# Probabilistic Word Embeddings in Neural IR: A Promising Model That Does Not Work as Expected (For Now)

Alberto Purpura
purpuraa@dei.unipd.it
University of Padua
Padua, Italy

Marco Maggipinto
marco.maggipinto@dei.unipd.it
University of Padua
Padua, Italy

Gianmaria Silvello
silvello@dei.unipd.it
University of Padua
Padua, Italy

Gian Antonio Susto
sustogia@dei.unipd.it
University of Padua
Padua, Italy

## ABSTRACT

In this paper, we discuss how a promising word vector representation based on Probabilistic Word Embeddings (PWE) can be applied to Neural Information Retrieval (NeuIR). We illustrate PWE pros for text retrieval, and identify the core issues which prevent a full exploitation of their potential. In particular, we focus on the application of elliptical probabilistic embeddings, a type of PWE, to a NeuIR system (i.e., MatchPyramid). The main contributions of this paper are: (i) an analysis of the pros and cons of PWE in NeuIR; (ii) an in-depth comparison of PWE against pre-trained Word2Vec, FastText and WordNet word embeddings; (iii) an extension of the MatchPyramid model to take advantage of broader word relations information from WordNet; (iv) a topic-level evaluation of the MatchPyramid ranking models employing the considered word embeddings. Finally, we discuss some lessons learned and outline some open research problems to employ PWE in NeuIR systems more effectively.

## CCS CONCEPTS

• **Mathematics of computing → Probabilistic representations**; • **Information systems → Document representation**; • **Computing methodologies → Natural language processing**.

## KEYWORDS

probabilistic word embedding; neural information retrieval; natural language processing

**ACM Reference Format:**
Alberto Purpura, Marco Maggipinto, Gianmaria Silvello, and Gian Antonio Susto. 2019. Probabilistic Word Embeddings in Neural IR: A Promising Model That Does Not Work as Expected (For Now). In *The 2019 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '19), October 2–5, 2019, Santa Clara, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3341981.3344217

This work was partially supported by the CDC-STARS project funded by the University of Padua. The work was also partially funded by the EXAMODE (contract n. 825292) part of the H2020-ICT-2018-2 call of the European Commission.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*ICTIR '19, October 2–5, 2019, Santa Clara, CA, USA*
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6881-0/19/10…$15.00
https://doi.org/10.1145/3341981.3344217

## 1 INTRODUCTION

Neural Information Retrieval (NeuIR) models have gained increasing attention in the past few years thanks to their ability to automatically extract relevance patterns from raw text [16, 19]. These methods, differently from traditional lexical approaches for IR, that rely mainly on exact matching signals between query and document terms, can retrieve relevant documents which do not contain any query term. The development of Word Embeddings (WE) played a crucial role in this methodological advancement. Indeed, WE have been widely used in NeuIR [16] for the generation of matching signals, which can be further interpreted by a neural model to return query-document relevance scores.

Much work has been done in the past few years in the IR community on the study and development of new neural architectures to interpret these semantic matching signals [16], but less attention has been devoted to the representation of words being used. Several approaches rely on WE trained with Word2Vec [15] or GloVe [22], or learn their own model-specific word representations like Neural Vector Space Model (NVSM) [26] and Deep Structured Semantic Model (DSSM) [10]. Some other works considered the possibility of learning WE specific for relevance matching in IR [33, 34].

In this paper, we analyze the application of Elliptical Probabilistic Embeddings [17], a form of WE that we believe can have a sizeable impact on NeuIR approaches. These embeddings are a recent development of Probabilistic Word Embeddings (PWE) [28] that model complex word patterns in text by introducing "uncertainty" in the embeddings. Their aim is to generate semantically richer matching signals to be used for similarity matching. Indeed, point embeddings, such as those obtained with Word2Vec, can be regarded as a particular (and degenerate) case of PWE where "the uncertainty is infinitely concentrated on a single point" [17]. Probability similarity measures based on PWE are broader and provide an opportunity for additional flexibility in the definition of similarity between words. The main advantage of PWE is the ability to encode more complex word relations such as hypernymy [17]. This ability however, comes at the cost of a complex training process given that the number of parameters of each word embedding increases exponentially. In this work, we describe and evaluate an effective training procedure for the selected type of PWE, and we adapt an existing NeuIR architecture – i.e. MatchPyramid [19] – to use this new representation of words. In order to better evaluate the advantage of encoding broader relations between terms in WE, we also compare the trained PWE to a similar type of word vectors, WordNet Embeddings (WNE) [25]. This representation is not obtained from raw text data like Word2Vec embeddings or PWE, but from a graph reproducing the relations between a selection of words such as WordNet. These relations are therefore encoded in the

WE, obtaining a set of vectors with similar properties to PWE. These embeddings, however, do not perform well in the retrieval task since the set of words and relations which can be encoded is limited by the size of the ontology adopted – this leads to a marked Out-Of-Vocabulary (OOV) problem. To address this issue, we propose an extension of the MatchPyramid architecture that combines WNE matching signals with those obtained from character *n*-grams embeddings pre-trained with FastText [6]. The overall purpose of this work is to understand the potential contribution of complex and flexible WE in Information Retrieval (IR). Hence, we consider MatchPyramid trained on traditional Word2Vec embeddings as our baseline and we compare our extended architectures against it. We show that PWE are promising for NeuIR and can lead to overall improvements in text retrieval, even though core limitations need to be overcome to be competitive with state-of-the-art approaches. By digging into PWE drawbacks, we analyze where they fail to provide the improvements we expect, explain why this happens and propose research directions to address the identified issues. The rest of the paper is organized as follows. In Section 2, we present some background information on WE with a focus NeuIR; in Sections 3 and 4, we introduce PWE and WNE; in Section 5 we present our experimental setup; in Section 6, a comparison between PWE, FastText Embeddings (FTE) and WNE, and their evaluation on the retrieval task; in Section 7 we draw our conclusions and discuss future work.

## 2 BACKGROUND

Neural models have been employed in IR to automatically detect regularities in matching patterns and exploit them for document retrieval. These models usually rely on WE that can be pre-trained or learned from scratch to enable *best* matching between query and document terms and consequently improve retrieval [16].

***Point word embeddings.*** Word2Vec [15] or GloVe [22] WE have been widely used in NeuIR systems because they incorporate a mixture of topical and typical notions of relatedness between terms, leveraging on their co-occurrences in large collections of documents. FastText [6] is an extension of Word2Vec to compute word representations based on character n-grams. Using subword-level information is particularly interesting to build vectors for unknown words. This is done by summing the character *n*-grams in the unknown term. For instance, the tri-grams in the word "apple" are "app", "ppl", and "ple" (ignoring the starting and ending of boundaries of words). Hence, the word embedding for "apple" will be the sum of these *n*-gram vectors. However, since these models are trained to encode only co-occurrence relations, they have some drawbacks when used in NeuIR. In co-occurrence based models, word embeddings of terms which are often used in the same contexts have a high similarity score despite having different meanings (e.g. the terms "dangerous" and "safe") [33] and this could lead to misleading matching signals for NeuIR models. BERT [3] or XLNET [32] are novel approaches to train WE which achieve state-of-the-art performance on lots of Natural Language Processing (NLP) tasks. However, they are extremely complex to train and, despite some differences in the architecture of the model and the training process, obtain word representations which have a similar structure to the previous models.

***WordNet embeddings.*** WordNet [4] [1] is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. WordNet has been widely used in IR, but was never competitive with state-of-the-art approaches [1, 5, 18, 27]. The main issues that need to be faced when WordNet is employed by an IR system – e.g., for query or document expansion [12, 14, 29] – are the lack of collection-specific terms in the database, the need to design a customized similarity measure among terms [31] or the difficulty of extracting the correct word sense from the ontology. With the diffusion of word embeddings, new approaches were also developed to employ this representation also for lexical resources such as WordNet, Wiktionary [2] or Wikidata [3] [24, 25]. A vector representation of words has the advantage of making the computation of the similarity between terms much faster than when using a graph-based distance metric such as Wu and Palmer similarity [31]. In this work, we employ WordNet to create word representations which do not present the same drawbacks in NeuIR of co-occurrence based models such as Word2Vec, and encode richer relations between words such as hypernymy.

***Probabilistic word embeddings.*** Typically, WE are obtained by embedding each word into a point/vector in the Euclidean space $\mathbb{R}^d$, where a distance-based similarity measure (e.g., cosine similarity) reflects the semantic similarity [15]. While this approach has been proven effective in many NLP and IR applications, recent trends started to generalize point embeddings using probability measures spaces. In this case, each word is represented by a mean vector and covariance matrix pair introducing uncertainty in the representations. The main advantage of such approach is the ability to encode the different senses of a word or complex word relations such as hypernymy through a covariance matrix [17, 28]. In the literature, various methods to train probabilistic embeddings have been proposed, starting from [28] which pioneered the method introducing Gaussian Embeddings where each word is represented by a Multivariate Gaussian distribution. The similarity metrics employed are the Probability Product Kernel or Kullback-Leibler Divergence (KLD). Nevertheless, these metrics have some drawbacks since they diverge for degenerate covariance matrices and assume high values when the variance is very small. These problems can be settled by working on the space of elliptical distributions (i.e., probability measures with elliptical level sets) endowed with the Wasserstein-Bures product [17]. Indeed, the Wasserstein-Bures product handles degenerate cases and has better numerical stability than the metrics employed in [28]. Moreover, elliptical embeddings are flexible since they can be transformed into Gaussian Embeddings or point embeddings by assuming the probability measures to be a Gaussian or a Dirac delta, respectively. Finally, elliptical embeddings are stable and provide interesting numerical properties that guarantee smooth training and easy employment with neural architectures; for these reasons, we prefer them to other recent developments in probabilistic embeddings, such as box embeddings [11].

[1] https://wordnet.princeton.edu/.
[2] https://www.wiktionary.org/.
[3] https://www.wikidata.org/.

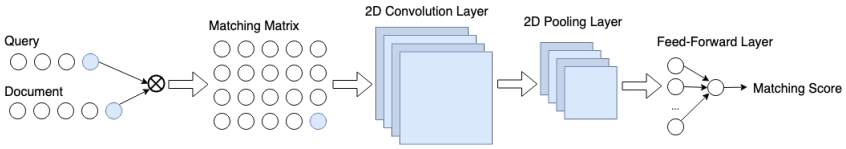



**Figure 1: Architecture of MatchPyramid.**

***NeuIR Models.*** With the exception of [33, 34], only a few other approaches have been studying new WE specifically tailored for IR. A meaningful example is described in [33], where Zamani and Croft leverage on millions of user queries from AOL logs [21] to train a set of word embeddings specific to relevance matching. However, the majority of neural architectures, either rely on pre-trained Word2Vec or GloVe embeddings [7, 19], or consider word representations as something to be learned by the NeuIR model [26], without any association to their semantic meaning.

In this work, we evaluate different WE – with a focus on PWE – and study how they impact the performance of a NeuIR system. The NeuIR model we employ is based on the MatchPyramid architecture [19]. We selected this architecture because, despite being a supervised deep neural model, it does not require lots of training data [19] and can be applied to relatively small IR collections such as TREC Robust04 [30]. Moreover, it relies solely on the interactions between query and document terms, and processes these raw matching signals directly in its neural model without performing any prior operation on them. This enables a more accurate evaluation of the impact of the new WE, than other NeuIR models – possibly also better performing, such as DRMM [7] – which alter the matching signals before processing them. Finally, MatchPyramid is a better performing approach than similar deep neural architectures for text matching such as DeepMatch [13] or other approaches based on Convolutional Neural Network (CNN) such as ARC-I/ARC-II [9]. The architecture of MatchPyramid is depicted in Figure 1. First, the model receives as input two embedded sequences of words, a query and a document. Second, it computes a similarity matrix between each pair of terms from the query and the document. In our experiments, we used cosine similarity which, despite not being the best performing one for the model [19], is the one used in Word2Vec and FastText for the training of the embeddings. Cosine similarity is the best metric to evaluate the WE characteristics since the relative positions of the embeddings in the space have been optimized during training according to this distance. Third, a 2D-CNN is applied to the matching matrix to extract local matching signals, followed by a dynamic pooling layer [20]. Finally, a shallow Feed-Forward (FF) neural network with two layers is used to compute the matching score between the query and the document in input. The model was trained to minimize the loss function: $L(q, d^+, d^-; \theta) = max(0, 1 - S(q, d^+) + S(q, d^-))$, where $\theta$ indicates the model parameters, $q$ represents a user query and $d^+, d^-$ are respectively a relevant and a non-relevant document for $q$. To decrease retrieval time, we filtered the documents to be ranked with MatchPyramid, computing the matching score for each query only for the top 2K documents retrieved with the QLM model [23] in Terrier v.4.1. [4]

## 3 ELLIPTICAL PROBABILISTIC EMBEDDINGS

PWE generalize classical word embeddings considering each word representation as a probability measure. In the case of elliptical distributions, a measure $\mu_{\boldsymbol{m},\boldsymbol{M}}$ is determined by its mean ($\boldsymbol{m} \in \mathbb{R}^d$) and covariance matrix ($\boldsymbol{M} \in \mathbb{R}^{d \times d}$). Since the covariance matrix is symmetric, the number of parameters required for each embedding can be computed as $d + \frac{d(d+1)}{2}$, where $d$ is the size of the vector $\boldsymbol{m}$. Let $\mu_{\boldsymbol{a},\boldsymbol{A}}$ and $\mu_{\boldsymbol{b},\boldsymbol{B}}$ be two elliptical measures, we define the Wasserstein-Bures pseudo-dot-product [17] as:

$$[\mu_{\boldsymbol{a},\boldsymbol{A}} : \mu_{\boldsymbol{b},\boldsymbol{B}}] = \langle \boldsymbol{a}, \boldsymbol{b} \rangle + Tr\left(\boldsymbol{A}^{\frac{1}{2}}\boldsymbol{B}\boldsymbol{A}^{\frac{1}{2}}\right)^{\frac{1}{2}} \tag{1}$$

Here $\langle \cdot, \cdot \rangle$ is the scalar product and $Tr(\cdot)$ is the trace operation. Such pseudo-dot-product can be seen as an extension of the standard dot product to the space of elliptical measures and defines a similarity measure between them. This allows us to train the embeddings using i.e. the Continuous Bag of Words (CBOW) [15] paradigm, maximizing the similarity between the target word and the words in its context window while minimizing it for words outside the context. For a word $w$, its context is defined as the $N_w^+$ words located before and after each occurrence of $w$ in the corpus; the rationale behind this training strategy is that words surrounded by similar contexts should be semantically correlated. Formally, this translates into minimizing the following hinge loss (Eq. 2) on the training set elements:

$$\sum_{w \in \mathcal{W}} \left[ \mathcal{M} - \frac{1}{N_w^+} \sum_{c^+ \in C_w^+} [\mu^{c^+} : \mu^w] + \frac{1}{N_w^-} \sum_{c^- \in C_w^-} [\mu^{c^-} : \mu^w] \right]. \tag{2}$$

Here, for each occurrence of the word $w \in \mathcal{W}$ in the corpus, we indicate with $C_w^+$ the sets of terms of size $N_w^+$ surrounding it (positive context), and with $C_w^-$, the negative samples (i.e. randomly sampled words not belonging to the current context of $w$) of cardinality $N_w^-$. Moreover, we indicate with $\mu^w, \mu^{c^+}, \mu^{c^-}$ respectively the embeddings of $w, c^+$ and $c^-$ parametrized by their mean and covariance matrices – we do not indicate them here in order to simplify the notation – and $\mathcal{M}$ is a slack variable. An important aspect of the training procedure is that the covariance matrices are required to be symmetric positive definite. In order to satisfy this requirement, we factorize them as $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^T$, learning $\boldsymbol{L}$ during the optimization. For more details on optimization and differentiation of the loss function we invite the readers who are interested to refer to [17]. The dot product defined in Eq. 1 is affected by the words frequency in the training corpus; hence, it is necessary to define its normalized counterpart to be used during the word

[4] The modified version of Terrier we used for our experiments is available at the url: https://github.com/gridofpoints.

similarity evaluation phase. Given two words $w_a$ and $w_b$ and their elliptical measures $\mu_{\boldsymbol{a},\boldsymbol{A}}$ and $\mu_{\boldsymbol{b},\boldsymbol{B}}$, we define the normalized dot product as:

$$\mathscr{C}[\mu_{\boldsymbol{a},\boldsymbol{A}} : \mu_{\boldsymbol{b},\boldsymbol{B}}] = \frac{\langle \boldsymbol{a}, \boldsymbol{b} \rangle}{||\boldsymbol{a}|| \cdot ||\boldsymbol{b}||} + \frac{Tr\left(\boldsymbol{A}^{\frac{1}{2}}\boldsymbol{B}\boldsymbol{A}^{\frac{1}{2}}\right)^{\frac{1}{2}}}{\sqrt{Tr\boldsymbol{A} \cdot Tr\boldsymbol{B}}} \tag{3}$$

$\mathscr{C}[\mu_{\boldsymbol{a},\boldsymbol{A}} : \mu_{\boldsymbol{b},\boldsymbol{B}}]$ has its maximum value 2 when $\mu_{\boldsymbol{a},\boldsymbol{A}} = \mu_{\boldsymbol{b},\boldsymbol{B}}$ and defines the similarity between $w_a$ and $w_b$ in an elliptical space.

We were able to employ PWE in the NeuIR MatchPyramid model, by changing the way the matching matrix between query and document terms is created. In this case, instead of cosine similarity, we employed the Wasserstein-Bures normalized dot product as a similarity measure between two terms.

## 4 WORDNET EMBEDDINGS

Classical word embedding models such as Word2Vec, GloVe or FastText, employ large collections of documents to train word representations. These approaches leverage on co-occurrence relations between words in order to encode semantic information in their vector representation. On the other hand, the goal of WordNet embeddings is to employ semantic networks (i.e. WordNet) to infer word representations. In this case, WE encode the same information of the semantic graph used to create them, but in a more compact way. This information is then used in neural systems with more efficiency and little modifications to existing architectures. A semantic network is a graph where each node represents a term and each edge indicates a relation between two terms. The strength of the semantic affinity of two words is proportional to the number of paths connecting two nodes – the higher the number of paths, the stronger the affinity – and to their length – the shorter the paths, the stronger the affinity.

WordNet Embeddings (WNE) are computed as follows. First, the semantic network $G$, is represented with an adjacency matrix $M$ where $M_{ij} = 1$ if words $w_i, w_j$ are connected by an edge in $G$, and 0 otherwise. Second, in order to also represent weaker relations between terms, the following operation is applied: $M_G^{(n)} = I + \alpha M + \alpha^2 M^2 + ... + \alpha^n M^n$, where $I$ is the identity matrix, $M^n$ is the matrix where $M_{ij}$ counts the number of paths of length $n$ between nodes $i, j$ in $G$, and $\alpha < 1$ is a decay factor determining how longer paths are dominated by shorter ones. This iterative procedure is repeated until $M_G^{(n)}$ converges into matrix $M_G$, which is analytically obtained by an inverse matrix operation given by $M_G = \sum_{e=0}^{\infty}(\alpha M)^e = (I - \alpha M)^{-1}$ [25]. Third, each row of $M_G$ is normalized using L2-norm and Positive Point-wise Mutual Information transformation is applied to reduce the eventual bias introduced by the conversion towards words with more senses. Finally, Principal Component Analysis (PCA) is used to reduce the size of the matrix and achieve WE of size 850. The obtained WNE are evaluated under the task of determining the semantic similarity between pairs of lexical units and obtained results around 15% superior to the ones of Word2Vec with the same evaluation data set SimLex-999 [8]. The main drawback of WNE is that the number of terms in the embeddings model is limited by the size of the semantic graph used to create them. In fact, the model we considered for our retrieval experiments contains only 60K terms: this represents a well-known limit for a NeuIR system. Hence, we developed an extension of the MatchPyramid architecture, which integrates the missing words with FTE. We call this model MP FTE WNE and its architecture is reported in Figure 2. We employ WNE in this model to generate one of the cosine similarity-based matching matrices in Figure 2. In particular, we compute the cosine similarity between each pair of terms in a query and a document, according to this WE model. If one of the terms is missing in the model, we consider the similarity of the pair as 0. Then, we sum this matrix as shown in Figure 2, to the one obtained analogously with pre-trained FTE, parametrizing the sum with a scalar coefficient learned by the model.

## 5 EXPERIMENTAL SETUP

Our experiments are based on the TREC Robust04 collection [30]. The Robust04 corpus is composed of 528,155 documents (news) from TIPSTER Disk 4&5 minus the CR, 249 topics and graded relevance judgments. We perform the retrieval experiments on this collection to reproduce the results obtained with the original MatchPyramid model [19], and also because it is a widely used collection to evaluate NeuIR systems for *ad-hoc* retrieval, i.e. DRMM [7] and NVSM [26]. Furthermore, since many NeuIR models do not scale well either during training or retrieval, it is therefore necessary to evaluate them on small/medium collections. We are aware that, being our evaluation based only on one experimental collection, all our results might not apply to any *ad-hoc* retrieval collection. However, also we believe that, with our in-depth embedding-based, statistical and topic-by-topic analyses, we are able to overcome the limitations of our experimental setup, and make more general claims. As in the original MatchPyramid paper, we consider only the topic title and the first 500 tokens for each document. We also stemmed all of the terms in the collection using Krovetz Stemmer [5] and removed stopwords using the INQUERY stoplist [2]. The MatchPyramid model used has a total of 4 layers: a convolutional and a pooling layer, followed by a two layer feed-forward neural network. The convolutional kernel has a size of $3 \times 3$, while the pooling size is $1 \times 10$. The MatchPyramid architecture we adopt to evaluate PWE (MP PWE) and FTE (MP FTE) is shown in Figure 1. In MP PWE, we compute the matching matrices using the Wasserstein-Bures dot product described in Section 3. The architecture we use for the evaluation of WNE (MP FTE WNE), is depicted in Figure 2. Here, as in MP FTE, we employ the standard cosine similarity to compute the matching matrices and keep the same configuration described above for the convolutional, pooling and feed forward layers.

To evaluate retrieval, we consider the P@5, nDCG@5 and MAP measures. This is done to ensure a comprehensive overview of the systems performance, especially in the top part of the rankings which is the most important for the end-user. In particular, P@5 gives us a set-based binary measure of the number of relevant documents in the first positions of our runs; nDCG@5 provides a refinement of this measure weighing the order of the documents and graded relevance. The MAP yields an overall evaluation of the rankings, that also takes into account the recall of the systems. We also perform a qualitative evaluation of different WE models: PWE,

[5] In order to use pre-trained word embeddings models with stemming, we stemmed all of the terms in the term dictionary of the model and if one stemmed term could be assigned to more than one embedding we computed its embedding as their average.

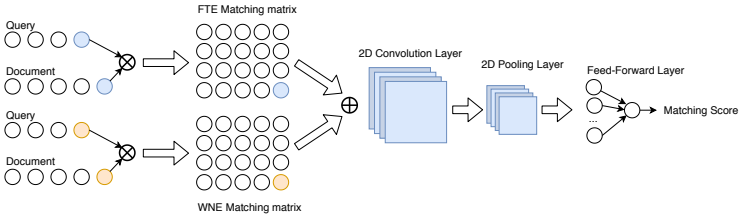



**Figure 2: Architecture of MatchPyramid with WNE.**

| Model | Training corpus | Number of elems. | Emb. size |
|---|---|---|---|
| W2V | Wikipedia | 190K | 50 |
| FTE | Wikipedia | 2.5M | 300 |
| WNE | WordNet v.3 [4] | 60K | 850 |
| PWE | Robust04 | 76K | 1325 |

**Table 1: Characteristics of the word embeddings models we employed for our experiments.**

FTE [6], WNE [7], and Word2Vec Embeddings [8]. The characteristics of these models are reported in Table 1. For the PWE training, we consider context windows of size 5 – i.e., we consider the two terms before and after each occurrence of a word as its context. In [17], the PWE were trained on ukWaC, with 2 billion words, and WaCkypedia_EN which is a dump of Wikipedia 2009 [9]. However, we could not train PWE on this large corpora due to the limited computational resources available. Indeed, as we described in Section 3, each elliptical probabilistic embedding has 1325 trainable parameters if we consider an embedding space with $d = 50$ dimensions. For this reason, we decided to use the Robust04 collection for the training of the PWE. Despite the limited training data, we were still able to achieve better retrieval results with PWE than with the pre-trained WE on Wikipedia with Word2Vec used in [19]. For reproducibility purposes, we share the code and the WE to perform all our experiments in a public repository. [10]

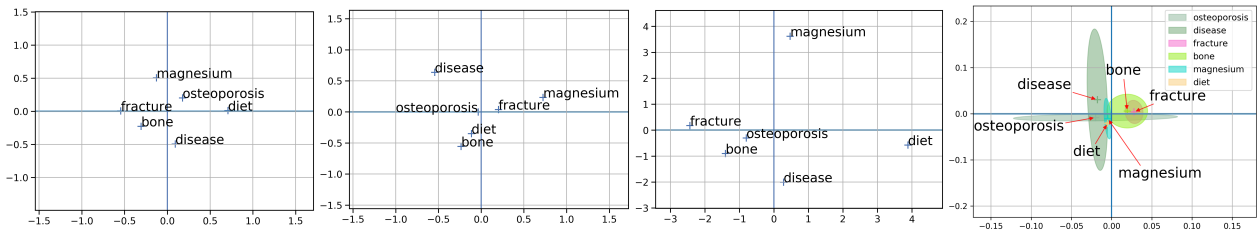
## 6 EVALUATION

***Word Embeddings.*** We conduct a qualitative evaluation of WE. To this end, we select a set of terms related to topic 403: "Osteoporosis", showing their 2D representation according to different WE models. [11] To visualize the relations between elliptical distributions, we adopt the following strategy. We begin performing PCA on the set of mean vectors $\mathbf{m_i}$ associated to the words $w_i$ to visualize. The resulting two-dimensional vectors are the centers of the ellipses to be displayed. Then, we consider the precision matrix $\mathbf{P_i}$ [12] associated to each $w_i$ and project its two main eigenvectors to the 2D space identified with PCA. The norms of these projections, multiplied by the normalized corresponding eigenvalues are the sizes of the two semi-axes of the ellipses. Finally, we compute the angle of the ellipses considering the projection of the main eigenvector of each $\mathbf{P_i}$ and the first principal component of PCA. This visualization process preserves – within the limits of the dimensionality reduction operation – the measure of term similarity adopted for the training of the PWE.

The plots in Figure 3, show how PWE correctly represents the affinity between the terms selected from the topic description, e.g. between the words "bone", "disease" and "osteoporosis" – where the ellipses intersections represent the concept that osteoporosis is a bone disease – or "osteoporosis" and "diet". This however, does not hold for all other models if we consider the cosine similarity between terms – which is proportional to the angle between the words in the scatter plots and is the measure used to compute term similarity for retrieval. For instance, in Figure 3a, we see a correlation in the Word2Vec model between the terms "osteoporosis" and "magnesium" (cosine similarity 0.11) – but not with "disease" (cosine similarity $-0.05$). Conversely, in the WNE model, the terms "osteoporosis" and "disease" are related (cosine similarity of 0.28) because of their relation in WordNet. However, the WNE model also comes with some drawbacks, since it does not capture the relation between "osteoporosis" and "fracture", most likely because of the longer path that connects the terms in WordNet semantic graph. Finally, the representations of FTE and PWE, are very close in terms of similarity scores between the term "osteoporosis" and the other words considered in the comparison – both models recognize a positive correlation of all the considered terms with "osteoporosis" – but the relative angles with this word differ greatly. In fact, if we only considered the mean vectors of the PWE – represented as the centers of the ellipses – we should have had a completely different set of similarity scores. The final similarity score of PWE is, in fact, obtained thanks to the contribution of the covariance matrices – associated to the probability measure of each term – in the Wasserstein-Bures product. Herein, this similarity component is represented by the overlap between the ellipses. Indeed, the covariance matrix associated to each PWE, allows to represent the correlation between terms in a selective way, along certain directions – associated to their different meanings or usage contexts – without penalizing their representation in the rest of the space, e.g. in the direction of the word senses/contexts which are used less frequently in the training samples.

***NeuIR Systems.*** First, we reproduce the original results obtained on the TREC Robust04 collection by the MatchPyramid model using a pre-trained Word2Vec model [13] (i.e. MP W2V) [19]. We obtain a P@20 of 0.275, very close to the value of 0.272 reported in the MatchPyramid original paper; the MAP and nDCG scores we obtain present an absolute difference with the original ones of less

[6] https://fasttext.cc/docs/en/crawl-vectors.html.
[7] https://github.com/nlx-group/WordNetEmbeddings.
[8] https://github.com/pl8787/MatchPyramid-TensorFlow.
[9] Available at: http://wacky.sslmit.unibo.it/doku.php?id=corpora; the pre-trained PWE are not publicly available.
[10] https://github.com/albpurpura/PE4IR.
[11] We use Principal Component Analysis (PCA) as a dimensionality reduction technique to display W2V, WNE, and FTE.
[12] The precision matrix is by definition the inverse of the covariance matrix.
[13] https://github.com/pl8787/MatchPyramid-TensorFlow.




**(a) W2V** **(b) WNE** **(c) FTE** **(d) PWE**

**Figure 3: Visualization of a set of terms related to the Robust04 topic 403, according to different WE models.**

| Model | P@5 | nDCG@5 | MAP |
|---|---|---|---|
| MP W2V | 0.3711 | 0.3617 | 0.1823 |
| MP PWE | 0.4130$^{\dagger}$ | 0.4060$^{\dagger}$ | 0.1840 |
| MP FTE | 0.4145$^{\dagger\dagger}$ | 0.3992$^{\dagger\dagger}$ | 0.1952 |
| MP FTE WNE | **0.4161**$^{\dagger\dagger}$ | **0.4099**$^{\dagger\dagger}$ | **0.1978**$^{\dagger\dagger}$ |

**Table 2: Retrieval performance of MatchPyramid with different word embedding models. †† and † indicate a significant difference (paired t-test) from MP W2V with $\alpha = 0.01$ and $\alpha = 0.05$, respectively.**

than 0.01. This is likely due to small differences in the document parsing and tokenization processes of the collection. Second, we test MatchPyramid employing PWE (MP PWE). The results of our retrieval experiments are reported in Table 2. In this case, we observe a 0.04 increase in P@5 and nDCG@5 compared to MP W2V. This improvement is obtained despite the great difference in the number of terms available in Word2Vec and PWE. In fact, there are more than 190K terms in MP W2V, while MP PWE has only 60K terms. Third, we test MatchPyramid employing FTE (MP FTE). In this case there are no OOV terms. We observe a small P@5 increase compared to MP PWE, and a slightly lower nDCG@5 value. Nevertheless, the most relevant difference with MP PWE is in terms of MAP which shows that the words that are missing in PWE have an effect on the number of relevant documents retrieved by the system. This however does not significantly affect the top ranked documents. Finally, we test the extended MatchPyramid architecture (MP FTE WNE) employing two different WE: FTE and WNE. In this case, we get the best results overall. We run a paired t-test between MP W2V (our baseline) and the other described models as reported in Table 2. We see that MP FTE WNE is the only system with significant improvements over the baseline for all the considered evaluation measures. MP PWE and MP FTE present significant improvements when we consider only the top ranked documents with P@5 and nDCG@5. Nevertheless, the improvements over the baseline are not as noticeable as we expected. PWE has an impact on NeuIR architectures, but from a quantitative analysis of the performances, it is not evident where and why PWE provides a gain or a loss.

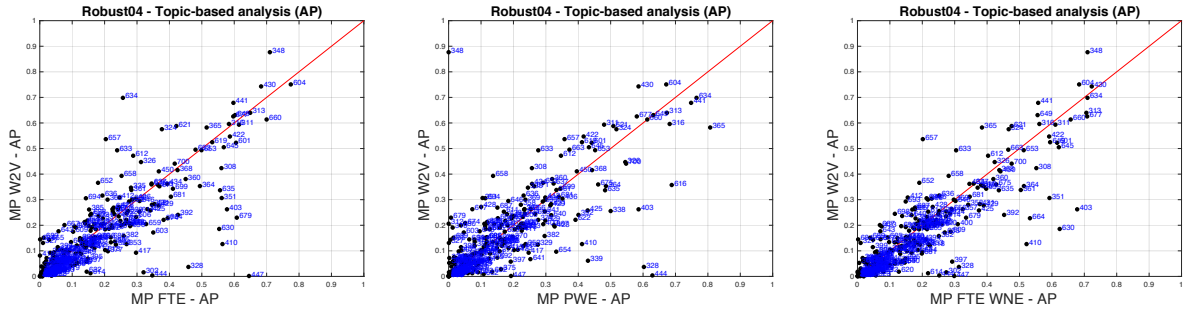
To further investigate this issue, we perform a topic-by-topic evaluation of the considered systems. In Figure 4, we compare, for each topic, the AP of MP W2V against MP PWE and MP FTE WNE. In the figure, if a point lies below the red bisector, it means that the system associated to the x-axis has a higher AP than the one on the y-axis (and viceversa). Whereas, the nDCG@5 performance difference between MP W2V and MP PWE or MP FTE WNE is reported in Figures 5b and 5a, respectively. The stem plots indicate the nDCG@5 score difference between the systems appearing on

| Topic ID | Title | Topic ID | Title |
|---|---|---|---|
| 312 | Hydroponics | 447 | Stirling engine |
| 328 | Pope beatifications | 630 | Gulf war syndrome |
| 348 | Agoraphobia | 647 | Windmill electricity |
| 353 | Antartica exploration | 657 | School prayer banned |
| 403 | Osteoporosis | 659 | Cruise health safety |
| 444 | Supercritical fluids | 697 | Air traffic controller |

**Table 3: Topics considered for the topic-level qualitative evaluation.**

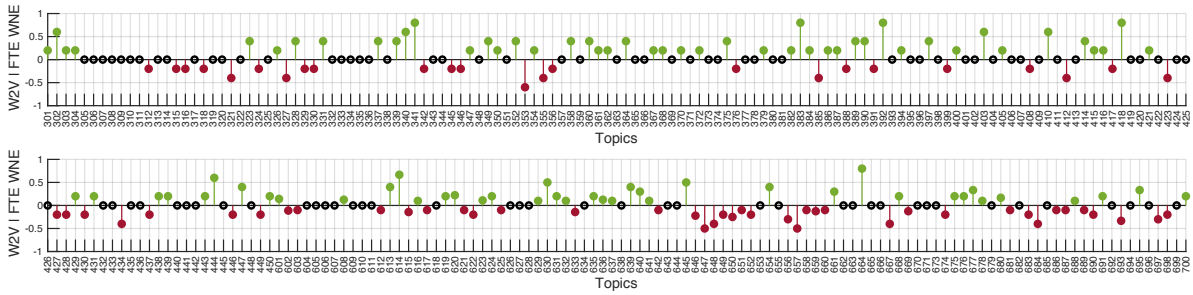
the y-axis. If the value is positive (green), the system associated to the positive side of the axis performs better than the one associated to the negative one with reference to the topics indicated on the x-axis (and vice versa for red points). Relying on the data visualized in Figures 4 and 5, we selected twelve topics – reported in Table 3 – where MP W2V and MP PWE show a sizeable performance difference both in terms of AP and nDCG@5.

First of all, in Figure 4 we notice that using different WE affects the performance on the selected topics. For instance, we observe a considerable performance difference in Figure 4a on topic 447 where MP W2V has an AP of 0.0, while MP FTE has an AP close to 0.65. A similar situation is true for topics 328 and 444 in Figure 4b, where the performance of MP PWE is over 0.60 and the AP of MP W2V is below 0.05. If we consider topics 403 and 630 in Figures 4a and 4c, we also notice the improvement obtained by combining FTE and WNE compared to relying only on FTE. In fact, the conjunct use of FTE and WNE improves the AP of about 0.10 with respect to employing FTE alone. Conversely, in other topics such as 447 or 444 (see Figures 4a and 4c) we observe a performance deterioration because MP FTE WNE cannot rely on WordNet for additional information since the terms "supercritical" and "stirling" are not contained in WNE. If we examine the stem plots in Figure 5a and consider the nDCG@5 difference for each topic of MP W2V and MP FTE WNE, we notice that there are only three topics where the nDCG@5 of MP W2V is evidently higher ($\geq 0.5$) than the nDCG@5 of MP FTE WNE. These topics are 353, 647, and 657. In these cases, the lower performance of MP FTE WNE is likely due to the lack of the query terms "antartica", "windmill" and "banned" in WNE. When we compare MP W2V with MP PWE in Figure 5b, the topics where the former system's nDCG@5 is over 0.5 higher than the one of the latter are: 312, 348, 659, and 697. The lower performance of MP PWE associated to topics 312 and 348, is due to the fact that all of the query terms are missing from the WE model. On the other hand, the performance difference on the topics 659 and 697, is likely due to the small training set of PWE. In fact, the key terms "cruise" and "controller" of topics 659 and 697 only appear respectively 3K and 1.5K times in the

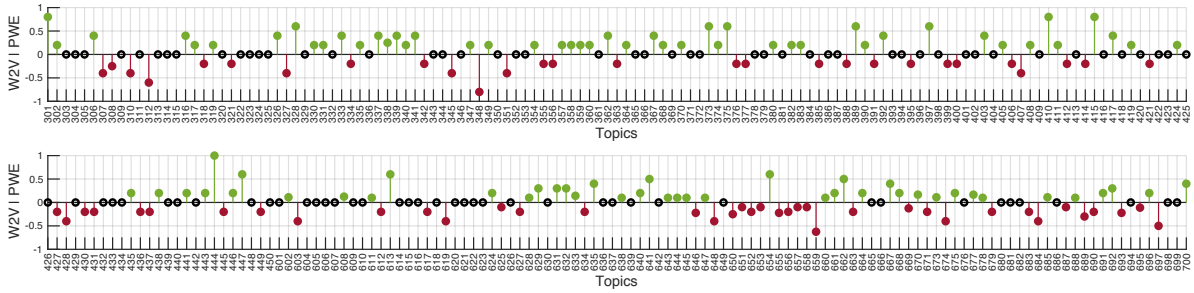



(a) (b) (c)

**Figure 4: Comparison of the AP of different MatchPyramid models employing FastText Embeddings (MP FTE), Probabilistic Word Embeddings (MP PWE) and FastText + WordNet Embeddings (MP FTE+WNE), against MatchPyramid with Word2Vec embeddings (MP W2V).**




**(a) MP W2V and MP FTE WNE nDCG@5 difference by topic.**




**(b) MP PWE and MP W2V nDCG@5 difference by topic.**

**Figure 5: The stem plots indicate the nDCG@5 score difference between the systems indicated on the y-axis. If the value is positive (green), the system associated to the positive side of the axis (MP FTE WNE in 5a and MP PWE in 5b) performs better than the one associated to the negative one (and vice versa, red points) on the topic indicated on the x-axis.**

training corpus – while the average frequency of the other terms in these topics is 53K. Hence, the lack of training contexts for these terms, which are the most important in both of the topics, led to a poor training of the corresponding WE and consequently to a low retrieval performance. The experiments performed in this section, highlight the potential of PWE in NeuIR systems. In fact, the qualitative evaluation of PWE in Figure 3d, underscores the role of the covariance matrix in the WE model. This component allows to balance the contribution of the mean vector in the similarity function we employed – the Wasserstein-Bures normalized dot product – adjusting it for the different contexts where a word is used in the training corpus. The potential of PWE emerged also when we employ this WE model for retrieval, where we obtained a statistically significant improvement in the top part of the rankings, compared to Word2Vec embeddings. Moreover, through the topic level analysis we performed, we could analyze and motivate why PWE led or did not lead to a performance improvement w.r.t. the baseline system employing Word2Vec embeddings. The potential of this innovative WE is also confirmed by the positive results obtained combining FTE with WNE which have similar properties to PWE. Indeed, the addition of WNE leads to the best performances overall, which is only constrained by the number of terms in the ontology used to train them.

## 7 CONCLUSIONS

In this work, we analyzed – for the first time to our knowledge – the performance in *ad-hoc* retrieval, of elliptical probabilistic word embeddings (PWE). This type of WE is able to encode richer relations among terms [17], hence to improve retrieval performance by generating meaningful similarity matching signals between query and document terms. In order to evaluate the generalization power of PWE, we first performed a qualitative evaluation of the WE, visualizing the elliptical probability distributions. Then, we employed a NeuIR architecture, MatchPyramid [19], to evaluate the quality of the matching signals, generated with different WE models, for the retrieval task. We selected this architecture for its capacity to compute the relevance score for a query-document pair, considering only the similarity scores between each term from the two texts, without any alteration. In this way, we eliminated any potential bias against different WE models with distinct characteristics, e.g. size, sparsity, or structure. We also extended the MatchPyramid model (MP FTE WNE) to employ two types of WE with different characteristics at the same time: FastText [6] and WordNet embeddings [25]. The former embedding model is based on character n-grams embeddings, which allowed us to obtain a semantically meaningful representation of all the terms in our experimental collection, eliminating the OOV problem. The latter, allowed us to enrich the matching signals with broader conceptual-semantic and lexical relations form WordNet. MP FTE WNE achieved, whenever the query terms were present in the WordNet embeddings models, a better performance than Word2Vec or FTE. To conclude, although our study does not yet lead to a sizeable improvement over traditional Word2Vec embeddings, PWE can nonetheless be considered a promising word vector representation of terms for NeuIR. In fact, analyzing the similarity matrices associated to these embeddings and the document rankings topic-by-topic, we observed that PWE can be a competitive word representation for retrieval, outperforming in many cases a pre-trained Word2Vec model. On the other hand, the large number of parameters of PWE, makes them difficult to use and train, especially on small collections. In fact, we encountered numerous engineering challenges to optimize retrieval and training time for this type of WE, and we could not train or use them on a larger collection. In the future, we plan to continue our optimization of the training and retrieval processes using these embeddings. We also plan to conduct a further study on the characteristics of each query to automatically identify and predict the cases where PWE will lead to a better performance over traditional WE. This would allow us to use PWE only when they are likely to assure significantly better performance compared to traditional WE.

## REFERENCES


[1] C. Basu, L. Dietz, and C. Fellbaum. 2018. WordNetContext: Information Retrieval-friendly Access to WordNet Senses. In *ProfS/KG4IR/Data:Search@SIGIR (CEUR Workshop Proceedings)*, Vol. 2127. CEUR-WS.org, 63–64.

[2] J. P. Callan, W. B. Croft, and J. Broglio. 1995. TREC and TIPSTER experiments with INQUERY. *Information Processing & Management* 31, 3 (1995), 327–343.

[3] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

[4] C. Fellbaum. 1999. WordNet: An Electronic Lexical Database. *Computational Linguistics* 25, 2 (June 1999), 292–296.

[5] J. B. Gao, B. W. Zhang, and X. H. Chen. 2015. A WordNet-based semantic similarity measurement combining edge-counting and information content theory. *Engineering Applications of Artificial Intelligence* 39 (2015), 80–88.

[6] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov. 2018. Learning Word Vectors for 157 Languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

[7] J. Guo, Y. Fan, Q. Ai, and W. B. Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *CIKM 2016*. 55–64.

[8] F. Hill, R. Reichart, and A. Korhonen. 2015. SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation. *Computational Linguistics* 41, 4 (2015), 665–695.

[9] B. Hu, Z. Lu, H. Li, and Q. Chen. 2014. Convolutional Neural Network Architectures for Matching Natural Language Sentences. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014*. 2042–2050.

[10] P. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. P. Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *CIKM 2013*. 2333–2338.

[11] X. Li, L. Vilnis, D. Zhang, M. Boratko, and A. McCallum. 2019. Smoothing the Geometry of Probabilistic Box Embeddings. In *ICLR*.

[12] S. Liu, F. Liu, C. T. Yu, and W. Meng. 2004. An effective approach to document retrieval via utilizing WordNet and recognizing phrases. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 266–272.

[13] X. Lu and H. Li. 2013. A Deep Architecture for Matching Short Texts. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013*. 1367–1375.

[14] R. Mihalcea and D. Moldovan. 2000. Semantic Indexing Using WordNet Senses. In *Proceedings of the ACL-2000 Workshop on Recent Advances in Natural Language Processing and Information Retrieval (RANLPIR '00)*, Vol. 11. Association for Computational Linguistics, Stroudsburg, PA, USA, 35–45.

[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 3111–3119.

[16] B. Mitra and N. Craswell. 2018. An Introduction to Neural Information Retrieval. *Foundations and Trends in Information Retrieval* 13, 1 (2018), 1–126.

[17] B. Muzellec and M. Cuturi. 2018. Generalizing Point Embeddings using the Wasserstein Space of Elliptical Distributions. In *NeurIPS 2018*. 10258–10269.

[18] V. M. Ngo, T. H. Cao, and T. M. V. Le. 2018. WordNet-Based Information Retrieval Using Common Hypernyms and Combined Features. *CoRR* abs/1807.05574 (2018).

[19] L. Pang, Y. Lan, J. Guo, J. Xu, and X. Cheng. 2016. A Study of MatchPyramid Models on Ad-hoc Retrieval. *CoRR* abs/1606.04648 (2016).

[20] L. Pang, Y. Lan, J. Guo, J. Xu, S. Wan, and X. Cheng. 2016. Text Matching as Image Recognition. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. 2793–2799.

[21] G. Pass, A. Chowdhury, and C. Torgeson. 2006. A picture of search. In *Proceedings of the 1st International Conference on Scalable Information Systems*. 1.

[22] J. Pennington, R. Socher, and C. D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*. 1532–1543.

[23] J. M. Ponte and W. B. Croft. 2017. A Language Modeling Approach to Information Retrieval. *SIGIR Forum* 51, 2 (2017), 202–208.

[24] Rothe S. and H. Schütze. 2015. AutoExtend: Extending Word Embeddings to Embeddings for Synsets and Lexemes. In *ACL 2015*, Vol. 1. 1793–1803.

[25] C. Saedi, A. Branco, J. A. Rodrigues, and J. Silva. 2018. WordNet Embeddings. In *Proceedings of The Third Workshop on Representation Learning for NLP: Rep4NLP@ACL*. 122–131.

[26] C. Van Gysel, M. De Rijke, and E. Kanoulas. 2018. Neural vector spaces for unsupervised information retrieval. *TOIS* 36, 4 (2018), 38.

[27] G. Varelas, E. Voutsakis, P. Raftopoulou, E. G. M. Petrakis, and E. E. Milios. 2005. Semantic Similarity Methods in wordNet and Their Application to Information Retrieval on the Web. In *Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management (WIDM '05)*. ACM, New York, NY, USA, 10–16.

[28] L. Vilnis and A. McCallum. 2015. Word Representations via Gaussian Embedding. In *ICLR*.

[29] E. M. Voorhees. 1994. Query Expansion Using Lexical-Semantic Relations. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*. 61–69.

[30] E. M. Voorhees. 2004. Overview of the TREC 2004 Robust Track. In *TREC 2004*.

[31] Z. Wu and M. Palmer. 1994. Verbs Semantics and Lexical Selection. In *Proceedings of the 32nd Annual Meeting of ACL (ACL '94)*. ACL, Stroudsburg, PA, USA, 133–138.

[32] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding.

[33] H. Zamani and W. B. Croft. 2017. Relevance-based Word Embedding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 505–514.

[34] H. Zamani, M. Dehghani, W. B. Croft, E. G. Learned-Miller, and J. Kamps. 2018. From Neural Re-Ranking to Neural Ranking: Learning a Sparse Representation for Inverted Indexing. In *CIKM 2018*. 497–506.