Projection Aléatoire Non-Négative pour le Calcul de Word Embedding

Behrang Q. Zadeh[1] Laura Kallmeyer[1] Aurelie Herbelot[2]
(1) DFG SFB 991, Universität Düsseldorf, Düsseldorf, Germany
(2) CLIC-CIMEC, University of Trento, Italy
zadeh@phil.hhu.de., kallmeyer@phil.hhu.de, aurelie.herbelot@cantab.net

RÉSUMÉ

Dans cet article nous proposons une nouvelle méthode pour le calcul de word embeddings en utilisant des projections aléatoires. Notre approche est telle que des méthodes de pondération comme positive pointwise mutual information (PPMI) peuvent être appliquées après la construction de notre modèle véctoriel, i.e., après avoir déjà réduit la dimensionalité. Les word embeddings peuvent alors être tranferés d'une manière efficace et avec une performance de calcul supérieure vers des espaces sémantiquement discriminants. De plus, l'approche se distingue par une mise-à-jour facilitée de l'espace vectoriel et une interopérabilité augmentée. Nous évaluons notre méthode par rapport à plusieurs tâches sémantiques. Nous montrons qu'elle donne des résultats comparables à ceux des meilleures méthodes appliquées à des corpus monolingues.

ABSTRACT

Non-Negative Randomized Word Embedding

We propose a word embedding method which is based on a novel random projection technique. We show that weighting methods such as positive pointwise mutual information (PPMI) can be applied to our models after their construction and at a reduced dimensionality. Hence, the proposed technique can efficiently transfer words onto semantically discriminative spaces while demonstrating high computational performance, besides benefits such as ease of update and a simple mechanism for interoperability. We report the performance of our method on several tasks and show that it yields competitive results compared to neural embedding methods in monolingual corpus-based setups.

MOTS-CLÉS : Word Embedding, Projection Aléatoire.

KEYWORDS: Word Embedding, Word Vectors, Random Projections, Lexical Semantics.

1 Introduction

Word embedding techniques have become a cornerstone of several modern NLP systems. From a methodological perspective, two broad categories of embedding algorithms are commonly identified and used. Following the terminology proposed in Baroni *et al.* (2014), these are A) the classic *count models*, which employ the chain of processes of collecting co-occurrence frequencies in high-dimensional vectors followed by weighting and dimensionality reduction processes (see Turney & Pantel, 2010) ; and, B) the more recent *neural predictive techniques* in which the process of learning word embeddings is modelled as a (type of convex) optimisation task, and implemented using a kind of neural network—e.g., Mikolov *et al.* (2013b), Pennington *et al.* (2014), and so on. Evidently, both methods capture statistical information about word usage and can be explained using vector space mathematics (Levy & Goldberg, 2014).

In this paper, we ask which features an ideal vector-based representation of word meaning should have. We maintain that an ideal model must A) show human-like behaviour on linguistic tasks of interest ; B) have low dimensionality for 'easy' manipulation ; C) be efficiently acquirable from large corpora to allow for fast experimentation ; D) be linked to a transparent model, so that linguistic and computational hypotheses and experimental results can be systematically analysed and explained ; and E) be easy to update (e.g., allow the addition of new context elements or target entities and the removal of old ones, as required in the analysis of frequently-updated text data, such as text streams from social media).

Despite their successes, both count and predictive models fall short of some of these desiderata. Due to the heavy-tailed distribution of words' co-occurrences (e.g., Figure 1a), count-based models often fail on A and B, unless they embrace weighting and dimensionality reduction processes. Weighting and dimensionality reduction processes, however, are carried out over high-dimensional vectors and thus impose high computational complexity (C). In addition, once count-based models are weighted and compressed, they are fixed and cannot be updated without running those expensive steps anew (E). Even with these provisions, in some tasks, the performance of these models lags behind that of predictive models (A), unless a variety of parameters is added to the basic model (Levy *et al.*, 2015), in which case the model becomes more difficult to interpret (D).

On the other hand, despite their competitive performance (A), neural-predictive models are relatively slow and resource-intensive to train and involve a time-consuming parameter-tuning process (C). Further, neural embeddings are difficult to interpret (D) : there have been a number of publications dedicated to understanding these methods and/or questioning their lack of interpretability, e.g., (Fyshe *et al.*, 2015; Li *et al.*, 2015). Simply put, we do not know exactly what sort of information the dimensions of these models convey. Even worse, this lack of interpretability results in the lack of *interoperability* : There is no guarantee that the comparison of embeddings obtained from different runs of these methods or/and on different corpora will be meaningful. [1]

To find a middle ground solution for the above-mentioned problem, we introduce a new technique for generating low-dimensional embeddings using *non-negative* random projections. The construction of models using our method does not require training but simply generating random vectors and adding these random vectors together. In contrast to previous random-projection-based methods, our proposed embeddings can undergo weighting transformations such as PPMI *after* their construction. We argue that this model retains the main strengths of both count and predictive models, while smoothing out some of their respective disadvantages. In § 2, we set the problem and propose our solution in § 3. In § 4, we examine parameters of our method and report its performance on several tasks. We conclude in § 5.

2 Problem Statement

We regard the problem of producing word embeddings as an operation over raw word-context co-occurrences. To derive an embedding matrix $\mathbf{P}$, a co-occurrence matrix $\mathbf{C}$ undergoes a set of

1. This is also true for count models compressed using a matrix factorisation technique.

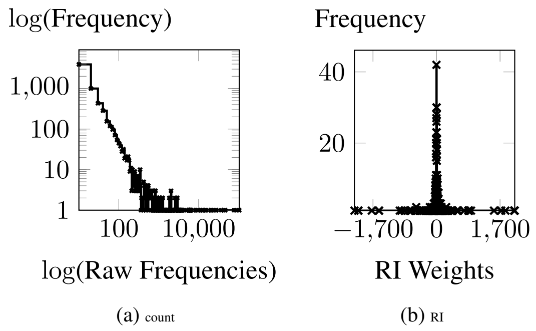

FIGURE 1: A histogram of the distribution of frequencies of weights built from 1+1 context-windows for the lemma ‘abandon’ in the UKWaC corpus : long-tailed distribution in count models and a standard asymptotic Gaussian in RI.

transformations $\mathbf{T}$:

$$\mathbf{C}_{p\times n} \times \mathbf{T}_{n\times x} = \mathbf{P}_{p\times x}. \tag{1}$$

The given process in Equation 1 maps p vectors of dimensionality n to an x-dimensional space $\mathbf{P}$ (often, but not necessarily, $x \neq n$ and $x \ll n$). $\mathbf{T}$ can be a composition of several transformations, e.g., a *weighting* process $\mathbf{W}$ followed by a *dimensionality* reduction $\mathbf{R}$, i.e., $\mathbf{T}_{n\times x} = \mathbf{W}_{n\times n} \times \mathbf{R}_{n\times x}$. The two broad categories of embedding techniques mentioned in § 1 differ in the way that they compute and implement $\mathbf{T}$.

In the so-called count models, $\mathbf{T}$ is often the combination of PPMI weighting (for increasing the discriminatory power of models) and singular value decomposition (SVD) truncation (for dimensionality reduction). The PPMI weight for a component $c_{xy} \in \mathbf{C}$ is (Church & Hanks, 1990; Turney, 2001; Bouma, 2009) :

$$ppmi(c_{xy}) = \max(0, \log \frac{c_{xy} \times \sum_{i=1}^{p}\sum_{j=1}^{n} c_{ij}}{\sum_{i=1}^{p} c_{iy} \times \sum_{j=1}^{n} c_{xj}}). \tag{2}$$

SVD truncation involves solving a least squares problem : computing ℓ_2 distances between column vectors. Evidently, weighting process and SVD truncation of a large and high-dimensional model is cumbersome and time consuming, and they restrain future updates of the model.

In predictive models, $\mathbf{T}$ is computed using neural embedding techniques. Simply put, $\mathbf{T}$ is derived by solving an optimization problem defined by the *objective function* of the neural net. Although neural networks can excel in task such as finding a (near-)optimum $\mathbf{T}$ and thus $\mathbf{P}$, their usage (similar to PPMI+SVD) can be computationally expensive and their training is time-consuming. Moreover, using these methods involves an often complex parameter tuning process, which entails lengthy experiments. The Word2Vec system (Mikolov *et al.*, 2013b), for instance, involves selecting training parameters of ‘negative samples’, ‘learning rate’, ‘subsampling rate’, and ‘smoothing factor’, in addition to the dimensionality of the embeddings and the size of context window. It is difficult to understand which particular combination of these parameters gives the best performance for a given corpus and task.

Apart from the methods mentioned above, $\mathbf{T}$ can be computed using a random projection techniques. In the context of this paper, perhaps the most well-known example of this class of algorithms is random indexing (RI) (Kanerva *et al.*, 2000). In RI, $\mathbf{T}$ is a randomly generated matrix that is very sparse and has an asymptotic standard Gaussian distribution. As a result, in randomized methods such as RI, devising $\mathbf{T}$ and thus deriving $\mathbf{P}$ can be faster and easier than count and predictive methods. Particularly, since $\mathbf{T}$ is determined independently of $\mathbf{C}$, in methods such as RI, $\mathbf{P}$ can be built *incrementally* and be updated and used at any time. Furthermore, interoperability can be achieved by sharing and using similar random projection matrices across tasks and corpora.

Unfortunately, RI and previous random projection-based techniques suffer from low discriminatory power. Literally, RI is a *distance-preserving* randomized dimension reduction technique : the ℓ_2 distances between entities in a $\mathbf{P}$ constructed using RI are (approximately) the same as the ℓ_2 distances between entities in $\mathbf{C}$. Consequently, although the RI method can be much faster and simpler than other embedding techniques, because crude ℓ_2 distances in $\mathbf{C}$ are not sufficient in many tasks, RI lags behind count-based PPMI-weighted and neural embeddings. Note that for RI, the sums of components of row and column vectors in $\mathbf{P}$ are also always zero (given the standard Gaussian distribution of the projections). Hence, RI embeddings cannot be weighted after their construction due to the problem of *division by zero* (e.g., PPMI weighting is not applicable to RI embeddings since the inner summations in Equation 2 are always zero—see Figure 1b). Likewise, note that the use of RI and distance-preserving random projections is limited to the α-normed space that they are designed for (e.g., $\alpha = 2$ for RI). Hence, these methods cannot be applied after using all types of weightings.

In this paper, we propose a method that shows all flexibilities of random projection-based techniques while it displays state-of-the-art performance in semantic similarity assessment tasks. Like RI, we propose a $\mathbf{T}$ that is generated randomly. However, unlike RI, the sums of row and column vectors in our embeddings are always greater than zero. In contrast to RI and count-based models, weighting of our embeddings can take place after their construction. Since vectors resulting from our approach are small (i.e., with a dimensionality of a few hundreds), weighting (e.g., PPMI) can be applied to our models incredibly faster than in classical count-based models. Since our method devises $\mathbf{T}$ independently of $\mathbf{C}$ and skips solving optimization tasks, it is much faster than predictive neural techniques. For similar reasons, models can be built incrementally and updated at any time. Our model is interpretable in the sense that the information each dimension of the model conveys can be elucidated using $\mathbf{T}$: Each dimension of our model corresponds to a bag of words. By the same token, interoperability can be achieved by the simple mechanism of sharing random projection matrices. In the following, we detail our method.

3 The Proposed Embedding Technique

3.1 Construction of Embeddings

In our method, the transformation (i.e., $\mathbf{T}$ in Equation 1) consists of a projection matrix $\mathbf{R}$ which is followed by, recommended but not necessary, a weighting process $\mathbf{W}$, i.e., $\mathbf{T} = \mathbf{R} \times \mathbf{W}$. $\mathbf{R}_{n\times m}$ (for $m \ll n$, e.g., $m = 500$) is simply a sparse randomly generated matrix. The elements r_{ij} of $\mathbf{R}$ have the following distribution :

$$r_{ij} = \begin{cases} 0 & \text{with probability } 1-s \\ 1 & \text{with probability } s \end{cases}, \tag{3}$$

in which s is an extremely small number such that each row vector of $\mathbf{R}$ has, at least, *one* element that is not zero (i.e., $\sum_{i=1}^{m} r_{ji} > 0$ for each row vector $\vec{r_j} \in \mathbf{R}$). Note that the coordinates of nonzero elements are independently and identically distributed (i.i.d.). Given Equations 1 and 3, and using the distributive property of multiplication over addition in matrices, a desired space at a reduced dimensionality m can be built using the familiar two-step procedure of incremental semantic space construction :

Step 1. Each context element is mapped onto an m-dimensional *index vector* $\vec{r}$. $\vec{r}$ is randomly generated such that most elements in $\vec{r}$ are 0 and only a few are *positive integers*.

Step 2. Each target entity/word that is embedded in the model is represented by an m-dimensional *target vector* $\vec{v}$. All the elements of $\vec{v}$ are initially set to 0. For each encountered co-occurrence of this target entity with a context element in an input corpus, $\vec{v}$ is updated by adding the index vector $\vec{r}$ of the context element to it (i.e., $\vec{v} = \vec{v} + \vec{r}$).

This process builds a model directly at the reduced dimensionality m. The first step corresponds to the construction of the randomly generated projection matrix $\mathbf{R}$: Each index vector is a row of $\mathbf{R}$. The second step is an implementation of the matrix multiplication in Equation 1 : Each target vector is an *un*weighted row of $\mathbf{P}$, which is computed in an iterative process. Once these embeddings are obtained, they can undergo a weighting process such PPMI, however, at the reduced dimensionality m instead of n.

3.2 Similarity Measurement and Weighting

Once a model is constructed, if desirable without applying any additional weighting process, similarities between entities are computed using an appropriate measure of correlation coefficient between vectors. The choice of a measure of correlation can be intuitively motivated by examining the role that cosine plays for computing similarities between vectors that are derived from a standard Gaussian random projection. In these models, despite the common interpretation of the cosine similarity as the angle between two projected vectors, cosine can be seen as the product-moment Pearson's correlation between vectors due to the fact that $\mathbf{R}$ and thus the obtained projected spaces have $\mathrm{E} = (0)$. For two vectors $\vec{x}$ and $\vec{y}$ of the same dimension n, Pearson's correlation r is given by :

$$r(\vec{x}, \vec{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \tag{4}$$

in which $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$. Because in these projected spaces $\bar{x}$ and $\bar{y}$ are always 0, Equation (4) can be simplified to $\frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$, which is the same as the cosine definition. Given this explanation, we propose that in Gaussian random projections, cosine gives similarities with respect to the linear correlation between vectors.

To compute similarities between unweighted embeddings obtained from our method, we propose Goodman and Kruskal's γ coefficient (Goodman & Kruskal, 1954) (without ruling out the use of measures of distributional similarities such as Pearson's r, coefficient of determination r^2 and so on). Goodman and Kruskal's γ belongs to a family of *nonparametric* correlation coefficient measures that are known to be more resilient to the presence of outliers and noise, and when dealing with data of non-Gaussian nature (Chen & Popovich, 2002). Since our embedding technique results in models that have non-Gaussian long-tailed distribution, in which some of the coordinates can be uninformative, we expect γ to be a more reliable measure of similarity between vectors than, for instance, Pearson's

r (as confirmed in our experiments). We emphasise that any correlation measure can be regarded for quantifying a degree of relationship between two embeddings.

To compute γ, we define the notions of *concordant* and *discordant* pairs. Given any pairs such as (x_i, y_i) and (x_j, y_j) from two m-dimensional vectors $\vec{x}$ and $\vec{y}$ and the value $v = (x_i - x_j)(y_i - y_j)$, these two pairs are concordant if $v > 0$ and discordant if $v < 0$. If $v = 0$, the pair is neither concordant nor discordant. Let p and q be the number of concordant and discordant pairs in embeddings $\vec{x}$ and $\vec{y}$, respectively. Goodman and Kruskal's γ is given by $\gamma = \frac{p-q}{p+q}$ (Ibid., p. 86.).

As with count models, in our embeddings a weighting process is recommended to eliminate uninformative or irrelevant collected co-occurrence frequencies, and to boost the impact of informative and discriminatory ones. Depending on the task in hand, a range of weighting processes can be applied to our embeddings. Particularly, we suggest using 'dimension-wise estimates of association strength', i.e., a method of normalising raw coordinate values (i.e., the collected frequencies) by the expected and marginal frequencies, e.g., odds ratio, relative risk, and so on, and certainly, PPMI (definitions of these methods are listed in (Evert, 2004, chap. 3, p. 84)). Besides the commonly known function of weighting techniques (distilling statistical information that really matters), applying the above mentioned weighting transformations results in embeddings in which the values of coordinates have a rectified left-truncated Gaussian distribution with a mean much greater than zero.

From a statistical perspective, many weighting processes (and their involved sub-processes such as the log transformation in the definition of PPMI) can be seen as an attempt to stabilize variance of vector coordinates and therefore to make the observations more similar/fit to Gaussian distribution (a practice with long history in many research, particularly in biological and psychological sciences). Hence, for computing similarities between *weighted* embeddings, we anticipate that using a correlation measure such as Pearson's r yields a better result than using γ (as confirmed, to a degree, in our experiments). [2] Overall, we emphasise the importance of considering the bilateral relationship in choosing similarity measure and weighting technique.

Weighting transformations can be composed and applied one after each other to our embeddings. For instance, we observe that an iterative application of PPMI weightings to our embeddings results in a more accurate identification of syntactic regularities/relationships between words, e.g., in tasks such as clustering words by their part-of-speech tags and finding grammatical analogies proposed by (Mikolov *et al.*, 2013a). As stated earlier, since we are dealing with low-dimensional vectors, such compositions can be computed fast and with a low memory footprint. Note that weighting can be done on demand : To accelerate this process, the sum of all the column vectors, which is as small as one m-dimensional vector, can be kept in memory at all time.

4 Empirical Investigations

4.1 Method's Parameters

To use our method, three basic parameters must be set : 1) the way co-occurrences are collected (i.e., the configuration of context window), which is common to any word embedding technique, 2)

2. Note that since the left-truncated Gaussian distributions in our models have a mean much greater than zero, we expect that the truncated tails have a small effect on computed correlations. Also, the length of truncation can be tuned by choosing a value other than zero such as used in (Levy *et al.*, 2015).

	Context Window Size (a+a)					
Method	2+2	3+3	5+5	7+7	9+9	11+11
OM-raw+γ	0.733	0.704	0.643	0.607	0.581	0.566
OM-PPMI+r	0.845	0.835	0.797	0.775	0.75	0.737
RI	0.648	0.629	0.602	0.617	0.583	0.573
Count-raw	0.650	0.638	0.600	0.625	0.600	0.588
Count-PPMI	0.750	0.738	0.662	0.638	0.612	0.600
W2V-CBOW	0.838	0.825	0.825	0.813	0.775	0.763

TABLE 1: The TOEFL Test : Various Context Size.

the dimensionality of embeddings, and 3) the number of positive integer elements in index vectors. To provide a picture of our method's behaviour with respect to these parameters, we report its performance under various settings.

To do so, we begin our empirical investigations with two well-known tasks : (a) the TOEFL synonymy test (Landauer & Dumais, 1997) and (b) the MEN test collection, i.e., a similarity and relatedness test (Bruni *et al.*, 2014). The TOEFL test is a multiple choice task in which, given a target word and four candidate lexical items, the correct synonym for the target must be selected. The test contains 80 questions and results are normally reported in terms of accuracy (i.e., the proportion of questions answered correctly). The MEN test is a collection of 3000 word pairs, associated with similarity/relatedness ratings obtained from human annotators. Evaluation takes the form of a calculating correlation (i.e., Spearman's rank correlation ρ) between the human-induced scores and the scores assigned to word pairs by the system. Note that our aim is to compare our method's performance with other embedding techniques and *not* to obtain the highest achievable performance for these tasks. As for instance, (Bullinaria & Levy, 2012) report that with a careful collection of co-occurrences, PPMI-weighted count model can achieve the highest score for the TOEFL test.

For the experiments in this subsection, we collect co-occurrences from the tokenised preprocessed UKWaC corpus (Baroni *et al.*, 2009). However, we do not use any additional information or processes (i.e., no frequency cut-off for context selection, no syntactic information, no part-of-speech tags, etc.). Reports in this section can be compared with those from Baroni *et al.* (2014), Lapesa & Evert (2014), Tsvetkov *et al.* (2015), and Schnabel *et al.* (2015).

We introduce four baselines, namely, A) a raw count model in which similarities are computed using cosine ; B) the RI method+cosine ; C) high-dimensional PPMI-weighted vectors+cosine ; and D) Word2Vec continuous bag-of-word (cbow) embeddings+cosine. To obtain Word2Vec embeddings, parameters are set to values that have shown the best performance in experiments ran by (Baroni *et al.*, 2014) : dimension 400, 10 negative samples, and 1e-5 for subsampling.

4.1.1 Baselines and Context Window Size

Thorough examination of context window parameters, due to space limitations, goes beyond the scope of this paper. We therefore limit ourselves to context windows of various size that extend symmetrically at both sides of target words. In these experiments, for RI and our method (from now on OM), we report results averaged from 10 independent runs of the algorithms with index vectors of dimensionality 1500. For our method, index vectors contain 4 non-zero elements generated as described by Equation 3. For RI, index vectors have 4 non-zero elements (two +1 and two -1).

Tables 1 and 2 summarize the results obtained for the TOEFL and MEN tests, respectively. Besides

	Context Window Size (a+a)					
Method	2+2	3+3	5+5	7+7	9+9	11+11
OM-raw+γ	0.543	0.541	0.546	0.545	0.545	0.545
OM-PPMI+r	0.712	0.731	0.745	0.745	0.751	0.744
RI	0.331	0.361	0.393	0.422	0.443	0.461
Count-raw	0.326	0.356	0.391	0.419	0.440	0.457
Count-PPMI	0.731	0.748	0.749	0.749	0.750	0.751
W2V-CBOW	0.769	0.757	0.773	0.767	0.769	0.773

TABLE 2: The MEN Test : Various Context Size.

the results for the baselines, we show performance of OM where A) the unweighted vectors are directly used together with the γ for similarity calculation (OM-raw+γ), and B) vectors are PPMI weighted before using Pearson's r for similarity measurement (OM-PPMI+r). For the TOEFL test, OM-PPMI+r shows comparable performance to Word2Vec vectors. Clearly, PPMI weighting of embeddings enhances the performance and OM-PPMI+r outperforms baselines other than Word2Vec by a large margin ; this is true even for OM-raw-γ (i.e., without PPMI weighting). Note that in these experiments, the average performance enhances if we increase the dimensionality of vectors.

The effect of PPMI-weighting on our embeddings is clearer on the MEN test. OM-PPMI+r constantly outperforms OM-raw+γ. Again, OM-PPMI+r performs at the same level as PPMI-weighted count vectors and Word2Vec. Further, OM (with inclusion of the PPMI weighting) performs this task almost twice as fast as Word2Vec and using a smaller amount of memory. [3]

These results also support our earlier claim that RI preserves ℓ_2 distances between vectors. As anticipated, RI shows a performance similar to the raw count vectors. While RI fails in delivering a high performance in these tests, it is important to note that RI, however, attains its goal as a dimensionality reduction method, i.e., ℓ_2 distance preservation. When the dimensionality of RI models increases and the number of non-zero elements set to about 0.001, the results from RI are almost identical to unweighted count vectors, almost in all runs of the algorithm.

4.1.2 The Method's Parameters, its Random Behaviour, and Performance

OM is a randomized method and its performance is influenced by some parameters, namely, the distribution of words in the input corpus (which all models are dependent on) and the random projection matrix configuration—i.e., the hyper-parameters of dimensionality of vectors, and the number of non-zero elements (i.e., m and s in Equation (3)). In any application for a random method such as OM, as explained by (Stein, 2007), the effect of these hyper-parameters must be explained and decided with respect to the amount of 'information loss' that can be tolerated. In distance-preserving methods, the objective is clear : to have least errors (with highest probability) in the estimated distances between projected points ; that is, information loss is interpreted by the amount of distortions in projected distances. Given this interpretation, the hyper-parameters are then explained and set, both in theory and practice—e.g., see (Dasgupta & Gupta, 2003).

When it comes to word embedding methods, however, we are not interested in the sheer distance-preservation ability of methods. Instead, we are aiming for building semantically/linguistically

3. For the UKWaC and the words it contains, context window of size 5+5, learning embeddings using OM took 37 minutes while it took 71 minutes for Word2Vec. Note that in these experiments, Word2Vec was run as a 10-threaded process while OM was run as a single threaded process. Both methods are implemented and tested on the the Java platform. Source codes are available as part of supplement material.

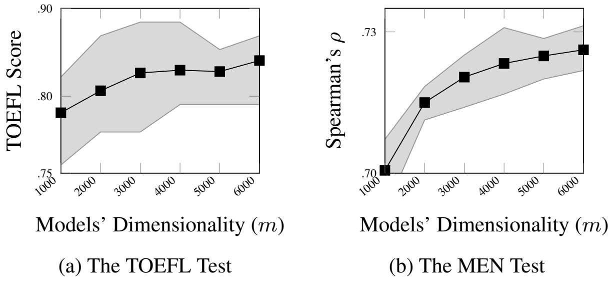

FIGURE 2: Changes in OM+PPMI+r's performance when the dimensionality of models increases and s is the fixed value 0.001. The average performance is shown by the marked line. The margins show the minimum and maximum performance observed in 10 independent executions.

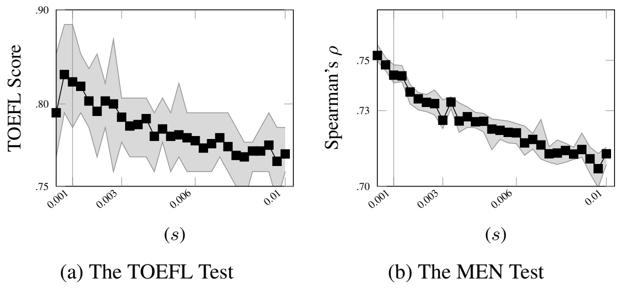

FIGURE 3: Changes in OM+PPMI+r's performance when the dimensionality of models is fixed to $m = 3000$ but s (i.e., the number of positive integers in index vectors) increases.

discriminative models, which is also the case for OM. Taking this into consideration, because we do not have a clear definition of 'information loss' for OM, we are not able to provide theoretical accounts for its behaviour and the role of its hyper-parameters with respect to the 'information loss'. In return, we rely on some empirical results. Consequently, we study OM's behaviour by reporting its performance on the TOEFL and MEN tests under various parameter settings. To keep reports in a manageable size, we focus on OM-PPMI+r configuration and use a context window of size 2+2.

Figure 2 shows the method's performance when the dimensionality m increases while the number of non-zero elements remains at the same ratio of $s = 0.001$. For each m, $1000 \leq m \leq 6000$, the models are built 10 times and the average as well as the maximum and the minimum observed performances in these experiments are reported. From this experiment, we observe that OM follows a similar behavioural pattern to other random projection techniques in the sense that increasing the dimensionality enhances the performance and reduces the variance of task-based performances (and their probabilities). Namely, for the TOEFL and MEN tests, increasing the dimensionality boosts the average performance as well as the worst and the best obtained performances. Evidently, an increase in the average performance entails a low probability for bad performance and a high probability for good performance.

To study the effect of the number of non-zero elements in index vectors, we repeat the above experiments while fixing the dimensionality of vectors to $m = 3000$ and changing s to different

	Similarity Tests											
Vector Set	**YP130**	**RW**	**MTURK287**	**MTURK771**	**MC30**	**MEN**	**WS353**	**RG65**	**TOEFL**	**WS-SIM**	**WS-REL**	**SimLex999**
W2C-B	0.50	0.32	0.66	**0.71**	**0.81**	**0.79**	0.63	**0.83**	0.91	0.75	0.53	0.46
OM-O	0.51	**0.42**	0.60	0.62	0.76	0.73	0.69	0.74	0.88	0.74	**0.64**	0.37
OM-R	**0.69**	0.39	**0.69**	0.67	0.79	0.74	**0.72**	0.77	**0.95**	**0.78**	**0.64**	0.49
OM-E	**0.69**	0.36	0.65	0.65	**0.81**	0.72	0.67	0.81	**0.95**	0.77	0.56	**0.50**

TABLE 3: Extended Evaluation

values. Figure 3 shows the obtained results. As s increases, i.e., $s \geq 0.001$, the performance of the method starts to decrease. While achieving best results using a small s is good news (since a small s yields better computational and memory complexity), we are not able to clearly pinpoint the cause. We verified these observations across tasks other than TOEFL and MEN and we found a similar pattern. Based on our experiments, we propose index vectors of *only* one positive integer value, particularly when embeddings are induced from huge corpora and they are weighted prior to similarity measurement (i.e., the most likely scenario in which this method is used).

For practical reasons, this is, indeed, good news. Since index vectors that contain only one non-zero element (which have the fixed value of +1) are sufficient, and because in most applications dimensionality won't exceed a couple of thousands, index vectors can be represented and reside in memory using only one byte of data (for storing the dimension that carries the non-zero value). In addition, the computational complexity of OM does not depend on the model's dimensionality ;[4] it is a function of s (besides the size of corpus). Consequently, OM shows not only a better task-based performance but also a much lower computational and space complexity. Note furthermore that OM offers a solution that avoids float arithmetic operations during the construction of models.

4.2 Extending Relatedness Evaluations

To provide a better picture of OM's task-based performances, we extend our evaluations to relatedness tests other than MEN. This includes `WS353` (Finkelstein *et al.*, 2001), `WS-SIM` and `WS-REL` by (Agirre *et al.*, 2009), the classic tests of `MC30` (Miller & Charles, 1991) and `RG65` (Rubenstein & Goodenough, 1965), the Stanford Rare Word dataset (`RW`) (Luong *et al.*, 2013), `MTurk287` by (Radinsky *et al.*, 2011), `MTurk771` (Halawi *et al.*, 2012), the more recent `SimLex999` (Hill *et al.*, 2015), and `YP130` verb relatedness (Yang & Powers, 2006). (Faruqui & Dyer, 2014) provides tools for easy evaluation of these data-sets (as well as a number of baselines). In all the above-mentioned tests, the figure of merit (i.e., Spearman's correlation ρ) is computed similarly to the MEN test. To provide a baseline, we took the off-the-shelf Word2Vec embeddings provided by (Baroni *et al.*, 2014). These embeddings are learnt from the set of ukWaC, WaCkypedia (Baroni *et al.*, 2009), and the British National Corpus (BNC) corpus. After thorough parameter tuning, (Baroni *et al.*, 2014) report that these vectors showed the best performance in their evaluations (denoted by W2C-B). We perform our experiments by applying OM to the same corpora, using the same context window size 5+5, in order to obtain embeddings for the same set of 300,000 words provided by (Baroni *et al.*, 2014). For OM, we use index vectors with one non-zero element and set the dimension to 777 (denoted by OM-O). Table 3 summarizes the obtained results.

4. In contrast to neural embedding techniques.

4.3 Boosting Performances

Is it possible to improve the performance reported in § 4.2 ? The answer is certainly yes, through several means. Evidently, like many embedding techniques, OM can be adapted to each task, e.g., by choosing an appropriate context window size, and in general, by making linguistically more informed decisions. The performance for the `YP130` verb relatedness test is for instance boosted by filtering context elements through dependency parses (we achieved $\rho = 0.71$). In addition, as shown in § 4.1, increasing the dimensionality of models boosts performance.

Apart from these, OM allows for heterogeneous context types : For example, in addition to context words, one may also use word character-grams to capture the morphosyntactic structure of context words (we observed slight improvement across tasks), or even go further and use bags-of-visuals (i.e. to say, there is no limitation on defining and using various types of contexts at once). In comparison, this is not that straightforward with neural embedding techniques : often the objective function (which defines the underlying optimization task) must be modified and adapted for the new context types. For instance, we might improve `WS-REL` performance by incorporating information about word co-occurrences in documents : We assigned each document d_i in our input corpora to an index vector $\vec{r_i^d}$. In addition to index vectors of context words, we added $\vec{r_i^d}$ to target vectors of words that co-occurred in the same document d_i. Subsequently, `WS-REL` performance increased from 0.64 (using only context words) to 0.69.

Applying guidelines provided by (Levy *et al.*, 2015) is another avenue for improving the results. We witnessed that using different weighting methods (including scaled and shifted PPMI) and correlation measures for similarity assessments for different tasks can also enhance performances. For example, using the set of OM-O vectors (Table 3) and scaled PPMI weighting and different similarity measures (noted in parentheses), the performance for `YP130` increases to 0.54 (r), `MC30` to 0.84 (r^2), `MEN` to 0.75 (r), `RG65` to 0.79 (γ), and `SimLex999` to 0.41 (r). Nevertheless, we leave the systematic evaluation of these methods of performance boosting for future.

Besides the classic methods suggested above, two interesting methodologies have emerged recently : *retrofitting*, i.e., refining word embeddings using lexical relations available in lexical resources (Faruqui *et al.*, 2015) ; and, *meta-embedding*, i.e., ensembling several sets of embeddings (Yin & Schütze, 2016). OM can easily be adapted for retrofitting and meta-embedding. For instance, we implement a notion of retrofitting by simply adding the index vector of a word to the vectors of all other words that co-occur with it within a lexical entry (e.g. checking for co-occurrence across all WordNet synsets). We update OM-O vectors using this mechanism and with the lexical resources used in experiments by (Faruqui *et al.*, 2015), i.e., WordNet (Miller, 1995), FrameNet (Baker *et al.*, 1998), and PPDB (Ganitkevitch *et al.*, 2013). The result is a considerable improvement in performances (see OM-R in Table 3). We also observe a noticeable boost in results by adapting a simplified meta-embedding approach (see OM-E in Table 3). Using the same set of index vectors, we build several models using context windows of different sizes (namely, 2+2 to 10+10). Given a word in our vocabulary, its meta-embedding is obtained by adding vectors that represent this word in each model.

5 Conclusion

We proposed an embedding method based on positive-only random projections. Each dimension of our model represent a randomly chosen bag-of-words (or any other types of context). Compared to count and neural-predictive techniques, our method provides a simpler mechanism for building and updating distributional models. Thanks to its underlying randomized projection, our method is fast and has low computational complexity. Particularly, by allowing weighting at the reduced dimensionality, our method achieves a performance comparable to many recent embedding techniques. However, in contrast to these approaches, our method is more flexible, it can accommodate several types of contexts at once, it can be updated at any time, and it does not require time-consuming training and parameter tuning.

Acknowledgments

This work was supported by the CRC 991 "The Structure of Representations in Language, Cognition, and Science" funded by the German Research Foundation (DFG). We would like to thank the anonymous reviewers for their helpful comments.

Références

AGIRRE E., ALFONSECA E., HALL K., KRAVALOVA J., PAŞCA M. & SOROA A. (2009). A study on similarity and relatedness using distributional and wordnet-based approaches. In *NAACL '09*, p. 19–27, Stroudsburg, PA, USA : ACL.

BAKER C. F., FILLMORE C. J. & LOWE J. B. (1998). The berkeley framenet project. In *ACL '98 - Volume 1*, p. 86–90, Stroudsburg, PA, USA : ACL.

BARONI M., BERNARDINI S., FERRARESI A. & ZANCHETTA E. (2009). The wacky wide web : a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, **43**(3), 209–226.

BARONI M., DINU G. & KRUSZEWSKI G. (2014). Don't count, predict ! In *ACL*, p. 238–247, Baltimore, Maryland.

BOUMA G. (2009). Normalized (pointwise) mutual information in collocation extraction. In *Proceedings of the Biennial GSCL Conference*.

BRUNI E., TRAN N. K. & BARONI M. (2014). Multimodal distributional semantics. *J. Artif. Int. Res.*, **49**(1), 1–47.

BULLINARIA J. A. & LEVY J. P. (2012). Extracting semantic representations from word co-occurrence statistics : stop-lists, stemming, and svd. *Behavior Research Methods*, **44**(3), 890–907.

CHEN P. Y. & POPOVICH P. M. (2002). *Correlation : Parametric and Nonparametric Measures*. Quantitative Applications in the Social Sciences. Sage Publications.

CHURCH K. W. & HANKS P. (1990). Word association norms, mutual information, and lexicography. *Comput. Linguist.*, **16**(1), 22–29.

DASGUPTA S. & GUPTA A. (2003). An elementary proof of a theorem of johnson and lindenstrauss. *Random Struct. Algorithms*, **22**(1), 60–65.

EVERT S. (2004). *The statistics of word cooccurrences : word pairs and collocations*. PhD thesis, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart.

FARUQUI M., DODGE J., JAUHAR S. K., DYER C., HOVY E. H. & SMITH N. A. (2015). Retrofitting word vectors to semantic lexicons. In *NAACL-HLT*, p. 1606–1615.

FARUQUI M. & DYER C. (2014). Community evaluation and exchange of word vectors at wordvectors.org. In *ACL : System Demonstrations*, Baltimore, USA : ACL.

FINKELSTEIN L., EVGENLY G., YOSSI M., EHUD R., ZACH S., GADI W. & EYTAN R. (2001). Placing search in context : the concept revisited. In *WWW*.

FYSHE A., WEHBE L., TALUKDAR P. P., MURPHY B. & MITCHELL T. M. (2015). A compositional and interpretable semantic space. In *NAACL*.

GANITKEVITCH J., VAN DURME B. & CALLISON-BURCH C. (2013). PPDB : The paraphrase database. In *NAACL-HLT*, p. 758–764, Atlanta, Georgia : ACL.

GOODMAN L. A. & KRUSKAL W. H. (1954). Measures of association for cross classifications. *Journal of the American Statistical Association*, **49**(268), 732–764.

HALAWI G., DROR G., GABRILOVICH E. & KOREN Y. (2012). Large-scale learning of word relatedness with constraints. In *KDD '12*, p. 1406–1414, New York, NY, USA : ACM.

HILL F., REICHART R. & KORHONEN A. (2015). SimLex-999 : Evaluating semantic models with genuine similarity estimation. *Comput. Linguist.*, **41**(4), 665–695.

KANERVA P., KRISTOFERSON J. & HOLST A. (2000). Random indexing of text samples for latent semantic analysis. In *CogSci*, p. 103–6 : Erlbaum.

LANDAUER T. & DUMAIS S. (1997). A Solution to Plato's Problem : The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge. *Psychological Review*, **104**, 411–240.

LAPESA G. & EVERT S. (2014). A large scale evaluation of distributional semantic models : Parameters, interactions and model selection. *TACL*, **2**, 531–545.

LEVY O. & GOLDBERG Y. (2014). Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems 27*, p. 2177–2185.

LEVY O., GOLDBERG Y. & DAGAN I. (2015). Improving distributional similarity with lessons learned from word embeddings. *TACL*, **3**, 211–225.

LI Y., XU L., TIAN F., JIANG L., ZHONG X. & CHEN E. (2015). Word embedding revisited : A new representation learning and explicit matrix factorization perspective. In *IJCAI*, p. 3650–3656.

LUONG T., SOCHER R. & MANNING C. D. (2013). Better word representations with recursive neural networks for morphology. In *CoNLL*, p. 104–113 : ACL.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013a). Efficient estimation of word representations in vector space. *CoRR*, **abs/1301.3781**.

MIKOLOV T., SUTSKEVER I., CHEN K., CORRADO G. S. & DEAN J. (2013b). Distributed representations of words and phrases and their compositionality. In *NIPS*, p. 3111–3119. Curran Associates, Inc.

MILLER G. A. (1995). Wordnet : A lexical database for english. *Commun. ACM*, **38**(11), 39–41.

MILLER G. A. & CHARLES W. G. (1991). Contextual correlates of semantic similarity. *Language and Cognitive Processes*, **6**(1), 1–28.

PENNINGTON J., SOCHER R. & MANNING C. (2014). Glove : Global vectors for word representation. In *EMNLP*, p. 1532–1543, Doha, Qatar : ACL.

RADINSKY K., AGICHTEIN E., GABRILOVICH E. & MARKOVITCH S. (2011). A word at a time : Computing word relatedness using temporal semantic analysis. In *WWW '11*, p. 337–346, New York, NY, USA : ACM.

RUBENSTEIN H. & GOODENOUGH J. B. (1965). Contextual correlates of synonymy. *Commun. ACM*, **8**(10), 627–633.

SCHNABEL T., LABUTOV I., MIMNO D. & JOACHIMS T. (2015). Evaluation methods for unsupervised word embeddings. In *EMNLP*, p. 298–307, Lisbon, Portugal : ACL.

STEIN B. (2007). Principles of hash-based text retrieval. In *SIGIR '07*, p. 527–534, New York, NY, USA : ACM.

TSVETKOV Y., FARUQUI M., LING W., LAMPLE G. & DYER C. (2015). Evaluation of word vector representations by subspace alignment. In *EMNLP*, p. 2049–2054, Lisbon, Portugal : ACL.

TURNEY P. D. (2001). Mining the web for synonyms : PMI-IR versus LSA on TOEFL. In *EMCL '01*, p. 491–502, London, UK, UK : Springer-Verlag.

TURNEY P. D. & PANTEL P. (2010). From frequency to meaning : Vector space models of semantics. *J. Artif. Int. Res.*, **37**(1), 141–188.

YANG D. & POWERS D. M. W. (2006). Verb similarity on the taxonomy of wordnet. In *The 3rd International WordNet Conference (GWC-06)*, Jeju Island, Korea.

YIN W. & SCHÜTZE H. (2016). Learning word meta-embeddings. In *ACL*.