k-NN Embedding Stability for word2vec Hyper-parametrisation in Scientific Text

Amna Dridi[1], Mohamed Medhat Gaber[1], R. Muhammad Atif Azad[1], and Jagdev Bhogal[1]

School of Computing and Digital Technology, Birmingham City University, Millennium Point, Birmingham B4 7XG, United Kingdom

Abstract. Word embeddings are increasingly attracting the attention of researchers dealing with semantic similarity and analogy tasks. However, finding the optimal hyper-parameters remains an important challenge due to the resulting impact on the revealed analogies mainly for domain-specific corpora. While analogies are highly used for hypotheses synthesis, it is crucial to optimise word embedding hyper-parameters for precise hypothesis synthesis. Therefore, we propose, in this paper, a methodological approach for tuning word embedding hyper-parameters by using the stability of k-nearest neighbors of word vectors within scientific corpora and more specifically Computer Science corpora with Machine learning adopted as a case study. This approach is tested on a dataset created from NIPS[1] publications, and evaluated with a curated ACM hierarchy and Wikipedia Machine Learning outline as the gold standard. Our quantitative and qualitative analysis indicate that our approach not only reliably captures interesting patterns like *"unsupervised_learning is to kmeans as supervised_learning is to knn"*, but also captures the analogical hierarchy structure of Machine Learning and consistently outperforms the 61% sate-of-the-art embeddings on syntactic accuracy with 68%.

Keywords: word embedding, word2vec, skip-gram, hyper-parameters, k-NN stability, ACM hierarchy, Wikipedia outline, NIPS.

1 Introduction

Word embeddings (WEs) are a class of natural language processing techniques that represent individual words as real-valued vectors in a predefined vector space. They were first introduced in the 1990s using statistical approaches [2, 8] with vectors computed as rows of lexical co-occurrence [8] through matrix factorization [2].

However, interest in WEs has recently skyrocketed and has found many applications. The surge in interest is due both to popularity of neural networks [1] which exploit WEs for NLP tasks, and the success of low-dimensional embeddings like *word2vec* [12] and *GloVe* [16]. Due to their ability to detect semantics and meanings of words, WEs have been used as features in a variety of applications, such as document clustering [5] and classification [22], Linear Discriminant

[1] Conference on Neural Information Processing Systems

Analysis (LDA) [17], information retrieval [10], named entity recognition [20], sentiment analysis [19], and semantic discovery [21].

Typically, the reported research work that used WEs as features computed their vector representations with a default or arbitrary choice of embedding hyper-parameters. Examples of these hyper-parameters include *vocabulary size* and type (single words or phrases), *vector dimensionality*, that is, the length of the vector representations of words, and *context size* which is the span of words in the text that is taken into account - both backwards and forwards - when iterating through the words during model training. However, as this paper will show, these hyper-parameters of word embedding vectors are crucial to the prediction performance as they directly affect the accuracy of the generated analogies. Given that analogies can be used in hypotheses synthesis, consequently an accurate analogy will led to a precise hypothesis. For example, *"decision tree"* is a component of *"ensemble"* and *"decision tree"* is a *"classifier"*. So, by analogy any classifier should be a component of ensemble.

This work concerns then hyper-parametrisation of WEs in a domain-specific context with varying vocabulary sizes, and represents a gap in knowledge on present practice of WEs. The considered topic is a key practical issue while learning WEs, and is so chosen because not only is the literature on learning embedding hyper-parameters rather limited [6, 14], it does not offer a method to efficiently set these hyper-parameter values. Stimulated by this shortcoming, the work we present in this paper lies within the context of word embedding hyper-parametrisation for domain-specific use. The studied problem domain is *scientific literature* and more specifically *Machine Learning* literature, which is a subcategory of literature on *Computer Science*. The choice of a scientific domain is motivated by an increasing interest in knowledge extraction from scholarly data and scientific text to understand research dynamics and forecast research trends. Since WEs have proved their ability to capture relational similarities and identify relationships in textual data without any prior domain knowledge, this work does not need to justify the use of WEs for knowledge extraction from scientific literature; instead, the presented work is a methodical approach to setting the hyper-parameters of WEs for scientific knowledge extraction.

The motivation here is to deeply understand the embedding behavior within scientific corpora which is quite different to other corpora in terms of word distributions and contexts. For instance, the term *"learning"* appears obviously in the context of education in newspapers corpora; however, *"learning"* appears in a completely different context within *Computer science*. Therefore, WEs for scientific text are worth investigating.

There have been some efforts to integrate WEs in the scientific domain [3, 7, 22]; however, these efforts do not study learning the hyper-parameters suitable for a scientific text, and instead use either arbitrary or default settings (Mikolov's settings [11]).

In this research work, we aim to fill this gap. We hypothesise that by devising an approach for setting hyper-parameters of WEs in the scientific domain, this study adds a deep understanding of the sensitivity of embeddings to hyper-

parametrisation. To make our point, we propose using the stability of k-nearest neighbors of word vectors as a measure to set the hyper-parameters – mainly vector dimensionality and context size – of word vector embeddings; moreover, we propose using common-sense knowledge from the *ACM hierarchy*[2] and *Wikipedia outline of Machine learning*[3]. As a result, this work adds breadth to the debate on the strengths of using WEs for knowledge extraction from scientific text. To the best of our knowledge, the proposed work represents the first attempt to methodically set WEs hyper-parameters in a scientific domain.

We list the major contributions of this work as follows: (i) we propose the stability of k-nearest neighbors of word vectors as an objective to measure while learning word2vec hyper-parameters, (ii) we enhance the standard skip-gram model by bigrams using *word2phrase* – that attempts to learn phrases by progressively joining adjacent pairs of words with a '_' character – as a method for corpus augmentation, (iii) we create an analogy dataset for the *Machine Learning* by manually curating ACM hierarchy and Wikipedia outline of *Machine Learning*, and (iv) we evaluate our work quantitatively and qualitatively on a dataset comprising of abstracts published in the NIPS conference. Our embedding detected interesting semantic relations in *Machine Learning* such as "unsupervised_learning is to kmeans as supervised_learning is to knn". The obtained results are therefore both promising and insightful.

The rest of the paper is organised as follows. Section 2 summarises the existing approaches on word embedding hyper-parametrisation and gives an overview on work that attempted to integrate word embedding in scientific domains. Section 3 presents our methodology and how we employ stability of k-nearest neighbors to optimise word2vec hyper-parameters. Section 4 describes the NIPS dataset we have used, the analogy dataset we have created from ACM hierarchy and Wikipedia as gold standard, presents and discusses results. Finally, in section 5 we conclude and draw future directions.

2 Related Work

Word embedding methods depend on several hyper-parameters that have crucial impact on the quality of embeddings. For this reason, Mikolov et *a*l. [12, 13] and Pennington et *a*l. [16] –the inventors of the popular low-dimensional embedding word2vec and GloVe, respectively – have deeply studied the optimisation of the embedding parameters, mainly the vector dimension and the context size. The performance of the embeddings has been measured based on *w*ord similarity that uses cosine distance between pairs of word vectors to evaluate the intrinsic quality of such word representations, and *w*ord analogies that capture fine-grained semantic and syntactic regularities using vector arithmetic. The optimal parameters have been obtained through training on large Wikipedia and Google News corpora. But, no evidence was given for generalisation of these parameters to any other corpus with a general or specific topic and guarantee the performance

[2] https://dl.acm.org/ccs/ccs_flat.cfm
[3] https://en.wikipedia.org/wiki/Outline_of_machine_learning

of embeddings. However, most of the work using WEs relies on these parameters as the default ones.

Unlike work that uses default settings, literature on learning embedding hyper-parameters is relatively short [6, 14]. Levy and Goldberg [6] followed Mikolov et *a*l. [11, 12] and Pennington et *a*l. [16] and trained their embeddings on general topic using Wikipedia corpus. They basically tested their model with different vector dimensions and different window sizes aiming to study the impact of syntactic contexts – that are derived from automatically produced dependency parse-trees – on detecting functional similarities of cohyponym nature. While Miñarro-Giménez et *al.* [14] trained their word embeddings on a domain-specific corpus of medical data in order to study the ability of word embeddings (word2vec) to capture linguistic regularities on the medical corpora. Similar to the previous work, Miñarro-Giménez et *a*l. trained their word2vec embeddings with different parameter settings, *i*.e., dimensionality of vector space, context size, and different model architectures, *i*.e., continuous bag-of-words (CBOW) and Skip-gram (SG) [11], and simultaneously compared the relationships identified by word2vec with manually curated information from National Drug File – Reference Terminology ontology as a gold standard using word similarity and word analogies in order to evaluate the effectiveness of word2vec in identifying properties of pharmaceuticals and medical relationships. The obtained results (49% accuracy) revealed the unsuitability of word2vec for applications requiring high precision like medical applications. While this research work seems interesting mainly with its appeal to setting hyper-parameters for domain-specific word embeddings, it does not bring a defined method to efficiently set these parameters.

Leading on from the aforementioned observation, the work we present in this paper lies within the context of word embedding hyper-parametrisation for domain-specific use. The proposed domain to investigate is the *s*cientific domain and more specifically *C*omputer Science with *M*achine learning, as a case study.

There have been some efforts to integrate word embeddings in the scientific domain [3, 7, 22] for clustering scientific documents based on their functional structures [7] or for identifying problem-solving patterns in scientific text [3] or for paper-reviewer recommendation [22]. All the previous research work integrated word embeddings as features for their learning algorithms using either arbitrary or default settings (Mikolov's settings [11]). However, none of them has focused on training the embeddings and methodologically setting the hyper-parameters suitable for scientific text.

To the best of our knowledge, the proposed work represents the first attempt to methodologically set word embeddings hyper-parameters in the scientific domain.

3 Methodology

This study focuses on word2vec hyper-parameter optimisation applied to scientific publications, i.e., how to tune the hyper-parameters that have the largest impact in the prediction performance and what are the adoptable techniques

to test the potential of word embeddings for identifying relationships from unstructured scientific text. Accordingly, the *k-NN algorithmic stability* is adopted to investigate the marginal importance of hyper-parameters of *Skip-Gram* architecture in a scientific setting. This allows us to identify three hyper-parameters, namely *vocabulary subsampling*, *vector dimensionality* and *context size* which can significantly affect the embedding performance. In this study, we use the popular variant word2vec architecture *Skip-Gram* as it is consistently yielded superior results comparing to *CBOW* architecture [11].

3.1 The Skip-Gram model

Previous results reported in the literature have shown that *Skip-Gram* [11] model does not only produce useful word representations, but it is also efficient to train. For this reason, we focus on it to build our embeddings for scientific text in this study. The main idea of *Skip-Gram* is to predict the *context* c given a word w. Note that the *context* is a window around w of maximum size L. More formally, each word $w \in W$ and each context $c \in C$ are represented as vectors $\overrightarrow{w} \in \mathbb{R}^d$ and $\overrightarrow{c} \in \mathbb{R}^d$ respectively, where $W = \{w_1, \cdots, w_V\}$ is the words vocabulary, C is the context vocabulary, and d is the embedding dimensionality. Recall that the vectors parameters are latent and need to be learned by maximising a function of products $\overrightarrow{w} \cdot \overrightarrow{c}$.

More specifically, given the word sequence W resulted from the scientific corpus, the objective of *Skip-Gram* model is to maximise the average log probability: $L(W) = \frac{1}{V}\sum_{i=1}^{V}\sum_{-l\leq c\leq l, c\neq 0} logProb(w_{i+c}|w_i)$ where l is the context size of a target word. *Skip-Gram* formulates the probability $Prob(w_c|w_i)$ using a softmax function as follows: $Prob(w_c|w_i) = \frac{\exp(\overrightarrow{w_c}\cdot\overrightarrow{w_i})}{\sum_{w_i\in W}\exp(\overrightarrow{w_c}\cdot\overrightarrow{w_i})}$ where $\overrightarrow{w_i}$ and $\overrightarrow{w_c}$ are respectively the vector representations of target word w_i and context word w_c, and W is the word vocabulary. In order to make the model efficient for learning, the hierarchical softmax and negative sampling techniques are used following Mikolov et *al.* [11].

Word embedding vectors learned with *Skip-Gram* can be used for computing word similarities. The similarity of two words w_i and w_j can simply be measured with the inner product of their word vectors, namely $similarity(w_i, w_j) = \overrightarrow{w_i}\cdot\overrightarrow{w_j}$. Recall that *cosine distance* is the measure used to calculate the similarity between embedding vectors $\overrightarrow{w_i}$ and $\overrightarrow{w_j}$ as following:

$$similarity(w_i, w_j) = cosineDistance(\overrightarrow{w_i}, \overrightarrow{w_j}) = \frac{\overrightarrow{w_i}\cdot\overrightarrow{w_j}}{\|\overrightarrow{w_i}\|\cdot\|\overrightarrow{w_j}\|} \tag{1}$$

As discussed in the Introduction section, we aim to evaluate the representation capability of WEs within scientific text using word similarities as a pivot to stabilise the embedding hyper-parameters.

Skip-Gram model uses a target word w to predict the surrounding window of context words. It weights nearby context words more heavily than more distant context words [11, 12]. Results of word2vec training are sensitive to parametrisation. To this end, the aim of hyper-parameter optimisation is to find a tuple

of hyper-parameters that yields an optimal model minimising the *loss function* for negative samples $(w, \bar{c})$ where $\bar{c}$ does not necessarily appear in the context of w. This *loss function* $\mathscr{L}$ is defined as follows: $\mathscr{L} = -log(\sigma(\overrightarrow{w} \cdot \overrightarrow{c})) - \sum_{k=1}^{n} log(\sigma(-\overrightarrow{w} \cdot \overrightarrow{\bar{c}_k}))$ where σ is the sigmoid function. For each pair (w, c), the *Skip-Gram* model forms n negative pairs $(w, \bar{c}_k)_{k \in \{1,\cdots,n\}}$ by sampling words that are more frequent than some threshold θ with a probability: $Prob(c) = \frac{freq(c)-\theta}{freq(c)} - \sqrt{\frac{\theta}{freq(c)}}$ where $freq(c)$ represents the frequency of the word c.

word2vec has different hyper-parameters, but *sub-sampling* that automatically affects the *corpus size*, *vector dimensionality* and *context window* are described by the developers of word2vec [11, 12] as the most important ones for achieving good results. Consequently, in this study we focus of these hyper-parameters to produce a distributed representation of words in scientific text and evaluate the quality of embeddings in a domain-specific vocabulary.

Sub-sampling: vocabulary size It has been proved in the literature [11, 12, 16] that word2vec embedding quality increases as the corpus size increases. This is expected as longer corpus typically produce better statistics. Following on from this premise, we aim to investigate the role of vocabulary size in generating accurate embeddings for scientific text.

Unlike previous work that intuitively increments the vocabulary size by combining corpus, we propose to use the same corpus trained in two different ways that led to different vocabulary sizes. First, we train word2vec with unigrams. Second, we train the model with bigrams by using *word2phrase* – defined by Mikolov et *al.* [12] – that learns phrases by progressively joining adjacent pairs of words with an '_' character. Additionally, we sub-sample the frequent words on two steps which result into two different vocabulary sizes. Firstly, we remove all stop words and highly frequent academic words appearing in all publications. Secondly, we restrict the vocabulary to words that occur at least 10 times in the scientific corpus. According to Mikolov et *al.* [11], this sampling has proved to work well in practice. It accelerates learning and significantly improves the accuracy of the learnt embedding vectors, as it will be shown in Section 4.

Vector dimensionality and context window The optimisation of *vector dimensionality* and *context window* parameters is supposed to be very crucial to achieve accurate results. The quality of embeddings increases with higher dimensionality under the assumption that it increases together with the amount of training data. But after reaching some point, the marginal gain will diminish [11].

The window size hyper-parameter corresponds to the span of words in the text that is taken into account, backwards and forwards when iterating through the words during model training. Similarly to the vector dimensionality hyper-parameter, the larger window size results in more topicality. Nevertheless, after a certain point, the marginal gain decreases.

Due to the sensitivity of these hyper-parameters and since hyper- parametrisation is generally known to be data and task dependent [4], we expect optimal hyper-parameter setting to be different for scientific text. Thus, we propose to

study the marginal importance of word2vec hyper-parameters defined above using the *stability of k-nearest neighbors* of word vectors based on word similarities computed with *cosine distance* (Equation (1)) between embedding vectors.

***k*-NN stability for word2vec hyper-parametrisation** Stability is an important aspect of a learning algorithm. It has been widely used in clustering problems [18] to assess the quality of a clustering algorithm. Also, it has been applied in high-dimensional regression [15] for training parameter selection. Analogously and considering that word embedding presents high-dimensional word representations that led to word clusters, we propose to apply the *k-nearest neighbors* to tune the hyper-parameters of word2vec. k-NN is used to cluster similar words based on their cosine similarities.

The basic idea of word embedding stability is the following: embedding quality inevitably depends on tuning hyper-parameters defined previously, namely *vector dimensionality* and *context window*. If we choose accurate values of the tuning hyper-parameters, then we expect that the k similar words to a target word w from different embeddings should be similar. Specifically, we propose to fix one hyper-parameter, tune the second one by trying different values and training the model for each value. After each training, word similarities are computed and k-nearest neighbors words are defined. The *k-NN stability* is defined as a simple overlap rate of similar words resulted from two embeddings with different settings.

$$stability = \frac{\mathcal{S}^w_{E_h} \cap \mathcal{S}^w_{E_{h'}}}{k} \times 100 \tag{2}$$

where $\mathcal{S}_{E_h}$ and $\mathcal{S}_{E_{h'}}$ are two sets of similar words to a target word w resulted respectively from two embeddings E_h and $E_{h'}$ with different hyper-parameter values. k is the number of nearest neighbors to w given by the cosine similarity. In this study, k is set to 5. This choice is motivated by our aim to keep the word similarities as fine-grained as possible in order to evaluate the quality of word2vec within scientific text.

3.2 Scientific linguistic regularities and analogies

word2vec embeddings gain their success from their ability to capture syntactic and semantic language regularities. Surprisingly, they characterise each relationship by a relation-specific vector offset [13]. For example, the famous analogy *"king is to queen as man is to woman"* is encoded in the vector space by the vector arithmetic *"king - man + woman = queen"*. More specifically, the word analogy task aims at answering the question *"man is to woman as king is to — ?"* given the two pairs of words that share a relation ("man:woman", "king:queen") where the identity of the fourth word ("queen") is hidden.

Motivated by this ability of word2vec to identify relationships and capture analogies in textual data without any prior domain knowledge, we evaluate this ability in a domain-specific corpus, namely, scientific publications. Our aim is to assess as to what extent word2vec is able to correctly answer analogical questions

in scientific text given the complexity of scientific language comparing to natural language.

The scientific word analogy we adopt is to query for scientific regularities captured in the vector model through simple vector subtraction and addition. More formally, given two pairs of words $(a : b')$ and $(b : b')$, our aim is to answer the question *(a is to a' as b is to —?)*. Thus, the vector of the hidden word b' will be the vector $(a' - a + b)$, suggesting that the analogy question can be solved by optimising:

$$\arg\max_{b' \in W}(similarity(b', a' - a + b)) \tag{3}$$

where W is the vocabulary and *similarity* is the cosine similarity measure defined in Equation (1).

This task is challenging for scientific language as no gold standard is available to evaluate the efficiency of word2vec in identifying linguistic regularities on unstructured scientific text, unlike existing work that use either the gold standard defined by Mikolov et *al.* [13] for general natural language tasks or predefined ontologies like NDF-RT ontology[4] for medical domain. To overcome this problem, we manually curate relationships related to *machine learning* research area from the *ACM hierarchy* and the *Wikipedia Machine Learning outline*, and we define a test set of analogy questions as *semantic questions* following the relation described above. The semantic questions are formed based on the hierarchical tree structure of both the ACM and Wikipedia outline that led to different *"Parents-Children"* relationships. For example, *"supervised_learning"* and *"unsupervised_learning"* are considered two parents for the two children *"classification"* and *"clustering"* respectively. Accordingly, the analogical question should be *"classification to supervised_learning is as clustering to —?"* To correctly answer the question, the model should identify the missing term with a correspondence counted as a correct match by finding the word *"unsupervised_learning"* whose vector representation is closest to the vector *("supervised_learning" - "classification" + "clustering")* according to the cosine similarity. Recall that for the specificity and complexity of scientific language and respecting the interchangeability of scientific terms, instead of using the exact correspondence as the correct match, we adopt an approximate correspondence that considers an answer as correct if it belongs to the 10 nearest words given by cosine similarity in order to guarantee the applicability of our embeddings in scientific text. This is applied only for *semantic questions*. However, for *syntactic questions*, we adopt an exact correspondence. For example, the syntactic question *"classifier to classifiers is as forest to —?"* is considered correctly answered if and only if the word *"forests"* is the closest to the vector *("classifiers" - "classifier" + "forest")* according to the cosine similarity.

In addition to the *semantic questions* manually curated from ACM and Wikipedia, we define *syntactic questions* which are typically analogies about verb tenses/forms and singular/plural forms of nouns, in order to test the ability of word2vec to capture the syntactic regularities of scientific language.

[4] National Drug File -Reference Terminology

4 Experimental Evaluation

4.1 NIPS dataset: Description and Vocabulary Setup

To evaluate word embedding for scientific language, we used a subset of 2789 papers in the area of Machine Learning, published in NIPS (Neural Information Processing Systems) between 2012 and 2017. The dataset is publicly available on Kaggle[5] and contains information about papers, authors and the relation papers-authors. We used the papers database that defines six features for each paper: the *id*, the *title*, *the event type*, i.e., poster, oral or spotlight presentation, the *PDF name*, the *abstract* and the *paper text*.

The dataset needs to be pre-processed before being used for training the embedding model, since word2vec is very sensitive to vocabulary granularities like punctuation, lowercase, stop words, *etc.* which have a direct impact on the quality of generated word embeddings. After removing all punctuations and lowercasing the corpus, the pre-processing has the following steps:

(i) We removed stop-words using Stanford NLP stop word list[6] enriched by a list of 170 academic stop words that we defined from common academic vocabulary like "*introduction, abstract, table, figure, etc.*", (ii) We constructed bag of words where words are either *unigrams* used for standard word2vec training or *bigrams* used for *word2phrase* learning. The two settings resulted into different vocabulary sizes $|W_{unigrams}| = 35k$ and $|W_{bigrams}| = 96.7k$, and (iii) We discarded less frequent words that appear less than 10 times in the vocabulary in order to accelerate learning. This led to a different vocabulary size $|W_{downsampled}| = 57k$.

4.2 word2vec training details: Hyper-parameters optimisation

As described in Section 3, *k-NN stability* was used to optimise the word2vec hyper-parameters, namely, *vector dimensionality* and *context window size*.

Vector dimensionality. *k-NN stability*, with $k = 5$, was used to evaluate the influence of the vector dimensionality hyper-parameter using vector models generated with 20, 30, 50, 100, 150, 200, 300 and 500 dimensions, *skip-gram* architecture and three different vocabulary sizes as described in Section 4.1.

In Table 1, we show the results of *k-NN stability* values that vary vector length and vocabulary size. word2vec model was initially learned with 20-vector dimension. This trained model was used as a seed setting to start computing *k-NN stability*. More specifically, *k-NN stability* at 30-vector dimension was computed based on the 20-vector dimension following Equation (3) and respectively each k-NN stability value is computed based on the results generated by the previous dimensionality setting. The reported results correspond to the stability average of the top 100 frequent words (unigrams and bigrams) in the vocabulary.

It has been clearly seen from the three vocabulary sizes that the stability increases considerably as the dimensionality increases. But after reaching some

[5] https://www.kaggle.com/benhamner/nips-papers/data

[6] github.com/stanfordnlp/CoreNLP/blob/master/data/edu/stanford/nlp/patterns/surface/stopwords.txt

point, it diminishes or becomes slightly invariant. For instance, for the *unigram vocabulary*, k-NN stability reached 67% with 100-dimension vector performing good results comparing to 30 and 50 dimensions. However, it remains basically steady with a slight increase of 1% at 200-dimension. This increase is not remarkable enough to consider 200-dimension better than 100-dimension since a higher dimension of the vectors implies a bigger size of the resulting vector model and more training time. Then, we notice that the stability decreases with larger dimensions (300 and 500). Consequently, these results suggest that 100-dimension vector consistently yielded better stability with unigrams vocabulary.

Similarly, *bigrams vocabulary* shows a substantial improvement in k-NN stability from 30-dimension to 200-dimension with 68%. Then, it increases slightly with 300 and 500 dimensions with a 1% gain. Hence, for this vocabulary, we can fix the optimal dimensionality value to 200. Interestingly, the stability results of the *unigram vocabulary* and the *bigram vocabulary* confirm the hypothesis that vector dimensionality and the amount of training data should be increased together to have better results. As a matter of fact, 100 has shown to be the better vector length for *unigram vocabulary* of $35k$ size, while 200 is better for *bigram vocabulary* of $96.7k$ size. On the other hand, by looking at the stability

Table 1: k-NN stability for vector dimensionality optimisation

	D30	D50	D100	D150	D200	D300	D500
unigrams	42%	53%	67%	67%	68%	66%	65%
bigrams	51%	47%	56%	64%	68%	70%	71%
downsampled bigrams	58%	61%	65%	73%	81%	n/a	n/a

values at high dimensions (300 and 500), we noticed the excess in stability of *bigrams vocabulary* comparing with *unigrams vocabulary*. This is comprehensibly justified by three facts: *(i)* this confirms the hypothesis that word2vec model quality increases as corpus size increases [14], *(ii)* this proves that n-gram enhanced skip-gram model performed better than regular skip-gram based only on unigrams, *(iii)* this confirms the specificity of scientific language and mainly the *Computer Science* area that contains an important number of bigrams like *"machine-learning"*, *"artificial-intelligence"*, *etc.*

Based on these findings, mainly *(i)* and *(ii)*, we ignored the 300 and 500 dimensions for training the *downsampled vocabulary* which is resulted from downsampling the *bigram vocabulary* as the vocabulary size is obviously smaller ($57k$). It is worthy to note that this downsampling improved the training speed and most importantly made the k-NN stability values more important with 81% at 200-dimension while it was 68% with *bigram vocabulary* at the same dimension. This was expected as downsampling makes the word representations significantly more accurate [12].

Overall, the k-NN stability results obtained through vector dimensionality optimisation show that bigram enhanced skip-gram model performs better with scientific language, 200 is the optimal vector length for the used dataset and the *downsampled bigram vocabulary* significantly outperforms the two other vocabu-

laries in term of k-NN stability and computation time. Note that for all word2vec training rounds with different vocabularies and different vector dimensionalities, the hyper-parameter *window context* was set to 5, the default window size value provided by *gensim*[7].

Window context Similarly to the setting followed to optimise vector- dimensionality, k-NN stability was adopted to find the optimal window size for the used scientific corpus in this study. Building on previous results, the trained vocabulary used is the *downsampled vocabulary* and the *vector dimensionality* is 200. word2vec embeddings were generated with skip-gram model and 7 different window sizes ranging from 2 to 8. word2vec was initially trained with a context window of size 2 as a starting point. Then k-NN stability was computed respectively based on the previous embedding results. Figure 1 presents the values of k-NN stability that vary context window size. It is clearly seen from the figure that the optimal window size is 6 with a stability of 70% for the used scientific corpus. Our results confirm the fact that larger window size results in more topicality and accordingly better accuracy of word representations. However, the marginal gain decreases after a certain point. Overall, our findings

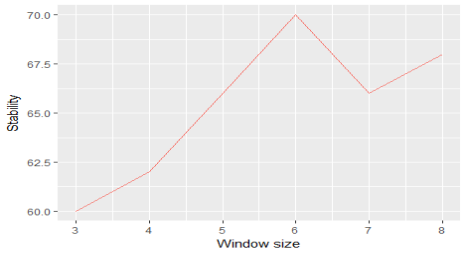


Fig. 1: k-NN stability for context window size optimisation

show that the combination of 200-vector dimension with context window of size 6 and downsampled bigram vocabulary proved to be the best configuration of skip-gram word2vec model. Additionally, the proposed *k-NN stability* – based on word similarity as embedding properties, that we adopted in this study to optimise the word2vec hyper-parameters for scientific text – confirms all hypotheses related to word embeddings supported in the literature and even goes beyond them by giving a standard way to be sure about the stability of results.

4.3 Analogy evaluation

As described in section 3.2, the word analogy task attempts to query for scientific regularities captured in the embedding model – trained with the previously optimised hyper-parameters – through simple vector subtraction and addition.

The analogy dataset we created contains 1991 analogical questions, divided into 1871*semantic questions* and 120 *syntactic questions*. The *semantic questions*

[7] https://radimrehurek.com/gensim/

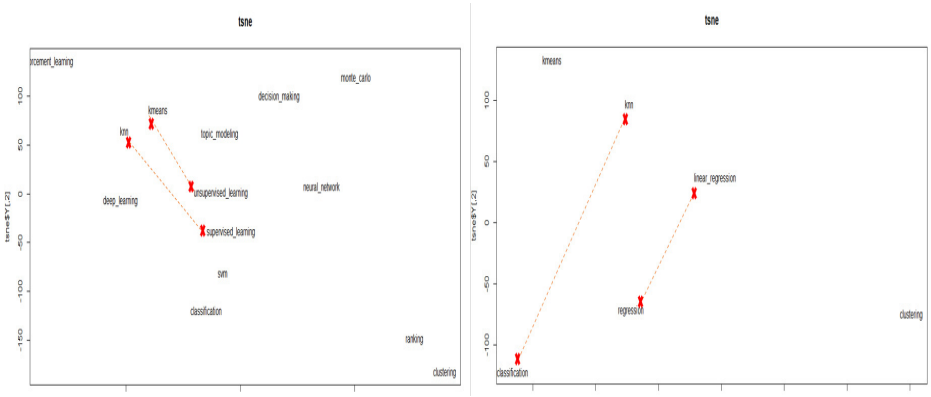
were manually curated from ACM hierarchy (406 questions) and Wikipedia outline of *Machine Learning* (1465 question). The number of relationships generated from Wikipedia are by far greater than the ACM counterpart. This justified by the fact that ACM is more coarse-grained as it covers all the *Computer Science* area, while the Wikipedia outline is a fine-grained hierarchy generated specifically for *Machine Learning* with very detailed algorithms and applications of the area. We remove from the analogy dataset all questions that contain words that do not exist in our vocabulary in order to fairly evaluate embedding analogies. This resulted into 1573 questions (322 ACM questions and 1251 Wikipedia questions). Similarly to *semantic questions*, *syntactic questions* were a manually generated subset that we created from the scientific text using typical analogies about verb tenses/forms and singular/plural forms of nouns, in order to test the ability of word2vec to capture the syntactic regularities of scientific language. The number of questions is relatively small due to our aim to only preliminarily test the word2vec ability to cover syntactic scientific regularities that do not differ from natural language, while the *semantic questions* do. That is why we focus more on these latter. Our analogy dataset is available online for more reproducibility and any further use by researchers[8].

For evaluating our embeddings in capturing linguistic regularities and analogies, we performed both quantitative and qualitative analysis.

Quantitative analysis In this analysis, we empirically evaluate our proposed bigram-enhanced word2vec model trained with hyper-parameters experimentally tuned. Our goal of these experiments is two-fold. First, we aim to evaluate whether our hyper-parametrisation method of word2vec is useful for resulting embeddings able to cover linguistic regularities and analogies within scientific text. Second, we aim to assess whether word embeddings are worth using in domain-specific vocabularies such as the scientific one.

To do so, we computed the *accuracy* of word embeddings to answer the semantic and syntactic questions following the methodology detailed in Section 3.2. For semantic questions, 50 out of 322 ACM questions were correctly answered with an accuracy of 15.52% while 75 Wikipedia questions were correct out of a subset of 413 questions from the 1251 questions in the dataset, with an accuracy of 18%. The difference in accuracy between ACM and Wikipedia questions was expected as Wikipedia relationships were more detailed and covered *Machine Learning* names of algorithms and applications that widely occur in the vocabulary, while ACM was more coarse-grained. Although, the accuracy of both of them is very low. This could be justified by three different reasons. First, the corpus size we used is relatively small with only $57k$ while it has been shown that word2vec quality increases as corpus size increases. For instance, Mikolov et *al.* [12] trained their model on a corpus of $1B$ and obtained a semantic accuracy of 61%. Second, the used NIPS dataset is about very recent publications (between 2012 and 2017). So that, the vocabulary is more probably about recent topics and accordingly recent *Machine Learning* vocabulary, i.e., names of algorithms and applications might gain more frequencies in the text than the

[8] https://github.com/AmnaKRDB/Machine-Learning-Analogies



(a) r_1 vector offset (b) r_2 vector offset

Fig. 2: Vector offsets examples of Machine Learning semantic relationships

old ones, which in turn would highly affect the word representations, at the time when ACM hierarchy or Wikipedia outline are time-independent and contain generic *Machine Learning* vocabulary. Third, the scientific language is complex and does not contain explicit and accurate relationships as natural languages does. For instance, 'accuracy' and 'error rate' in the machine learning literature are used in similar contexts, despite having opposite semantics.

For all these reasons, the semantic accuracy of word embedding within the used scientific corpus is considered modest. But, it is promising as it is interpretable and improvable on one hand. On the other hand, it reveals challenges about scientific word embedding. More specifically, it is worth investigating the convergence and divergence of some *Machine Learning* algorithms and applications over time which consistently affects the word representations. Interestingly, it is challenging to find a suitable way to train and evaluate word embeddings in such dynamic vocabularies.

For syntactic questions, we computed the accuracy across the 120 questions we defined. Interestingly, we found 82 questions out of 120 correctly answered with an accuracy of 68%. This result is interesting despite the small size of our vocabulary. It outperforms the syntactic accuracies of Mikolov et *al.* [12] which reached 61% with $1B$ vocabulary and 300-dimension vector.

Qualitative analysis The embedding we learned revealed interesting patterns in *Machine Learning* vocabulary through relation-specific vector offsets. For instance, it captured different semantic relationships mapping *Machine Learning* techniques and related algorithms such as r_1 *"unsupervised_learning is to kmeans as supervised_learning is to knn"*, and r_1 *"classification is to knn as regression is to linear_regression"*. We illustrate these patterns by plotting word vector representations with *t-distributed stochastic neighbor embedding (t-SNE)* [9] as a qualitative way to evaluate our embeddings following Yao et *al.* [21]. Figure 2a and Figure 2b show the t-SNE representations of r_1 and r_2 respectively.

In addition to the t-SNE visualisation used to qualitatively evaluate the accuracy of our embeddings to detect interesting patterns in the scientific text, we suggested to evaluate the capability of our model to capture the hierarchy structure *"Parent-Children"*. To do so, we computed and compared similarities between every word *"parent"* and the corresponding words *"children"*. The model is considered accurate if the distances are approximately equal. For instance, the distances between the parent *"supervised_learning"* and its children {*"classification", "regression", "ranking", "cost_sensitive"*} are approximately equal with slight differences as presented here respectively (0.369; 0.241; 0.173; 0.223) similarly to the parent *"unsupervised_learning"* and its children {*"clustering", "dimensionality_reduction", "topic_modeling", "anomaly_detection", "mixture_modeling", "source_separation"*} with approximately similar distances (0.259; 0.307; 0.237; 0.145; 0.145; 0.135; 0.253).

Similarly, we followed the same reasoning to compare the average distances between *"Parents-Children"*. The model is accurate if the average distance between every parent and its children is similar to others parents' average distances. With respect to the example above, we computed the average distance of the parents *"supervised_learning"* and *"unsupervised_learning"* with their corresponding children. And interestingly, we found that the average distances are respectively equal to 0.25 and 0.22 which proves the accuracy of our embedding to detect granularities of scientific text, not only the semantic relationships but also the hierarchical structure.

5 Conclusions and future work

Despite their popularity in overwhelming state of the art performance in semantic similarity and analogy tasks, word embeddings are still treated as black boxes and uniformly use the hyper-parameters without a methodological setting. From this perspective and aiming to provide a precise hypotheses synthesis, this work addressed word embedding hyper-parametrisation for domain-specific use, namely the scientific domain. By proposing the stability of k-nearest neighbors of word vectors, we were able to methodologically set the hyper-parameters suitable for scientific text. Our method has been validated quantitatively and qualitatively on semantic and syntactic analogies curated from ACM and Wikipedia as gold standard and has proved its effectiveness.

As a short term objective, we plan to apply our method on larger scientific vocabulary, then generalise it on different research areas. For long term objectives, we plan to investigate more settings for word embeddings within the scientific area, aiming to detect trendy and evolving patterns by performing time-aware vocabulary augmentation and sliding windows.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. Journal of Machine Learning Research **3**, 1137–1155 (2003)
2. Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science **41**(6), 391–407 (1990)

3. Heffernan, K., Teufel, S.: Identifying problems and solutions in scientific text. Scientometrics (Apr 2018)
4. Hutter, F., Hoos, H., Leyton-Brown, K.: An efficient approach for assessing hyperparameter importance. In: 31st Int. Conf. on Machine Learning. pp. 754–762 (2014)
5. Kusner, M.J., Sun, Y., Kolkin, N.I., Weinberger, K.Q.: From word embeddings to document distances. In: 32Nd Int. Conf. on Machine Learning. pp. 957–966 (2015)
6. Levy, O., Goldberg, Y.: Dependency-based word embeddings. In: 52nd Annual Meeting of the Association for Computational Linguistics. pp. 302–308 (2014)
7. Lu, W., Huang, Y., Bu, Y., Cheng, Q.: Functional structure identification of scientific documents in computer science. Scientometrics **115**(1), 463–486 (Apr 2018)
8. Lund, K., Burgess, C.: Producing high-dimensional semantic spaces from lexical co-occurrence. Behavior Research Methods, Instruments, & Computers **28**(2), 203–208 (1996)
9. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research **9**, 2579–2605 (2008)
10. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA (2008)
11. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR **abs/1301.3781** (2013)
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: 26th Int. Conf. on Neural Information Processing Systems. pp. 3111–3119 (2013)
13. Mikolov, T., Yih, W.t., Zweig, G.: Linguistic regularities in continuous space word representations. In: HLT-NAACL. pp. 746–751 (2013)
14. Miñarro-Giménez, J.A., Marín-Alonso, O., Samwald, M.: Applying deep learning techniques on medical corpora from the world wide web: a prototypical system and evaluation. CoRR **abs/1502.03682** (2015)
15. Nicolai Meinshausen, Peter Bhlmann: Stability selection. Journal of the Royal Statistical Society **72**(4), 417–473 (2010)
16. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: EMNLP. vol. 14, pp. 1532–1543 (2014)
17. Petterson, J., Buntine, W., Narayanamurthy, S.M., Caetano, T.S., Smola, A.J.: Word features for latent dirichlet allocation. In: Advances in Neural Information Processing Systems, pp. 1921–1929 (2010)
18. Rinaldo, A., Singh, A., Nugent, R., Wasserman, L.: Stability of density-based clustering. Journal of Machine Learning Research **13**(1), 905–948 (Apr 2012)
19. dos Santos, C.N., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: COLING. pp. 69–78 (2014)
20. Turian, J., Ratinov, L., Bengio, Y.: Word representations: A simple and general method for semi-supervised learning. In: 48th Annual Meeting of the Association for Computational Linguistics. pp. 384–394 (2010)
21. Yao, Z., Sun, Y., Ding, W., Rao, N., Xiong, H.: Dynamic word embeddings for evolving semantic discovery. In: 11th ACM Int. Conf. on Web Search and Data Mining. pp. 673–681 (2018)
22. Zhao, S., Zhang, D., Duan, Z., Chen, J., Zhang, Y.p., Tang, J.: A novel classification method for paper-reviewer recommendation. Scientometrics pp. 1–21 (Mar 2018)