents retrieved from BM25 were normalised to [0,1]

Table 3. Results of retrieval experiments with 95% confidence intervals.

Model	Zhang et al.		Replicated		Replicated Best BM25	
	MAP	P@20	MAP	P@20	MAP	P@20
BM25	24.10	33.70	22.80 ± 2.51	33.21 ± 2.95	22.94 ± 2.50	33.63 ± 2.99
LSI	3.40	3.90	0.39 ± 0.33	1.20 ± 0.74	–	–
LDA	4.70	5.60	0.37 ± 0.13	0.96 ± 0.37	–	–
cBow	7.20	11.10	3.84 ± 0.78	8.84 ± 1.33	–	–
FV-GMM	9.80	12.40	5.72 ± 1.14	11.24 ± 1.79	–	–
FV-moVMF	11.20	13.90	3.16 ± 0.60	7.71 ± 1.23	–	–
BM25+LSI	25.30	36.60	23.11 ± 2.54	33.29 ± 2.95	23.22 ± 2.53	33.73 ± 3.00
BM25+LDA	25.30	36.30	22.88 ± 2.52	33.37 ± 2.96	22.99 ± 2.53	33.96 ± 2.98
BM25+cBow	25.30	36.50	23.78 ± 2.55	34.22 ± 2.95	23.88 ± 2.55	34.72 ± 2.98
BM25+FV-GMM	25.40	36.30	23.51 ± 2.55	34.08 ± 2.93	23.68 ± 2.54	34.54 ± 2.96
BM25+FV-moVMF	25.60	36.70	23.45 ± 2.53	33.88 ± 2.95	23.58 ± 2.54	34.34 ± 2.96

with the min-max normalisation method, and were used in calculating the scores of the documents for the combined models [14].

The original results, those of our replication experiments with standard ($k_1 = 1.2$ and $b = 0.75$) and best BM25 parameter values—measured in terms of Mean Average Precision (MAP) and Precision at 20 (P@20)—are outlined in Table 3.

4 Conclusions

We replicated previously reported experiments that presented evidence that a new mixture model, based on von Mises-Fisher distributions, outperformed a series of other models in three tasks (classification, clustering, and retrieval—when combined with standard retrieval models).

Since the source code was not released in the original paper, important implementation and formulation details were omitted, and the authors never replied to our request for information, a significant effort has been devoted to reverse engineer the experiments. In general, for none of the tasks were we able to confirm the conclusions of the previous experiments: we do not have enough evidence to conclude that FV-moVMF outperforms the other methods. The situation is rather different when considering the effectiveness of these document representations for clustering purposes: we find indeed that the FV-moVMF significantly underperforms, contradicting previous conclusions. In the case of retrieval, although Zhang et al.'s proposed method (FV-moVMF) indeed boosts BM25, it does not outperform most of the other models it was compared to.

Acknowledgments. Authors are partially supported by the H2020 Safe-DEED project (GA 825225).

References

1. Banerjee, A., Dhillon, I.S., Ghosh, J., Sra, S.: Clustering on the unit hypersphere using von Mises-Fisher distributions. J. Mach. Learn. Res. **6**, 1345–1382 (2005)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
3. Clinchant, S., Perronnin, F.: Aggregating continuous word embeddings for information retrieval. In: Proceedings of the Workshop on Continuous Vector Space Models and Their Compositionality, pp. 100–109 (2013)
4. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. J. Am. Soc. Inf. Sci. **41**(6), 391–407 (1990)
5. Harris, Z.S.: Distributional structure. Word **10**(2–3), 146–162 (1954)
6. Hofstätter, S., Hanbury, A.: Let's measure run time! Extending the IR replicability infrastructure to include performance aspects. In: Proceedings of the Open-Source IR Replicability Challenge Co-Located with 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, OSIRRC@SIGIR 2019, Paris, France, 25 July 2019, pp. 12–16 (2019)
7. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)

8. Lipani, A., Lupu, M., Hanbury, A., Aizawa, A.: Verboseness fission for BM25 document length normalization. In: Proceedings of the 2015 International Conference on the Theory of Information Retrieval, pp. 385–388. ACM (2015)
9. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013)
10. Onal, K.D., et al.: Neural information retrieval: at the end of the early years. Inf. Retrieval J. **21**(2), 111–182 (2018). https://doi.org/10.1007/s10791-017-9321-y
11. Pang, B., Lee, L.: A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, ACL 2004, pp. 271–278 (2004)
12. Pang, B., Lee, L.: Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, ACL 200, pp. 115–124. Association for Computational Linguistics (2005)
13. Voorhees, E.M.: The TREC robust retrieval track. SIGIR Forum **39**(1), 11–20 (2005)
14. Zhang, R., Guo, J., Lan, Y., Xu, J., Cheng, X.: Aggregating neural word embeddings for document representation. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) ECIR 2018. LNCS, vol. 10772, pp. 303–315. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76941-7_23
15. Zhang, Y., Zhang, Y., Zhang, M., Shah, C.: EARS 2019: the 2nd international workshop on explainable recommendation and search. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, 21–25 July 2019, pp. 1438–1440 (2019)