Explaining Topical Distances Using Word Embeddings

Nils Witt
German National Library of Economics
Kiel, Germany
Email: n.witt@zbw.eu

Christin Seifert
University of Passau
Passau, Germany
Email: christin.seifert@uni-passau.de

Michael Granitzer
University of Passau
Passau, Germany
Email: michael.granitzer@uni-passau.de

***Abstract*—Word and document embeddings have gained a lot of attention recently, because they tend to work well in text mining tasks. Yet, they elude humans intuition. In this paper we are making the attempt to explain the arithmetic difference between two document embeddings by a series of word embeddings. We present an algorithm that iteratively picks words from a vocabulary that closes the topical gap between the documents. Moreover, we present the Econstor16 corpus that was used for the experiments. Although not all words that are found are great matches, the algorithm is able to find sets of words that are reasonable to a human that reads both documents. Remarkably, some of the well-explaining words are mentioned in neither documents.**

1. Introduction

Tasks that deal with text understanding problems have to answer the question of how to represent text. This is, how can the text be represented in a machine-understandable way. A fixed length text representation is important, as most text mining algorithms require fixed length inputs. Furthermore, the preservation of the hidden structure of a document is crucial properties as well. For quite a long time bag of words (BOW) and n-gram [1] where most established. But BOW and n-gram fail to preseve the order of words (the n-gram window is usually small, which leads to a small context) and suffer from sparse and high dimensional vectors.

Later, LDA [2] was introduced. LDA reveals the topics that a document is made of, thereby yielding an improved performance in most text mining tasks. Interestingly, it was shown that topic models (like LDA) can be trained with the goal of associating similar topics to documents as humans do [3]. More recently, distributed word [4] and document [5] embeddings gained attention as they outperformed previous state-of-the-art systems in various classification tasks like sentiment analysis ([5], [6]).

Although it was shown that distributed embeddings capture semantics, the vectors themselves are hardly comprehensible by humans. However, the distances between those representations exhibit interesting properties, that are human-interpretable. Mikolov et. al. showed in [7] that the nearest neighbor of $X = wv(biggest) - wv(big) + wv(small)$ is $wv(''smallest'')$, where $wv(''word'')$ refers to the word vector of ”word”. Dai et. al. showed in [6] that it is also possible to add word and document vectors and obtain meaningful results. For example, they could find the Japanese equivalent of Lady Gaga by computing $dv(''LadyGaga'') - wv(''American'') + wv(''Japanese'')$ (where $dv(''document'')$ refers to the document vector of ”document”). In this paper we use semantics properties of vector embeddings for identifying a path (i.e. a set of words) between two document, that explains their differences. The contributions of this paper are the following:

- We will propose an algorithm that explains the difference between two documents by a adding and subtracting multiple words. For example:

  $$dv(''report2013'') \approx dv(''report2012'') - wv(''2012'') + wv(''2013'').$$

  More formally, we are looking for a set of words W that is a subset of the vocabulary V ($W \subseteq V$) such that $D < \epsilon$ (ϵ is a distance threshold), where D is defined by:

  $$D = dist(dv(Y), dv(X) + \sum_{i=0}^{|W|}(-1)^n wv(W_i)). \tag{1.1}$$

  n determines whether a word vector is to be added ($n = 0$) or subtracted ($n = 1$), whichever approximates better. More formally:

  $$n = \begin{cases} 1 & \text{if } dv(Y) - dv(X) + wv(W_i) > \\ & \quad dv(Y) - dv(X) - wv(W_i) \\ 0 & \text{otherwise.} \end{cases}$$

  Moreover, ϵ serves as a convergence criteria, $dist(A, B)$ measures the distance between A and B and (as mentioned earlier), $dv(doc)$ is the document embedding representation of the document doc.
  X and Y are arbitrary documents and V is a vocabulary. This main idea will be explained in more detail in section 3 followed by experiments in section 4.

- We also introduce the Econstor16 corpus, that contains more than 90,000 documents from the economics domain along with their meta data. Econstor16 was used to conduct the experiments. The corpus will be presented in section 6.

2. Motivation

In order to improve humans intuition of documents in high dimensional vector spaces, we wanted to investigate the question whether the distance between documents is human-interpretable. More specifically, we explored options to convert the space between documents into a representation that allows humans to grasp the difference and to reason about it. Taking this idea further, we considered summarizing the difference between two documents, which, in contrast to automatic summarization techniques [8], is applied to a pair of documents rather than to a single document. Automatic difference summarization may be a means to investigate unknown documents that are similar to known documents.

3. Approach

The spatial distance between document embeddings encodes the topical difference of those documents, which is illustrated by Figure 1. The green cluster, that is dominated by the author M. Hashem Pesaran, deals with theoretical economics and statistics. In contrast, documents in the pink cluster on the bottom right are about fiscal policy and financial risks. Likewise, most other cluster represent a certain topic.

In order to explore this property deeper, we performed nearest neighbor searches in the following setting:

1) For each run, pick 1,000 documents (to limit computation time).
2) Compute the cosine distances between all documents.
3) Select the document pair with the maximum distance.
4) For one of the documents from the previous step (random choice), find the nearest neighbor.

This yields document triples (D_x, D_y, D_a) where D_x and D_y are close to each other and D_a is far away from either of the other documents. Table 1 shows one example. Merely from the title and the author-supplied keywords list it can be seen that D_x and D_y examine related topics whereas D_a has no obvious commonalities with D_x and D_y. Reading the abstracts makes the situation even clearer: D_x and D_y investigate the influence of an economic crisis (D_x: Great Depression, D_y: Real Estate Crisis beginning in 2007) on wealth and unemployment. Whereas D_a analyzes the impact of a norm on the international trade. The cosine similarities in table 2 confirms the results.

Similar results where found for most repetitions of the process.

Based on this intuition, we will introduce an iterative algorithm, which approximates a document vector through a additive combination of another document vector and multiple word vectors.

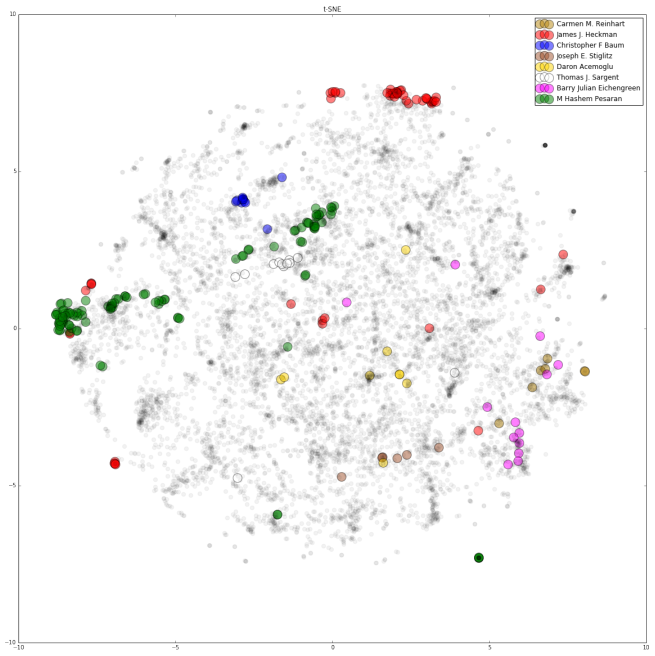

Figure 1. Visualization of a 600 dimensional document embedding of 15,000 documents randomly picked from the Econstor16 corpus. The dimensionality reduction was done using t-SNE [9]. The colors highlight documents from the top 8 US authors in the field of economics according to RePEc.

TABLE 1. Author Selected Keywords of Three Documents

Variable	Title	Key phrases
D_x	Wealth shocks, unemployment shocks and consumption in the wake of the Great Recession	Marginal Propensity to Consume, Wealth Shocks, Unemployment
D_y	Household debt and saving during the 2007 recession	Household Debt and Saving, 2007 Recession, Credit Access, Mortgage debt
D_a	The Impact of ISO 9000 Diffusion on Trade and FDI: A New Institutional Analysis	FDI, Trade, Transaction Costs, Institutions

TABLE 2. Cosine Similarities of the example documents on 100 dimensional vectors

Variables	Similarity
$dv(D_x), dv(D_y)$	0.5394
$dv(D_x), dv(D_a)$	-0.4052
$dv(D_y), dv(D_a)$	-0.1745

Given two documents (say, D_1 and D_2), how can we obtain words that, when added to one of the vectors, reduce the distance to the other? Or, reformulated in the regime of document embeddings: given two document vectors ($D_1^{dv} = dv(D_1)$ and $D_2^{dv} = dv(D_2)$) how can we find word vectors that, when added to one of the document vectors, reduce the distance to the other document vector? The optimal solution to this problem is simply to choose the difference ($\Delta = D_1^{dv} - D_2^{dv}$) of the vectors. But, since Δ most likely is not a word vector (with a corresponding word), we can't infer any semantic information from that fact. But instead we can search the vocabulary for the word vector that approximates Δ best. This will give us a vector Δ_{approx} (say, we want to approximate D_2^{dv} through D_1^{dv}) that has a smaller distance to D_2^{dv} than D_1^{dv}. This is explained more formally in Algorithm 1, where W_c contains the closest (i.e. the most similar) word whereas W_f contains the farthest word. In case $-W_f > W_c$, W_f will be subtracted, otherwise W_c will be added. Especially in very high dimensional space it is difficult to find an exact solution to this problem. Hence, a convergence criterion is needed, that, once it is met, stops the algorithm. We make three suggestions:

Algorithm 1 Document Vector Approximation

```
function DocVecApprox($model, D_1, D_2$)
    $path \leftarrow list()$
    $D_1^{dv} \leftarrow dv(D_1)$
    $D_2^{dv} \leftarrow dv(D_2)$
    $D_{approx} \leftarrow D_1^{dv}$
    while $not\ converged$ do
        $\Delta \leftarrow D_2^{dv} - D_{approx}$
        $W_c \leftarrow findClosestWord(\Delta, model)$
        $W_f \leftarrow findFarthestWord(\Delta, model)$
        if $W_c > -W_f$ then
            $D_{approx} \leftarrow D_{approx} + wv(W_c)$
            $append(W_c, path)$
        else
            $D_{approx} \leftarrow D_{approx} - wv(W_f)$
            $append(W_f, path)$
    return $path$
```

1) **Threshold**: The approximation is terminated once $abs(D_{approx} - D_2^{dv}) \leq \epsilon$ (see equation 1.1). This limits the runtime, but requires the user to define a threshold. An optimal threshold depends on the expected outcome and the dimensionality of the document space.
2) **K-nearest neighbors**: After each iteration the k-nearest neighbors of D_{approx} are determined. The approximation stops as soon as D_2 appears in that list. Likewise, the threshold criteria, a hyper parameters needs to be specified (i.e. the size of the list). But in contrast, this criteria guarantees certain quality (i.e. the result is among the k nearest neighbors of D_2).
3) **Local optimum found**: The distance stopped decreasing. This yields the best results on the cost of the longest execution time.

TABLE 3. Overview Document X, Example 1

Variable	X
Author	Hendrik Hagedorn
Title	In search of the marginal entrepreneur: Benchmarking regulatory frameworks in their effect on entrepreneurship
Keywords	Benchmarking method, entrepreneurship, incentives, dataset, regulation

TABLE 4. Overview Document Y, Example 1

Variable	Y
Author	John Hartwick
Title	Mining Gold for the Currency during the Pax Romana
Keywords	Gold coinage, Roman money supply, roman empire

TABLE 5. Approximation Process of two distant documents. The Similarity Columns indicate the Cosine Similarity between Y and the Current Approximation. Note that the Approximation was conducted in 100 Dimensional Space and then replicated in 600 Dimensional Space.

Iteration	**Vector**	**Similarity @100**	**Similarity @600**
initial	diff := Y - X	-0.44	-0.03
1	diff := diff - "job"	-0.14	0.01
2	diff := diff - "carbon"	0.12	0.03
3	diff := diff + "empire"	0.22	0.10
4	diff := diff + "goldsmith"	0.36	0.07
5	diff := diff - "country"	0.52	0.07
6	diff := diff - "interest"	0.61	0.07

4. Experiment

We conducted experiments in a semi-automatic fashion. The reason for this is, that the corpus (that will be presented in section 6) contained too many non-sense words. That was caused by flaws during the extraction of the plain text from the PDF files, which led to non-sense words in the vocabulary. The short term solution was to hand-pick the words that where added to the approximation path, rather than letting an algorithm decide. More precisely, the algorithm came up with a list of candidates and the first meaningful word was selected by hand. This effectively limited the amount of repetitions that could be undertaken. Nevertheless, during the semi-automatic testing, observations were made that will be exemplified using two examples. For the first example we selected two distant documents. The tables 3 and 4 give a brief overview over the content of the documents.

Table 5 presents the results of the approximation algorithm. There are some notable aspects:

1) Although the initial similarity is low (-1 is the maximum value here which denotes a diametrically opposed vector) it takes only a few iterations to achieve a high similarity.
2) Most of the words are actually meaningful to explain the difference between the two documents:

TABLE 6. Overview Document X, Example 2

Variable	X
Author	Hans-Werner Sinn
Title	Pareto Optimality int the Extraction of Fossile Fuels and the Greenhouse Effect
Keywords	global warming, resource extraction, Pareto optimality

TABLE 7. Overview Document Y, Example 2

Variable	Y
Author	Hans-Werner Sinn
Title	EU Enlargement and the Future of the Welfare State
Keywords	EU expansion, migration, labour market, welfare state

”empire” and ”goldsmith” account for the document Y and ”jobs” and ”interest” are specific to document X.

3) There are also words that are not obviously suitable, like ”carbon” and ”country”
4) It’s worth noting that, although it makes intuitively sense, the word ”goldsmith” is not used in either documents.
5) When the results of the approximation in 100 dimensions are replicated in 600 dimensions, we find, that iteration 3 produces the best result, and afterwards the process is stuck in a local optimum.

The setting for the second example differs in two aspects: (1) we used a 600 dimensional embedding rather than a 100 dimensional one to drive the approximation and (2) we selected documents that were written by the same author, but on different topics. An overview over these documents is given in table 6 and 7. The respective approximation process is depicted in table 8. Again, we find suitable words that explain the difference well (”fuel”, ”diesel”, ”debt”, ”stock”) and we find words that are less suitable (”non-statistical”, ”industrial”). But in contrast to the first example, the approximation converges much slower. This is due to additional dimensions of the document embedding. Likewise, the first example, we find a word that is suitable despite the fact that it’s not mentioned in either documents (”diesel”). Moreover, the convergence in 100 dimensions is comparatively slow and does not increase monotonically.

5. Discussion

Throughout the paper cosine similarity was used to compare vectors. At the beginning euclidean distance was used, but we found that nearby documents rarely had topical commonalities. Hence, we decided to use cosine similarity only. Cosine similarity was also used by Dai et al. [6].

Another observation made during the semi-automatic approximation was that author names were often among the candidates. This is presumably caused by the reference section at the end of papers that make up the corpus. We decided to exclude author names for this experiment, because domain knowledge is required to benefit from this information. However, author names may be useful in a different context. What we also found was, that words that were already on the path often reappeared in the candidate lists. Though, using them always led to an increased distance. Henceforth, words were never added twice to the path. Concerning the dimensionality of the document/word space two main conclusion can be drawn. (1) Fewer dimensions lead to faster convergence, which limits the length of the path describing the difference between two documents and (2) a path that was created using some dimensionality is not necessarily a suitable path in another dimensionality. Which raises the question, whether higher dimensional spaces produce better (i.e. topical more suitable) approximations, which will be subject to future work. One last aspect, although already mentioned, should be brought to attention again: occasionally, words that are in neither document can explain the differences well. This property is remarkable. It was inherited from word embeddings, where for example ”Monday” and ”Wednesday” are closely related, which makes them interchangeable (at least as far as the document embedding is concerned). Hence, a document that comprises ”Monday” may also be well explained by ”Wednesday”. Also, we want to make suggestions for potential applications that use the approximation algorithm in recommendation scenarios:

- Alongside a recommended document, a list of words can be provided, that illustrates how the recommended document differs to the document at hand.
- In a scenario where a user viewed several documents in a row, the search history can be represented as a graph where the nodes are documents and the vertices are the paths that the approximation algorithm produces. By comparing search histories only by the vertices, similar histories can be found although the documents that produced it may differ. Those foreign histories may contain interesting documents for the user.

TABLE 8. Approximation Process of two documents by the same author but on different topics. The Similarity Column indicates the Cosine Similarity between Y and the Current Approximation. Note that the Approximation was conducted in 600 dimensional Space and then replicated in 100 Dimensional Space.

Iteration	**Vector**	**Similarity @600**	**Similarity @100**
initial	diff := Y - X	-0.01	-0.44
1	diff := diff - ”stock”	0.04	-0.30
2	diff := diff + ”industrial”	0.11	-0.22
3	diff := diff - ”employee”	0.12	-0.24
4	diff := diff - ”fuel”	0.15	-0.20
5	diff := diff - ”diesel”	0.19	-0.22
6	diff := diff - ”non-statistical”	0.20	-0.18
7	diff := diff + ”debt”	0.21	-0.19

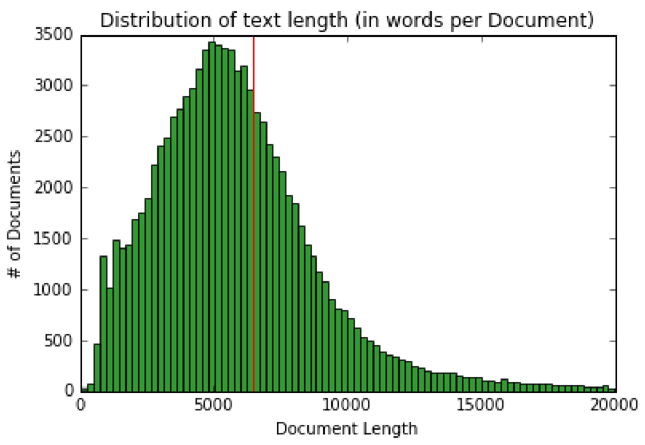

Figure 2. A histogram depicting the length of the documents in Econstor16. The red line marks the mean length of 6476 words per document.

- Given two documents, a user may get recommendations of documents in between (referring to the vector space). To do so, the approximation algorithm is not required. But the algorithm can be used to explain to users why they obtained that specific recommendation, by providing a list of words that explains the relation between both documents.

6. Corpus

This section will introduce the Econstor16 corpus. It is based on ZBWs[1] open access service Econstor[2] which is among the largest open access repositories in the field economics.

Besides the plain text of the documents and along with the usual meta information like author, title and publication year there are several other useful information like the number of citations the paper received, language, link to the original PDF file, author assigned keyword and keywords assigned by domain experts based on a controlled vocabulary (see STW[3]). Moreover, there are Econbiz[4] identifiers as well as RePEc[5] identifiers that allow fetching further information from other services.

To date, Econstor serves 108,000 documents to the users. But not all of them allow plain text extraction, which reduces the size of the corpus to 96,000. Since the repository is rapidly growing, this number will keep increasing.

Figure 2 illustrates the length of the documents. The bulk has less than 10,000 words. 77% of the documents are written in English, 19% in German. The remaining 4% are composed out of over 20 languages. Due to legal reasons the corpus can not be provided publicly, but the code that created this corpus is available[6]. It consist of two components that (1) download the documents and the meta information and (2) extract the plain text.

1. http://www.zbw.eu
2. http://econstor.eu/
3. http://zbw.eu/stw/version/latest/about
4. http://www.econbiz.de/
5. http://repec.org/
6. https://github.com/n-witt/EconstorCorpus

7. Conclusion and Future Work

We have presented an algorithm that is able to explain the topical difference between two documents by a list of words that, when added or subtracted to one of the documents, approximates the other. We also found that some words are intuitively suitable whereas others are less informative. Moreover, we introduced a new corpus that was derived from a dataset that is used in production, which makes the corpus relevant for practically orientated research.

Along at least two dimensions we make suggestions for improvement:

1) **Execution Speed:** At every iteration the current approximation is compared to all words in the vocabulary. Although this can be done in parallel, this is computationally-intensive task. In order to mitigate this issue, the vocabulary should be kept small only containing words that are useful for the task at hand.
   During the semi-automatic testing, it was noticed that candidates from iteration n are likely to reappear in iteration $n + 1$. Therefore, it might be reasonable to pick more than one word on each iteration. This reduces the computational effort on the cost of accuracy.
2) **Accuracy:** Since the purpose of the approach described in this paper is to find words that a human would consider suitable, it's crucial to achieve a high topically accuracy (the word ”that” may be a good approximator in terms of increasing the similarity, but is poor in explaining the topical distance). To overcome this issue the words that make up the vocabulary must be well-chosen. It was also noted that multiplying vectors (and thereby stretching or squeezing them) on the path can increase the similarity and hence accelerate the convergence. But this leaves us with the question of how to interpret these factors.

Furthermore, the impact of the dimensionality of the vector space remains unclear. The impact of this variable on the prior mentioned factors needs to be investigated.

We also introduced the Econstor16 corpus. It contains the plain text of over 90,000 documents, accompanied by several meta information such as author names, title, citation count, STW-descriptors and more. That makes the corpus appropriate for several text ming experiments. But since the text extraction process that was used is error-prone, some text representations are inaccurate or even faulty.

Acknowledgement. The presented work was developed within the EEXCESS project funded by the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement number 600601.

References

[1] Z. S. Harris, “Distributional structure,” *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[3] J. Chang, S. Gerrish, C. Wang, J. L. Boyd-Graber, and D. M. Blei, “Reading tea leaves: How humans interpret topic models,” in *Advances in neural information processing systems*, 2009, pp. 288–296.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in Neural Information Processing Systems 26*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119, word2vec.

[5] Q. V. Le and T. Mikolov, “Distributed representations of sentences and documents,” *arXiv preprint arXiv:1405.4053*, 2014.

[6] A. M. Dai, C. Olah, and Q. V. Le, “Document embedding with paragraph vectors,” *arXiv preprint arXiv:1507.07998*, 2015.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[8] A. Nenkova and K. McKeown, “A survey of text summarization techniques,” in *Mining Text Data*. Springer, 2012, pp. 43–76.

[9] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of Machine Learning Research*, vol. 9, no. 2579-2605, p. 85, 2008.