Semantic embedding for information retrieval

Shenghui Wang and Rob Koopman

OCLC Research, Leiden, The Netherlands
{shenghui.wang, rob.koopman}@oclc.org

Abstract. Capturing semantics in a computable way is desirable for many applications, such as information retrieval, document clustering or classification, etc. Embedding words or documents in a vector space is a common first-step. Different types of embedding techniques have their own characteristics which makes it difficult to choose one for an application. In this paper, we compared a few off-the-shelf word and document embedding methods with our own Ariadne approach in different evaluation tests. We argue that one needs to take into account the specific requirements from the applications to decide which embedding method is more suitable. Also, in order to achieve better retrieval performance, it is worth investigating the combination of bibliometric measures with semantic embedding to improve ranking.

1 Introduction

Many applications such as document clustering/classification and information retrieval depend on the semantic similarity between the words or documents. However capturing a truthful and computable semantics of words or documents is not easy.

1.1 Word embedding

Much research has adopted the notion of *Statistical Semantics* [5,17] based on the assumption of "a word is characterized by the company it keeps" [4] or in Linguistics the *Distributional Hypothesis* [6,16]: words that occur in similar contexts tend to have similar meanings. Various distributional semantic models have been proposed to represent (embed) words in a continuous vector space where semantically similar words are mapped to nearby points ('are embedded nearby each other').

There are two main categories of approaches [2]: 1) global co-occurrence count-based methods (e.g., Latent Semantic Analysis) which compute the statistics of how often some word co-occurs with its neighbour words in a large text corpus, and then use dimension reduction methods (e.g., Singular-Value Decomposition, Random Projection) to map these count-statistics down to a small, dense vector for each word, and 2) local context predictive methods (e.g. neural probabilistic language models) which directly try to predict a word from its neighbours or vice versa in terms of learned small, dense embedding vectors.

Both categories have their own disadvantages. Count-based methods leverage global statistics but perform poorly on the word analogy evaluation that suggests the resulting semantic space might not be optimal. Context predicting models including the recently popular Word2Vec method [12,13] perform better with the word analogy tasks but they only focus on separate local context windows and fail to take advantage of the vast amount of repetition in the whole corpus. Researchers have been trying to combine the benefits of both categories. For example, GloVe [14] proposed a global log-bilinear regression model to learn vector representations from the ratio of the co-occurrence probabilities of two words, instead of the co-occurrence probabilities themselves.

1.2 From word embedding to document distance

Abundant research has proposed to calculate the document distances based on words or, more recently, word embeddings. The traditional Latent Semantic Indexing directly works on the term-document matrix and generates vector representation of documents. Doc2Vec [11] extends Word2Vec to learn the correlations between words and documents which embeds documents in the same vector space where the words are embedded. Other research calculates the document distances from word embeddings without embedding documents themselves, such as using the Word Mover's Distance [10]. We also applied a simplistic approach of taking the weighted average of word embeddings to represent documents and performed reasonably well in a topic delineation challenge [8,9].

A desirable property of embeddings obtained by all the methods is computable similarity: the similarity between two words or two documents correlates with the cosine of the angle between their vectors. The exact type of relation that a high similarity value indicates is not easily specified. Literature has shown such relations could be hypo/hypernymy, co-hyponymy, meronymy, synonymy, antonymy, morphological similarity, or simple collocation. Even so, such measurable similarity/relatedness is useful for many applications including clustering, information retrieval, context visualisation, etc.

In this paper, we compare the word embedding results of the off-the-shelf Word2Vec [12,13] and GloVe [14] with our own Ariadne approach [8,9]. Also, we compare the neural-network-based document embedding method Doc2Vec with Ariadne in a specific information retrieval use case. For the word analogy test, the local context predictive Word2Vec method outperforms Ariadne. However, Ariadne performs equally well as Doc2Vec does in the information retrieval task, and is actually able to rank the target results higher in the result lists.

We argue that different embedding methods have its own characteristics and one needs to take into account the specific requirements from the applications to decide which method is more suitable. We also suggest that the combination of the embedding techniques and some biblimetric measures might help improve the retrieval performance.

2 Dataset

We extracted the metadata of nearly 27 million Medline articles from WorldCat.[1] These articles were published from 1809 to present, with more than half published in the past 20 years. The possible metadata fields are title, abstract, subject, author, affiliation, journal, citation, article type, publication date, language, etc. However, not all the fields were equally populated, for example, 88% of the articles have subjects, 60% with abstract and only 10% with citations. Also 17% of the articles are catalogued as non-English publications.

We considered each article as a sequence of words from title and abstract plus the entity tokens such as `subject:eczema_physiology` and `author:diefenbach_wc`. The whole Medline dataset contains nearly 5 million unique words and entity tokens. These article sequences are processed by Word2Vec/Doc2Vec and GloVe which produce word and document embeddings.[2] For our own Ariadne approach, we consider two words or entity tokens co-occur if they occur in the metadata of the same article. The resulting co-occurrence matrix is further processed by Random Projection [1,7] which embeds words and entity tokens in a vector space. We do not use any text segments preprocessing, such as tokenising, stemming, or stopword-filtering, etc.

3 Experiment I: Compare different word embedding techniques

We applied the Ariadne approach, Word2Vec and GloVe on the complete Medline dataset. We used the python implementation [15] of Word2Vec. In order to choose the optimal parameters, we ran different parameters over a subset of 1 million randomly selected articles. After evaluating against the Semantic–Syntactic Word Relationship test set [13], we chose the following parameters: sg=0 (using the Continuous Bag of Word model), hs=0 (no hierarchical softmax), negative=10 (using negative sampling), size=500 (the dimensionality of the vectors), min_count=10 (ignore words or entity tokens with total frequency lower than 10), window=10 (the maximum distance between the current and predicted word within a sequence).[3] For GloVe, we kept the default settings of the C implementation that is available at `http://nlp.stanford.edu/projects/glove/`.

3.1 Different models lead to different embeddings

Tables 1 gives the top 10 most *similar* words based on Ariadne, Word2Vec and GloVe trained on the whole Medline dataset. Different models give different lists

[1] `http://www.worldcat.org/`

[2] Our purpose is to compare how different embedding techniques perform on the same dataset. To be fair to our Ariadne method, we did not train Word2Vec and GloVe on top of the existing word embeddings pre-trained on the Google News dataset.

[3] More detailed explanation of these parameters could be found at `https://radimrehurek.com/gensim/models/word2vec.html`.

of the most related words. The Word2Vec identifies more hypo/hypernymys or co-hyponymys, such as *ankle* to *knee*, *amphibian* to *frog*, etc. However, it also introduces some distantly related words, or at a more abstract level, such as *turtle* and *salamander* to *frog*. The Ariadne seems to capture more contextual or attribute-related relatedness, such as *flexion* to *knee*, *efficacy* to *treatment*, etc. Different types of similarity/relatedness are mixed in the Ariadne results, which in a sense forms a tighter contextual view around the search term. The results from GloVe seem to be in-between Ariadne and Word2Vec. Looking at these different top related lists, it is not straightforward to judge which one is the best.

3.2 Word analogy evaluation

In [13], the authors proposed to measure the embedding accuracy based on questions such as which word is the most similar to *Italy* in the same sense as *Paris* is similar to *France*, i.e., the Semantic–Syntactic Word Relationship test or, simply, word analogy test. The task is to answer such questions by searching for the word closest to the vector $X = vector(\text{“}Paris'') - vector(\text{“}France'') + vector(\text{“}Italy'')$. Only the closest word is taken into account, so synonyms were considered as mistakes. In the test set,[4] there are in total 19,558 questions. However, not all the questions were used during the evaluation. The questions which contain a word which is not one of the top 30,000 most frequent words were ignored. In the end, the accuracy was measured based on 4175 questions.[5] Table 2 gives the performance of three methods.

Word2Vec and GloVe both perform pretty well in the word analogy test, with GloVe slightly worse. We were actually surprised by the comparable results with those reported in [13], given such a domain-specific corpus. This suggests that a general-purposed test set is still applicable in a domain specific use case, but it might not help to evaluated what is really important for the domain.

Ariadne is based on Random Projection over the co-occurrence matrix, and computationally much more efficient than Word2Vec and GloVe. With a single thread, Ariadne is twice as fast as Word2Vec using 16 threads. Although Ariadne does provide reasonable top related words (shown in Tables 1), it does not do well with this word analogy test. This is of course a known limitation of the count-based methods, as described in Section 1.1. However, we will show in the next section that one type of evaluation is not enough to measure a method in general.

[4] Available at `https://storage.googleapis.com/google-code-archive-source/v2/code.google.com/word2vec/source-archive.zip`

[5] This may indicate the inappropriateness of this test set which was originally for the evaluation of the word embeddings trained on the general Google News corpus. The questions are general enough though, such as *capital–country*, *man–woman*, *adjective–adverb*, which should also be valid in the Medline corpus. In the future we will investigate more domain-specific test sets as suggested in [3].

Word	Model	Top 10 most similar words
frog	a	sartorius, frogs, rana, liagushki, temporaria, liagushek, catesbiana, sartorii, amphibian, caudiverbera
	w	toad, bullfrog, amphibian, rana, frogs, turtle, bufo, salamander, caudiverbera, newt
	g	rana, frogs, amphibian, toad, temporaria, bullfrog, laevis, xenopus, ridibunda tadpoles
brain	a	brains, cortical, cortex, forebrain, cerebellum, neocortex, neuronal, neuroanatomical, neural, limbic
	w	cerebral, cerebellum, cns, brains, brainstem, hippocampus, forebrain, cerebrum, cortical, neocortex
	g	cerebral, brains, cns, nervous, neuronal, cerebellum, hippocampus, neurological, cortex, cerebrum
knee	a	knees, tibiofemoral, femorotibial, tibial, kneeling, joint, malalignment, flexion, unicompartmental, tka
	w	hip, ankle, elbow, knees, shoulder, joint, patellofemoral, wrist, patellar, acl
	g	knees, joint, hip, ankle, osteoarthritis, arthroplasty, joints, cruciate, elbow, flexion
depression	a	depressive, mood, nondepressed, subsyndromal, depressed, anxiety, dysthymia, phq, hamilton, anxious
	w	depressive, anxiety, insomnia, mdd, psychopathology, psychosis, ptsd, mood, suicidality, mania
	g	depressive, anxiety, depressed, mood, psychiatric, symptomatology, psychological, affective, psychopathology, emotional
insulin	a	hyperinsulinemia, glucose, hyperglycemia, insulinopenia, euglycemia, normoglycemic, hypoinsulinemia, insulinemia, nondiabetic, glycemia
	w	glucagon, gh, leptin, glucose, gip, hyperinsulinemia, adiponectin, niddm, glp, hyperinsulinemic
	g	glucose, diabetes, glucagon, mellitus, fasting, hyperglycemia, leptin, igf, diabetic, hyperinsulinemia
treatment	a	treated, treat, therapy, treating, efficacy, discontinued, received, discontinuation, clinical, option
	w	therapy, treatments, treating, monotherapy, pharmacotherapy, management, chemotherapy, prophylaxis, intervention, therapeutic
	g	treated, treatments, therapy, treating, therapeutic, effective, further, treat, with, results
vitamin	a	vitamins, vit, hydroxyvitamin, hypovitaminosis, vitd, cholecalciferol, calcidiol, supplements, supplementation, ergocalciferol
	w	vitamins, vit, vitamine, hypovitaminosis, hydroxyvitamin, avitaminosis, cholecalciferol, vitamina, folate, selenium,
	g	vitamins, supplementation, dietary, folic, tocopherol, supplements, ascorbic, deficiency, hydroxyvitamin, d3

Table 1. The top 10 most similar words according to Ariadne (a), word2vec (w), and GloVe (g)

Method	Accuracy(%)	Training time (seconds)	#CPU Thread
Ariadne	1.6	15,020	1
Word2Vec	62.7	38,364	16
GloVe	53.6	22,680	16

Table 2. Performance on the whole Medline dataset, trained on a server with two Intel XEON E5-2670 processors and 256G memory.

4 Experiment II: Document embedding for IR

Here we compare the Doc2Vec method [11] and Ariadne in the context of information retrieval. We applied both Doc2Vec[6] and Ariadne to generate the document embedding. For each document, Ariadne computes the weight average of the embeddings of all the words and entity tokens that occur in its metadata. The weight is adapted from the tf-idf of the word or entity token.

Similarly, the 16-threaded Doc2Vec ran six time slower than the single-threaded Ariadne. We now evaluate these document embeddings in a specific information retrieval use case.

IR use case: evidence-based medicine guidelines Evidence-based medicine guidelines is an easy-to-use collection of clinical guidelines for primary and ambulatory care linked to the best available evidence. They need to be continuously updated in order to follow the latest developments in clinical medicine and bring evidence into practice. After the updating process, same statements often stay but often with new references substituting the old ones. Some old references could also stay if there is no new literature supporting the statement better. Some references serve multiple statements too.

Here is an example:

Statement	There are no indications to suggest that a skin-sparing mastectomy followed by immediate reconstruction leads to a higher risk of local or systemic recurrence of breast cancer.
Old references (pmid)	9142378, 1985335
New references (pmid)	9142378, 9694613, 18210199

Our *use case* is therefore, given a statement which is present in both old and new guidelines, can the system find the new references to replace the old ones? It is essentially an Information Retrieval problem: given a query which consists a

[6] Doc2Vec is an extension of Word2Vec. We used the Python implementation available at `https://radimrehurek.com/gensim/models/doc2vec.html`. We set dm as 0 to use the distributed bag of words training (PV-DBOW) algorithm and the rest parameters the same as those for Word2Vec.

sentence and a few example articles, to find more articles which were published lately and matches the query the best.

Researchers at VU Amsterdam already compared pairs of medical guidelines of four diseases, namely breast cancer (2004&2012), hepatitis C (2006&2013), lung cancer (2005&2014), and ovarian cancer (2003&2013). They identified 29 statements which are present in both guidelines, each with an old and new reference lists.

- 29 statements (16 breast caner, 4 hepatitis C, 4 lung cancer, 5 ovarian cancer)
- 103 (96 unique) source articles, 156 (145 unique) target articles, in total 180 unique articles
- 66 articles are in both source and target lists, so the average baseline recall is 45.8%
- These articles were published between 1984 and 2012.

We randomly selected 1 million Medline articles which were written in English, with abstract and published between 1984 and 2012. Together with the 180 articles collected from the guidelines, these articles are our test dataset to which the Doc2Vec and Ariadne were applied. We carried out the information retrieval tasks as following: for each statement,

1. Combine the statement with the metadata of each old reference article as a query
2. Set the range of the publication year as between 1984 and 2012
3. Get the top n most similar candidates of each query
4. Re-ranked the candidates from all the individual queries
5. Measure the precision and recall with or without a cutting length (n)

Each individual query returns top n most related candidates, and all the candidates are then ranked by their highest similarity score if they occur in more than one returned list. With the "no cutting" option, all the candidates join the precision/recall calculation, i.e. the final returned list is normally longer than n. The "with cutting" option only takes the top n into account, i.e., the final returned list has the exact length of n.

Figure 1 gives the precision and recall at different n and Table 3 gives the detailed results when $n = 100$. As Figure 1 shows, the Doc2Vec without cutting gives the best recall. When top 100 candidates were returned by each individual queries, jointly, 143 out of 156 target articles were successfully returned, including 61 our of 90 new articles which were not in the old guidelines. The Ariadne also performs pretty well. With or without cutting, both methods perform almost the same when n is small. The difference gets bigger when n gets bigger and the result list is not cut. Figure 1 and Table 3 show that, with the "cutting" option, the difference between Doc2Vec and Ariadne is less though.

This gives a rather different comparison result from that in Table 2. Clearly failed in the word analogy test, Ariadne undoubtedly works very well in this retrieval task. Given the high efficiency of Ariadne, we think it could be a practical solution for many information retrieval applications.

As Table3 shows, when "without cutting", the average length of the resulting lists of Ariadne is shorter than that of Doc2Vec. This again indicates that Ariadne returns more concentrated results (i.e., different individual queries for the same statement returns more overlapping results). Ariadne is actually able to rank the target articles significantly higher in the final list.

Method	Cutting	Average recall (%)	Rank	Length
doc2vec	no	93.3	31.6± 57.9	238.8 ± 95.2
	yes	82.6	15.8±24.3	100
ariadne	no	86.3	19.3 ±38.6	191.6±61.3
	yes	80.2	10.8±17.6	100

Table 3. When top 100 candidates were returned

Table 4 gives the rank distribution of the target articles for Doc2Vec and Ariadne. Nearly 60% of the target articles are ranked within top 10 for both methods and 80% of the target articles are kept within top 100. Less than 8% of target articles are ranked below 1000 for the both methods.[7] Ariadne misses a few more target articles within top 1000. This is probably due to the fact that Ariadne returns more focused results, and those which are related at a more distant level could not be captured by Ariadne.

The ranking by the highest similarity score is to some extent effective but of course not optimal, because when using the "cutting" option, some target articles are discarded. More delicate ranking method is worth investigation. Certain bibliometrics measures could help here. Actually nearly 90% target articles are published in the journals with the highest impact factors. Highly cited articles or written by influential authors could be other indicators to improve the ranking. In the future, we will further test other compositionality methods for document embedding and investigate the possibilities of combining biblimetric measures with semantic embedding to improve the ranking.

5 Conclusion

In this paper, we compared a few word and document embedding techniques in different evaluation tests. For the word analogy test, the local context predictive Word2Vec method outperforms Ariadne which is a simple count-based method applying Random Projection over the global co-occurrence matrix. However,

[7] One of these missed articles is the introduction article of a general guideline, titled as "Introduction: Diagnosis and management of lung cancer: ACCP evidence-based clinical practice guidelines (2nd Edition)." There is no abstract in this WorldCat record (`http://worldcat.org/oclc/173782151`). Without some full-text analysis, it is difficult to get this record ranked higher, compared to other articles addressing the exact topic in the statement.

Rank		#TotalTarget
≤ 10	Doc2Vec	91
	Ariadne	89
≤ 100	Doc2Vec	127
	Ariadne	123
≤ 200	Doc2Vec	137
	Ariadne	131
≤ 1000	Doc2Vec	149
	Ariadne	143

Table 4. Rank distribution of the target articles

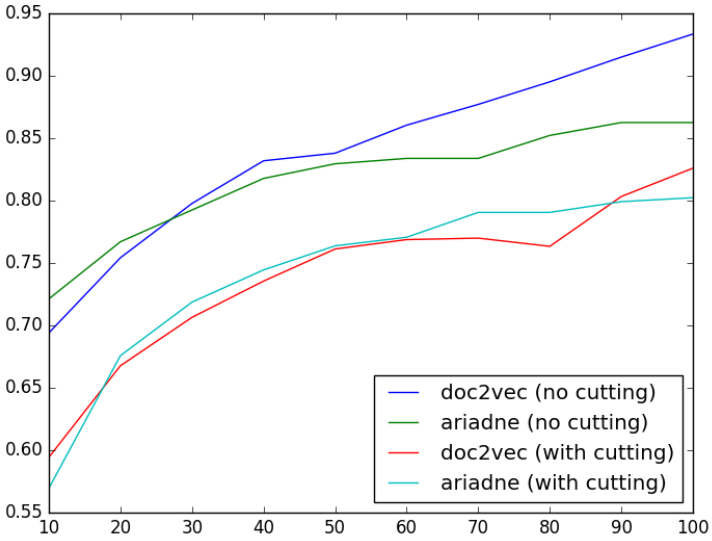
Ariadne performs equally well as Doc2Vec does in a specific information retrieval task, and is able to rank the target results higher.

We argue that one has to take into account the specific requirements from the applications to decide which embedding method is more suitable. For example, one should use Word2Vec to provide hypo/hypernymy recommendations. If the application is to provide a more contextual overview of a word, Ariadne might be a better choice. If the efficiency is more important, a simple weighted average can already get us pretty far in terms of document retrieval.

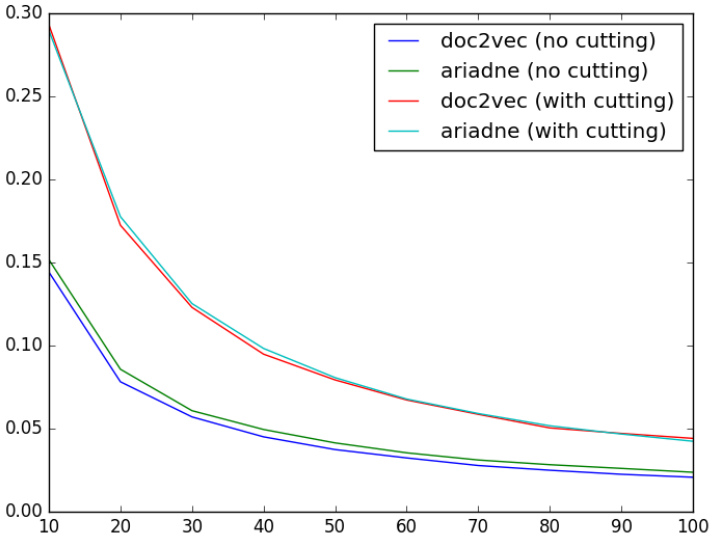
References

1. Dimitris Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003.
2. Marco Baroni, Georgiana Dinu, and German Kruszewski. Don't count , predict ! A systematic comparison of context-counting vs . context-predicting semantic vectors. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics.*, pages 238–247, 2014.
3. Trevor Cohen and Dominic Widdows. Empirical distributional semantics: Methods and biomedical applications. *Journal of Biomedical Informatics*, 42(2):390–405, 4 2009.
4. John R. Firth. A synopsis of linguistic theory 1930-1955. *Studies in Linguistic Analysis*, pages 1–32, 1957.
5. George W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. Statistical semantics: Analysis of the potential performance of keyword information systems. *Bell System Technical Journal*, 62(6):1753–1806, 1983.
6. Z. Harris. Distributional structure. *Word*, 10(23):146–162, 1954.
7. William Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Math.*, 26:189–206, 1984.
8. Rob Koopman, Shenghui Wang, and Andrea Scharnhorst. Contextualization of Topics - Browsing through Terms, Authors, Journals and Cluster Allocations. *Proceedings of ISSI 2015 Istanbul. 15th International Society of Scientometrics and Informetrics Conference, Istanbul, Turkey, 29th June to 4th July 2015*, pages 1042–1053, 2015.

9. Rob Koopman, Shenghui Wang, and Andrea Scharnhorst. Contextualization of topics – browsing through the universe of bibliographic information. In J. Gläser, A. Scharnhorst, and W. Glänzel, editors, *Same data – different results? Towards a comparative approach to the identification of thematic structures in science, Special Issue of Scientometrics*. 2017.
10. Matt J Kusner, Yu Sun, Nicholas I Kolkin, and Kilian Q Weinberger. From Word Embeddings To Document Distances. *Proceedings of The 32nd International Conference on Machine Learning*, 37:957–966, 2015.
11. Qv Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. *International Conference on Machine Learning - ICML 2014*, 32:1188–1196, 5 2014.
12. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
13. Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *Proceedings of the International Conference on Learning Representations (ICLR 2013)*, pages 1–12, 1 2013.
14. Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014.
15. Radim Rehurek and Petr Sojka. 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, 2010. ELRA.
16. Magnus Sahlgren. The distributional hypothesis. *Rivista di Linguistica*, 20(1):33–53, 2008.
17. Warren Weaver. Translation. In W.N. Locke and D.A. Booth, editors, *Machine Translation of Languages*, pages 15–23. Cambridge, Massachusetts: MIT Press, 1955.



(a) Average recall



(b) Average precision

Fig. 1. Comparison performance at rank n