Flexible Similarity Search of Semantic Vectors Using Fulltext Search Engines

Michal Růžička[1], Vít Novotný[1], Petr Sojka[1],
Jan Pomikálek[2], and Radim Řehůřek[2]

[1] Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czechia
{mruzicka,witiko}@mail.muni.cz, sojka@fi.muni.cz
ORCID: 0000-0001-5547-8720, 0000-0002-3303-4130, 0000-0002-5768-4007
[2] RaRe Technologies
{honza,radim}@rare-technologies.com

Abstract. Vector representations and vector space modeling (VSM) play a central role in modern machine learning. In our recent research we proposed a novel approach to 'vector similarity searching' over dense semantic vector representations. This approach can be deployed on top of traditional inverted-index-based fulltext engines, taking advantage of their robustness, stability, scalability and ubiquity.
In this paper we validate our method using varied datasets ranging from text representations and embeddings (LSA, doc2vec, GloVe) to SIFT descriptors of image data.
We show how our approach handles the indexing and querying in these domains, building a fast and scalable vector database with a tunable trade-off between vector search performance and quality, backed by a standard fulltext engine such as Elasticsearch.

1 Introduction

The most important challenge in document similarity searches is the retrieval of documents that are relevant to the query. [9] Responding to this challenge involves a full understanding and representation of the document's semantics and searching based on them. Decades of research and development have led to mature high-dimensional vector space models [16] for the representation of various data using distributional semantics and deep learning [2,1,11].

In our recent paper [13], we introduced a novel method of encoding semantic vectors to so-called 'feature tokens'. These tokens can then be indexed and searched using standard inverted-index-based information retrieval (IR) systems such as Sphinx, Lucene, Elasticsearch, or Solr, all of which are freely available. This allows us to exploit the years of development behind modern fulltext engines, meeting the scalability and robustness demands of modern IR applications.

In [13], we evaluated our method using Latent Semantic Analysis (LSA) [2] on English Wikipedia, which comprises over 4 million articles. Cosine similarity was used to rank and evaluate retrieval results. In this paper, we evaluate our method on datasets with different representations using new similarity measures.

2 Semantic Vector Encoding for Inverted-Index-Based IR Systems

The standard representation of documents in the Vector Space Model (VSM) [15] uses term feature vectors of extremely high dimensionality. To map the feature space onto a smaller and denser latent semantic subspace, one may use a body of techniques, including Latent Semantic Analysis (LSA) [2], Latent Dirichlet Allocation (LDA) [1] or the many variants of Locality-sensitive hashing (LSH) [4]. Dimensionality reduction techniques proposed in [3] allow a consistent speedup while showing that not all features are equally discriminative and have a different impact on efficiency, due to their density distribution.

We decided to combine this knowledge with robust and optimized ready-to-use IR systems such as Elasticsearch [5]. To that end, we encode the vector features of document vectors into string tokens – so called 'feature tokens' – which are indexed by standard IR systems such as Elasticsearch.

At query time, we extract feature tokens from the query document and use them to quickly retrieve similar vectors using standard fulltext searches. This small set of candidate vectors is then reranked by calculating a slow but exact similarity metric (such as cosine similarity) between every candidate vector and the query vector, which gives us an ordered list of matching vectors (documents).

Feature tokens are constructed from the vector features by encoding the feature position and the feature value at a *selected precision*, effectively making some of the previously distinct feature values the same for the search engine.

To further speed up the search process, we use two 'high-pass filtering' techniques on query feature tokens: *Trimming* discards features whose values are below a fixed threshold and *best* keeps only a fixed number of features with the greatest absolute values.

The intuition behind the encoding scheme is a trade-off between *increasing feature sparsity* and *retaining search quality*. We show that some types of sparsification only lead to a negligible loss in quality, allowing an efficient use of inverted-index IR engines.

A detailed description of our method is available in [13]. In the rest of this paper, we will evaluate our method on datasets with diverse vector-space representations and using various measures to rerank the candidate vectors. The results show that our method works reasonably well in all the tested cases.

3 Tested Datasets and Experimental Setup

To evaluate our method of document similarity search, we used ScaleText [14] based on Elasticsearch [5] as our IR system. Elasticsearch was used in its 'vanilla' setup without any advanced features of Elasticsearch, such as custom scoring, tokenization, or n-gram analyzers. Our method requires only basic text retrieval functionality and using a fulltext engine other than Elasticsearch is quite straightforward.

In our experiments, we used the following datasets:

en-wiki The English Wikipedia dataset, which consists of 4,181,352 articles. We converted all the documents into vectors using (1) **LSA** [2] with 400 dimensions,[3] and (2) **doc2vec** [8] with 400 dimensions.

wiki-2014+gigaword-5 A dataset of pre-trained word vectors using the **GloVe** representation on the export of Wikipedia from early 2014, and on the English Gigaword Fifth Edition[4] data as provided by the GloVe project. [12]
The vectors were trained on a corpus of 6 billion tokens (uncased) with 400 thousand vocabulary entries. The dataset is provided in four variants with 50, 100, 200, and 300 dimensions per vector.

common-crawl A dataset of pre-trained word vectors using the **GloVe** representation on the Common Crawl project[5] data as provided by the GloVe project.
The dataset was provided in two variants, each with a different training corpus: (1) 42 billion tokens (uncased) with 1.9 million vocabulary entries, and (2) 840 billion tokens (cased) with 2.2 million vocabulary entries. In both cases, 300 dimensions per vector were used.

twitter A dataset of pre-trained word vectors using the **GloVe** representation on the export of 2 billion short tweet messages from the Twitter social networkas provided by the GloVe project.
The vectors were trained on a corpus of 27 billion tokens (uncased) with 1.2 million vocabulary entries. The dataset is provided in four variants with 25, 50, 100, and 200 dimensions per vector.

texmex A dataset of vectors for the SIFT descriptors of image data [7] as provided by the TEXMEX project[6].
We have used two datasets provided by the project: (1) `ANN_SIFT10K` dataset consisting of 10 thousand vectors, and (2) `ANN_SIFT1M` dataset consisting of 1 million vectors. In both cases, 128 vector dimensions were used.

Please note that, except for the *en-wiki-lsa* and *en-wiki-doc2vec* datasets, we did not generate the semantic vectors ourselves. Rather, we used the pre-computed vectors as provided. The unmodified vectors were indexed using the ScaleText IR system.

3.1 Evaluation Method

The aim of our evaluation was to investigate how well our approximate nearest neighbor search performs in comparison with an exact brute-force search on the above datasets. We decided to use cosine similarity as the similarity metric for the reranking of candidate vectors.

From each dataset, we randomly selected 1,000 document vectors to act as our query vectors. By doing an exact brute force search over all vectors in the

[3] The same dataset was used in [13]
[4] `https://catalog.ldc.upenn.edu/LDC2011T07`
[5] `https://commoncrawl.org/`
[6] `http://corpus-texmex.irisa.fr/`

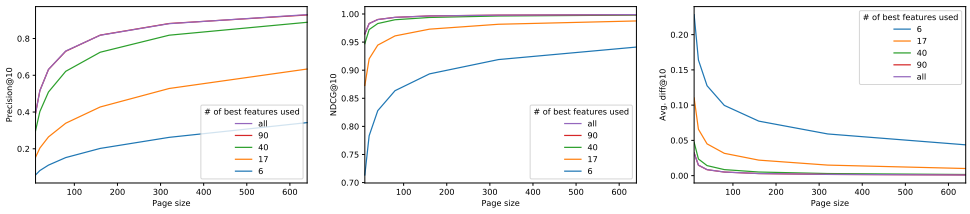
dataset, we identified the 10 most similar document vectors for each query vector. This became our 'gold standard'.

For each of the 1,000 query vectors, we performed a fulltext search. We measured the relevance of the retrieved documents using three metrics: Precision at 10 (Precision@10), which measures the overlap between the retrieved documents and the gold standard, the Normalized Discounted Cumulative Gain at 10 (NDCG@10) [10, Section 8.4], where the relevance value of a retrieved document vector was taken to be its cosine similarity to the query vector, and the mean cumulative loss at 10 (Avg. diff.@10), which compares cosine similarities between the top 10 retrieved document vectors and the query vector with cosine similarities between the 10 most similar document vectors and the query vector.

During the evaluation, we experimented with various configurations of the high-pass filtering (**trimming** and **best** parameters as described in Section 2), and with the number of vectors retrieved from Elasticsearch (the **page** parameter) to see their impact on the quality of results.

4 Results

In this section we present multiple graphs showing the impact of the number of **best** features selected (with no **trimming**) and the **page** size (the number of search results retrieved from Elasticsearch) on Precision@10, NDCG@10 and Avg. diff.@10 for numerous datasets (see Section 3).

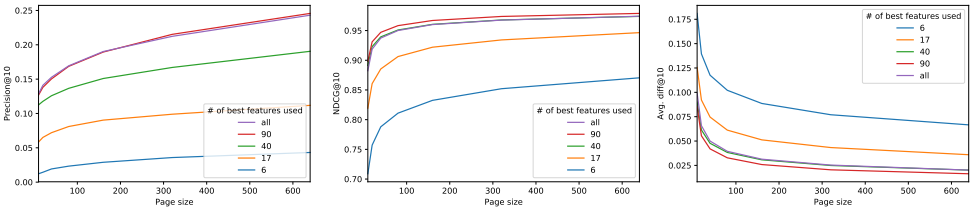


(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 1. The impact of the configurations for the *en-wiki-lsa-400d* dataset. Keeping only the best 90 features is almost indistinguishable from using all 400 features in terms of Precision@10. This result was reported in [13].

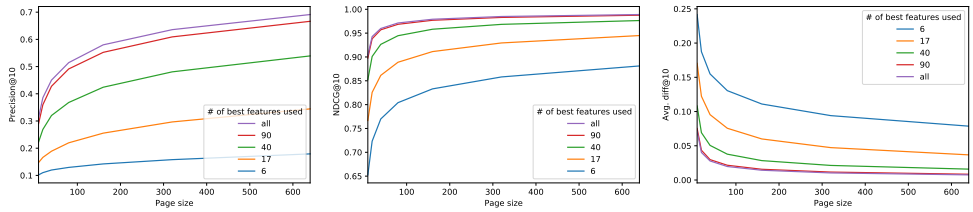
All of the graphs show that the documents retrieved with no high-pass filtering of query features closely approximate the gold standard, whereas reducing the number of features decreases the releavance of the retrieved documents by all metrics.

Comparing the results for the *wiki-2014+gigaword-5-glove* and *twitter-glove* datasets, where multiple variants with different dimensionalities are available (see Figures 3, 4, 5, 6, and Figures 9, 10, 11, 12), it seems that the dimensionality of the semantic vectors should not be significantly lower than 200 to achieve the best results. High-pass filtering using only 90 **best** features results in only a small



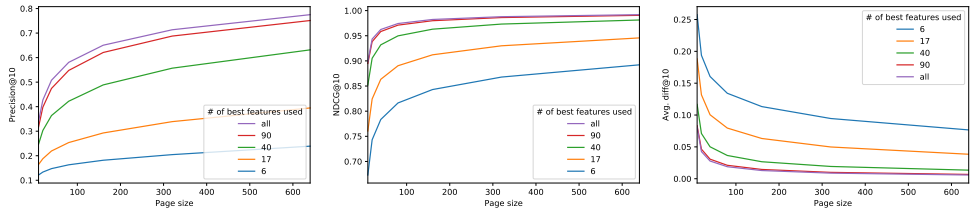
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 2. The impact of the configurations for the *en-wiki-doc2vec-400d* dataset. The markedly low Precision@10 compared to other datasets may indicate a suboptimal setting of the doc2vec algorithm configuration parameters. Compare this to the results for LSA in Figure 1.



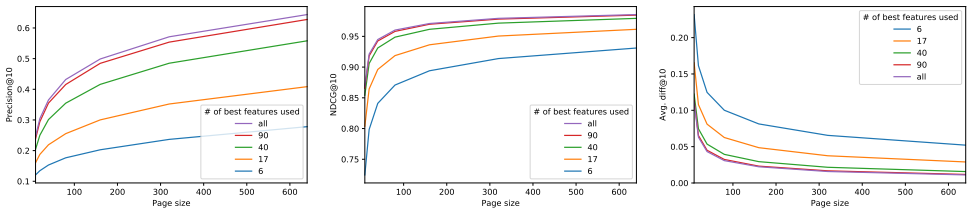
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 3. The impact of the configurations for the *wiki-2014+gigaword-5-glove-300d* dataset. Except for the slightly lower Precision@10, our search method achieves comparable results for the GloVe representation and for the LDA representation of a similar dataset *en-wiki-lsa-400d* in Figure 1.



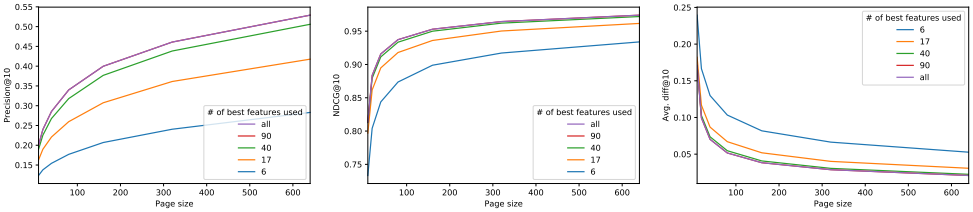
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 4. The impact of the configurations for the *wiki-2014+gigaword-5-glove-200d* dataset. Lowering the dimensionality of the dataset from 300 to 200 (cf. Figure 3) has virtually no impact on the results.



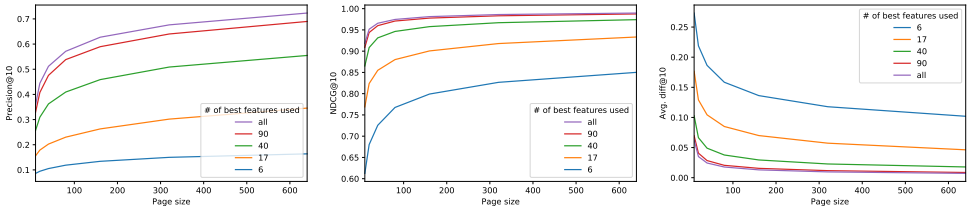
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 5. The impact of the configurations for the *wiki-2014+gigaword-5-glove-100d* dataset. Lowering the dimensionality of the dataset from 200 to 100 (cf. Figure 4) results in a small decrease in the relevance of the retrieved documents.



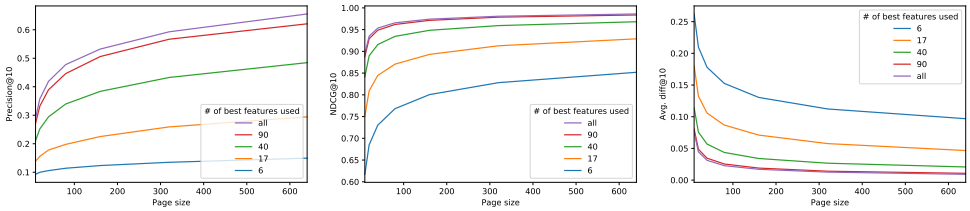
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 6. The impact of the configurations for the *wiki-2014+gigaword-5-glove-50d* dataset. Lowering the dimensionality of the dataset from 100 to 50 (cf. Figure 5) results in a further decrease in the relevance of the retrieved documents. A comparison of the curves reveals that when more dimensions are used (cf. Figures 3, 4, and 5), the documents retrieved with a smaller **page** size are more affected by the decrease in the dimensionality than the documents retrieved with a larger **page** size.



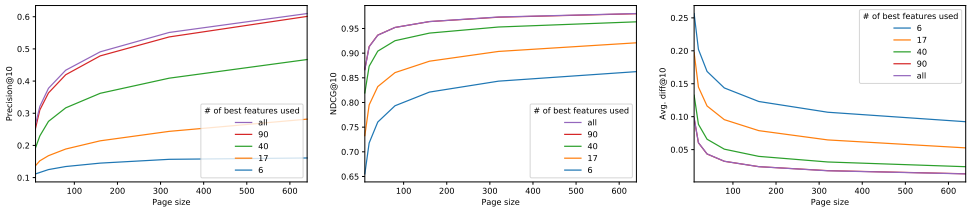
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 7. The impact of the configurations for the *common-crawl-glove-840B-300d* dataset. Using the GloVe representation in 300 dimensions on the Common Crawl project data gives us results that are almost identical to the results for the GloVe representation of the *wiki-2014+gigaword-5-glove* dataset with the same dimensionality (cf. Figure 3). This showcases the stability of our search method.



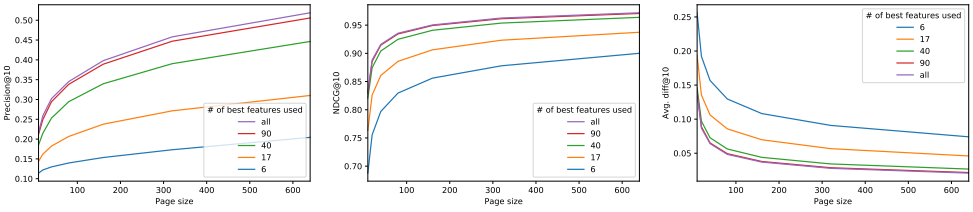
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 8. The impact of the configurations for the *common-crawl-glove-42B-300d* dataset. Compared to the results for the *common-crawl-glove-840B-300d* dataset with 840 billion cased tokens in Figure 7, a slightly lower Precision@10 is reached with 42 billion uncased tokes used in this dataset.



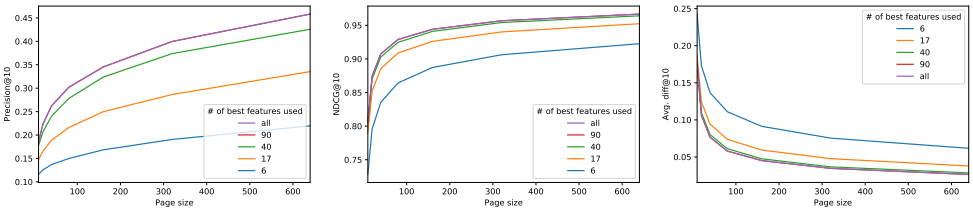
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 9. The impact of the configurations for the *twitter-glove-200d* dataset. Compared to the results for the *wiki-2014+gigaword-5-glove-200d* dataset with the same dimensionality in Figure 4, the relevance of the retrieved documents is slightly lower for this dataset. The documents retrieved with a lower **page** size are more affected.



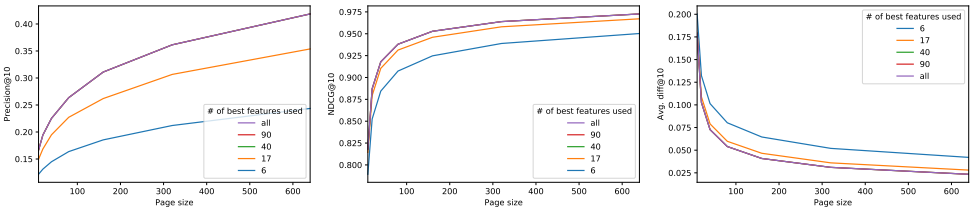
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 10. The impact of the configurations for the *twitter-glove-100d* dataset. The impact of the decrease in the dimensionality from 200 to 100 (cf. Figure 9) is comparable to the same decrease for the *wiki-2014+gigaword-5-glove* dataset (cf. Figures 4 and 5).



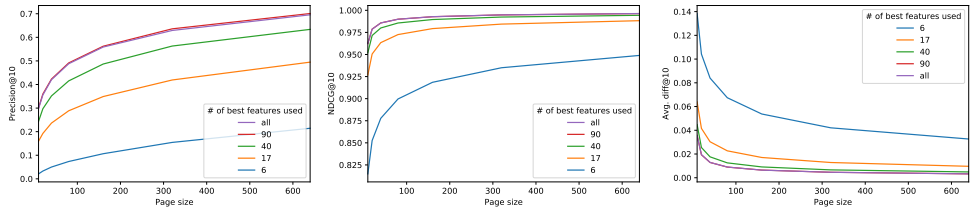
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 11. The impact of the configurations for the *twitter-glove-50d* dataset. Since the vectors in the dataset consist of only 50 features, selecting 90 **best** features leads to the same results as using all features. Selecting 40 **best** features then leads to only a small decrease in the relevance of the retrieved documents.



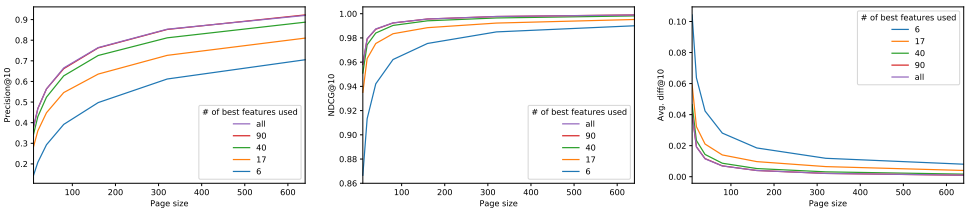
(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 12. The impact of the configurations for the *twitter-glove-25d* dataset. Using only 25 features results in a slight decrease in Precision@10 compared to using 50 features (cf. Figure 11).



(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 13. The impact of the configurations for the *texmex-sift-1M-128d* dataset. Despite using SIFT descriptors of image data rather than vectors derived from text documents, the curves are shaped similarly to the curves for the text datasets of Wikipedia+Gigaword and Twitter with comparable dimensionality (cf. Figures 5 and 10). However, the relevance of the retrieved documents is significantly higher for this dataset.



(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 14. The impact of the configurations for the *texmex-sift-10k-128d* dataset. In contrast to the *common-crawl-glove* dataset, where the larger dataset reached a higher relevance of the retrieved documents compared to the smaller one (see Figures 7 and 8), the results for the small TEXMEX dataset are better than for the larger one (cf. Figure 13).

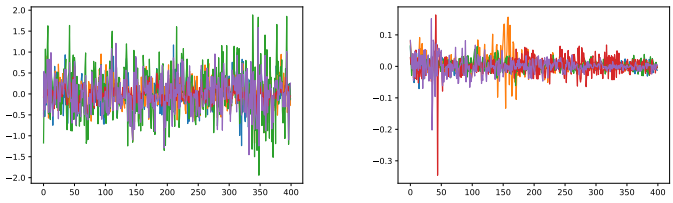
decrease in the relevance of retrieved documents while speeding up the search. This observation holds for datasets with a higher (≥ 200) original dimensionality (see Figures 1, 2, 3, 7, or 8). Even with these datasets, there is no clear benefit to using more than 90 **best** features for querying.

The absolute quality of the results improves all the way up to and including the maximum tested **page** size. This is expected, as increasing the number of candidate vectors preselected by Elasticsearch increases the chance of retrieving relevant documents. It is expected that a further increase in the **page** size would lead to further improvements in the relevance of the retrieved documents at the cost of performance.

The absolute values of Precision@10 show a significant drop in the relevance of the retrieved documents for datasets with low (≤ 100) dimensionalities. However, NDCG@10 and Avg. diff.@10 are almost unaffected. We expect that this is the result of the higher mutual similarity of the low-dimensional vectors, which makes it difficult to retrieve precisely the gold standard documents. Precision@10 penalizes the retrieval of all documents with the exception of those in the gold standard, whereas NDCG@10 and Avg. diff@10 will rate any document vectors similar to those in the gold standard as relevant.

We have concluded that the unusually low Precision@10 for the 400-dimensional *en-wiki-doc2vec* dataset 2) has a similar cause. Investigating the document vectors in the dataset, we discovered that the feature values lack any 'peaks' that determine the main topics of the documents in other datasets such as *en-wiki-lsa* (see Figures 15a and 15b). This gives all the *en-wiki-doc2vec* document vectors a strong mutual similarity. Tuning the doc2vec parameters to better differentiate between the documents could be expected solve this issue.

In general, the results on the other datasets confirm the behaviour of our method tested on the single *en-wiki-lsa* dataset in [13], which is what we set out to verify in this paper.



(a) doc2vec (b) LSA

Fig. 15. Visualization of all feature values for 5 random vectors from the *en-wiki-doc2vec-400d* dataset and for 5 random vectors from the *en-wiki-lsa-400d* dataset.

4.1 Using Euclidean Similarity for TEXMEX

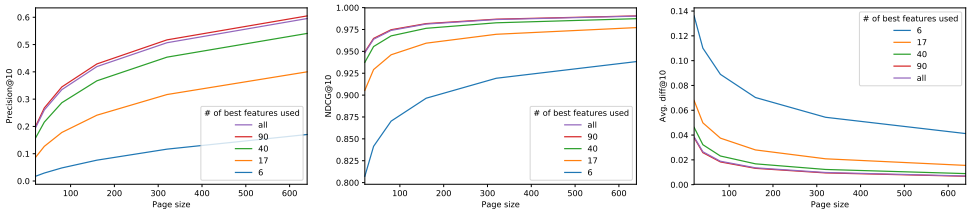
Since the *texmex* dataset is provided together with sets of query vectors and the corresponding ground truth for approximate nearest neighbor search evaluation, we wanted to take advantage of these independent relevance judgements to further validate our method. To do so, we needed to modify our evaluation method to be compatible with the provided query vectors and the ground truth.

The *texmex* vectors consist only of integers, so we divided the vectors by the Euclidean norm of the largest vector in the dataset prior to indexing them in ScaleText. This was to ensure that the ScaleText string representation of vectors (feature tokens), which expects all feature values to have a fractional part that can be rounded to a given precision, works properly. This precondition is typically ensured by the normalization of the indexed vectors by ScaleText, which we disabled here to retain Euclidean distances between the vectors. The only exception is **trimming**, which still uses a normalized version of the vectors, so that vectors close to the origin are not completely 'trimmed away' in this step.

Instead of the cosine similarity ($v_1 v_2 / (\|v_1\| \|v_2\|)$), the candidate vectors are reranked using the generalized Radial Basis Function (RBF) kernel with $\gamma = 1$ ($\exp(-\gamma \|v_1 - v_2\|^2)$), which is a form of Euclidean similarity [6]. The generalized RBF kernel is also used for the computation of NDCG@10 and Avg. diff.@10 instead of cosine similarity. Therefore the absolute values of NDCG@10 and Avg. diff.@10 for the cosine and Euclidean similarities cannot be directly compared. Precision@10 does not use similarities and the results for the cosine and Euclidean similarities can therefore be compared.

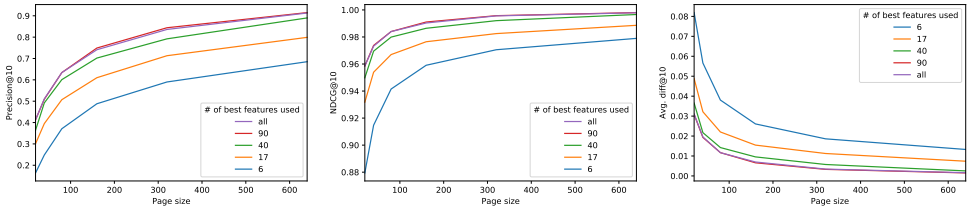
Results are shown in Figures 16 and 17. The shapes and the relative positions (in the case of NDCG@10 and Avg. Diff.@10) and even the absolute positions (in the case of Precision@10) of the curves are very similar to those for the cosine similarity (cf. Figures 13 and 14).

For the large (1M) SIFT dataset, using Euclidean similarity leads to slighly lower Precision@10 compared to cosine similarity. For the small (10k) SIFT dataset, there is no clear difference.



(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 16. The impact of the configurations for the *texmex-sift-1M-128d* dataset using Euclidean similarity.



(a) Precision@10 (b) NDCG@10 (c) Avg. diff.@10

Fig. 17. The impact of the configurations for the *texmex-sift-10k-128d* dataset using Euclidean similarity.

5 Conclusion

In this paper, we have demonstrated that our novel method of semantic vector encoding for inverted-index IR systems proposed in [13] works well not only on text corpora using traditional latent semantic techniques such as LSA with cosine similarity, but also on datasets from other domains, such as SIFT descriptors of image data, and using different representations such as doc2vec and GloVe with new similarity measures. Our method of vector encoding and filtering speeds up the search process, achieving an excellent approximation of the gold standard for all tested datasets, making our method suitable for diverse data sources and application domains.

Acknowledgments Funding by TA ČR Omega grant TD03000295 is gratefully acknowledged.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I., Lafferty, J.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (Mar 2003), `http://dl.acm.org/citation.cfm?id=944919.944937`
2. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by Latent Semantic Analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)

3. Digout, C., Nascimento, M.A., Coman, A.: Similarity search and dimensionality reduction: Not all dimensions are equally useful. In: Lee, Y., Li, J., Whang, K.Y., Lee, D. (eds.) Proc. of Database Systems for Advanced Applications: 9th Int. Conf., DASFAA 2004, Jeju Island, Korea, March 17–19, 2003. pp. 831–842. Springer (2004), http://dx.doi.org/10.1007/978-3-540-24571-1_73
4. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: VLDB '99, Proceedings of 25th International Conference on Very Large Data Bases, September 7–10, 1999, Edinburgh, Scotland, UK. pp. 518–529 (1999), http://www.vldb.org/conf/1999/P49.pdf
5. Gormley, C., Tong, Z.: Elasticsearch: The Definitive Guide. O'Reilly Media, Inc., 1st edn. (2015)
6. Haasdonk, B., Bahlmann, C.: Learning with Distance Substitution Kernels, pp. 220–227. Springer Berlin Heidelberg, Berlin, Heidelberg (2004), https://doi.org/10.1007/978-3-540-28649-3_27
7. Jegou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(1), 117–128 (Jan 2011)
8. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. CoRR abs/1405.4053 (2014), http://arxiv.org/abs/1405.4053
9. Li, H., Xu, J., et al.: Semantic matching in search. Foundations and Trends® in Information Retrieval 7(5), 343–469 (2014)
10. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA (2008)
11. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems. pp. 3111–3119 (2013)
12. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), http://www.aclweb.org/anthology/D14-1162
13. Rygl, J., Pomikálek, J., Řehůřek, R., Růžička, M., Novotný, V., Sojka, P.: Semantic vector encoding and similarity search using fulltext search engines (2017), https://arxiv.org/abs/1706.00957, preprint of the paper accepted to the ACL 2017 workshop RepL4NLP 2017.
14. Rygl, J., Sojka, P., Růžička, M., Řehůřek, R.: ScaleText: The Design of a Scalable, Adaptable and User-Friendly Document System for Similarity Searches: Digging for Nuggets of Wisdom in Text. In: Horák, A., Rychlý, P., Rambousek, A. (eds.) Proceedings of the Tenth Workshop on Recent Advances in Slavonic Natural Language Processing, RASLAN 2016. pp. 79–87. Tribun EU, Brno (2016), https://nlp.fi.muni.cz/raslan/2016/paper08-Rygl_Sojka_etal.pdf
15. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24, 513–523 (1988)
16. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975), https://doi.org/10.1145/361219.361220
17. Tange, O.: GNU Parallel – The Command-Line Power Tool. The USENIX Magazine 36(1), 42–47 (2011)