Integrating and Evaluating Neural Word Embeddings in Information Retrieval

Guido Zuccon[1] Bevan Koopman[2,1] Peter Bruza[1] Leif Azzopardi[3]

[1]Queensland University of Technology, Brisbane, Australia
[2]Australian e-Health Research Centre, CSIRO, Brisbane, Australia
[3]University of Glasgow, Glasgow, United Kingdom

g.zuccon@qut.edu.au, bevan.koopman@csiro.au, p.bruza@qut.edu.au, leif.azzopardi@glasgow.ac.uk

ABSTRACT

Recent advances in neural language models have contributed new methods for learning distributed vector representations of words (also called word embeddings). Two such methods are the continuous bag-of-words model and the skipgram model. These methods have been shown to produce embeddings that capture higher order relationships between words that are highly effective in natural language processing tasks involving the use of word similarity and word analogy. Despite these promising results, there has been little analysis of the use of these word embeddings for retrieval.

Motivated by these observations, in this paper, we set out to determine how these word embeddings can be used within a retrieval model and what the benefit might be. To this aim, we use neural word embeddings within the well known translation language model for information retrieval. This language model captures implicit semantic relations between the words in queries and those in relevant documents, thus producing more accurate estimations of document relevance.

The word embeddings used to estimate neural language models produce translations that differ from previous translation language model approaches; differences that deliver improvements in retrieval effectiveness. The models are robust to choices made in building word embeddings and, even more so, our results show that embeddings do not even need to be produced from the same corpus being used for retrieval.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval

1. INTRODUCTION

Statistical language models have proven effective in information retrieval, but are still primarily based on the exact matching of query terms. Since queries are generally terse and relevant documents may use a different vocabulary, these language models can only get us so far. Ideally the identification of relevant documents should be based on *semantic* matching rather than exact keyword matching. One way in which researchers have approached tackling this problem is with the use of translation language models, where semantic matching is modelled as a translation between terms in the query and those in relevant documents [5, 16]. A core component of such models is the estimation of the translation probability between terms.

At the same time, advances in neural language modelling have produced novel distributed representations of words that may allow for effective estimations of translation probabilities between terms. Specifically, two such models, the continuous bag-of-words model and the skipgram model [21], produce vector representations of words (also called word embeddings) that have proven effective on a number of linguistic tasks (including word similarity and word analogy [21]).

The hypothesis of this paper is that these word embeddings can be exploited in information retrieval. Specifically, we first show how word embeddings can be incorporated into a retrieval model by levaraging the translation language model framework. Second, we empirically determine the benefit of such an approach, including contributing anunderstanding of the impact that choices made when building word embeddings have on retrieval effectiveness.

The empirical evaluation shows that neural translation language models, which introduce a novel word meaning representation, provide superior retrieval effectiveness than previous translation language models when evaluated on a number of TREC test collections. Neural translation language models are robust to decisions on how the word embeddings are constructed. Even more so, our results show that the word embeddings do not even need to be produced from the same corpus being used for retrieval; thus the word embeddings, could, in fact, be derived from a general purpose collection. With the introduction of word representations based on neural language models, this research opens a number of avenues for exploiting implicit semantic relationships like operations on vectors of word embeddings [23] for improved retrieval effectiveness.

2. TRANSLATION LANGUAGE MODELS FOR INFORMATION RETRIEVAL

2.1 Statistical Translation Language Models

The use of language modelling for information retrieval is an attractive approach as it directly models how language

ADCS, December 08 - 09, 2015, Parramatta, NSW, Australia

is used to express meaning; in addition, it has proven an effective method for document retrieval [30]. In the language modelling framework, documents are ranked according to the log-likelihood

$$logp(q|d) = \sum_{i:c(q_i;d)>0} \log \frac{p_s(q_i|d)}{\alpha_d p(q_i|C)} + nlog\alpha_d + \sum_{i=1}^{n} logp(q_i|C) \quad (1)$$

where the last component of the equation can be ignored for ranking purposes as it is document independent.

Different smoothing strategies to language models instantiate the likelihood of seen terms $p_s(w|d)$ (and consequently α_d) in different ways. Dirichlet smoothing (used in this paper) estimates $p_s(w|d)$ according to the following equation:

$$p_\mu(w|d) = \frac{c(w,d) + \mu p(w|C)}{|d| + \mu} = \frac{p_{ml}(w|d) * |d| + \mu p(w|c)}{|d| + \mu} \quad (2)$$

In Dirichlet smoothing language models, $p_s(w|d)$ is estimated by mixing the maximum likelihood estimation, $p_{ml}(w|d)$, with the collection background probability, $p(w|C)$. Berger and Lafferty have proposed an alternative estimation of $p_s(w|d)$ inspired by models in statistical machine translation [5]. In that work, they modelled retrieval as a machine translation process and estimated the query likelihood by means of a translation model that computes the likelihood that the query has been produced by a translation of the document:

$$p_t(w|d) = \sum_{u \in d} p_t(w|u)p(u|d) \quad (3)$$

where $p_t(w|u)$ represents the probability of translating term u into w. As Karimzadehgan and Zhai have noted [16], the translation probability $p_t(w|u)$ allows for the incorporation of semantic relations between terms with non-zero probabilities: this provides a sort of "semantic smoothing" for $p(w|d)$. The new estimation $p_t(w|d)$ provided by translation language models can be injected into the Dirichlet smoothed language models by substituting $p_{ml}(w|d)$ with $p_t(w|d)$ [16].

The key challenge in translation language models then becomes how to estimate $p_t(w|u)$, i.e., the probability of translation of u into the (query) term w. Berger and Lafferty have proposed estimating the translation probabilities for each document by synthesising a query for which the document would be relevant [5]. This approach requires the availability of labelled training data (relevance assessments), is inefficient and does not guarantee translation probabilities are available for all query terms [16].

2.2 Estimation of Translation Probability based on Mutual Information

As alternative to the synthetic queries process, Karimzadehgan and Zhai have proposed a family of approaches to estimate $p_t(w|u)$ based on mutual information [16, 17]. In statistics, mutual information measures the mutual dependence between two random variables by determining how similar the joint distribution $p(X,Y)$ is to the products of the marginals, $p(X)p(Y)$. When applied to distributions of terms in documents, mutual information provides a measure of the strength of relation between two terms.

In mutual information based translation language models, for each term in the collection, scores are computed for words with high mutual information and further normalised [16]. The mutual information between terms w and u is computed as (refer to [16] for details):

$$I(w,u) = \sum_{X_w=0,1} \sum_{X_u=0,1} p(X_w, X_u) log \frac{p(X_w, X_u)}{p(X_w)p(X_u)} \quad (4)$$

where X_u and X_w are binary variables indicating the presence or absence of u and w, respectively. Mutual information values are then normalised to obtain the translation probability $p_{mi}(w|u)$ estimated based on mutual information:

$$p_{mi}(w|u) = \frac{I(w,u)}{\sum_{w'} I(w',u)} \quad (5)$$

We refer to the use of $p_{mi}(w|u)$ to estimate the translation probability $p_t(w|u)$ in Equation 3 as the translation language model based on mutual information (TLM-MI).

Table 1 provides sample translations obtained with TLM-MI (left), derived from TREC 1 query 55, "insider trading". The translation terms provided by TLM-MI were related to the query terms. Indeed, the method did unveil terms that were related to cases of insider trading reported in the collection (e.g., "drexel", "burnham", "lambert" refer to the Wall Street investment banking firm Drexel Burnham that was forced into bankruptcy due to its involvement in illegal activities[1]). However, there were also terms that, although related, may intuitively harm retrieval (e.g., more general terms such as "wall", "index" and "prices").

The estimation of the translation probability described in Equation 5 forms the basis of variations of translation language models based on mutual information. Mutual information does not guarantee that the self-translation probability $p(w|w)$ (i.e., the probability of translating a word w to itself) is higher than any other translation probability [16] (it is possible that $p(w|u) > p(w|w)$). An axiomatic analysis of translation language models has shown that this is not a desired situation because documents that match a query word q_i exactly may receive a lower score than documents that match translations of q_i that have a translation probability higher than the self-translation probability [17]. To overcome this issue, a heuristic has been proposed to control the effect of self-translation via a parameter α:

$$p_{mi-\alpha} = \begin{cases} \alpha + (1-\alpha)p_{mi}(u|u) & \text{if } w = u \\ (1-\alpha)p_{mi}(w|u) & \text{if } w \neq u \end{cases} \quad (6)$$

We refer to the translation language model that uses this estimation as TLM-MI-α.

Similarly, an alternative heuristic is to impose constant self-translation probabilities for all words in the vocabulary [17], i.e., setting $p(u|u)$ to a constant value s for every u. This produces another variant of TLM-MI, which we refer to as TLM-MI-s, where $p_t(w|u)$ is estimated according to:

$$p_{mi-s} = \begin{cases} s & \text{if } w = u \\ (1-s)\frac{p_{mi}(w|u)}{\sum_{v \neq u} p_{mi}(v|u)} & \text{if } w \neq u \end{cases} \quad (7)$$

3. A NEURAL TRANSLATION LANGUAGE MODEL

A variety of neural network-based language models[2] have emerged as effective approaches for generating representations of words [4, 26, 22].

A fundamental characteristic of neural language models is that in such architectures, words are mapped to vectors in a high dimensional, real valued space (forming a word

[1] http://en.wikipedia.org/wiki/Drexel_Burnham_Lambert

[2] For brevity we refer to these as neural language models.

TLM-MI				NTLM - cbow				NTLM - skipgram			
w = insider		w = trading		w = insider		w = trading		w = insider		w = trading	
u	$p(w\|u)$	u	$p(w\|u)$	u	$p(w\|u)$	u	$p(w\|u)$	u	$p(w\|u)$	u	$p(w\|u)$
insider	0.094	trading	0.050	insider	0.285	trading	0.216	insider	0.169	trading	0.164
trading	0.023	exchange	0.016	fraud	0.104	traders	0.103	fraud	0.102	traders	0.103
securities	0.023	stock	0.014	drexel	0.095	market	0.094	drexel	0.099	futures	0.099
fraud	0.015	market	0.013	criminal	0.084	stock	0.090	securities	0.096	stock	0.097
drexel	0.013	prices	0.012	securities	0.084	markets	0.085	racketeering	0.093	exchange	0.094
burnham	0.013	traders	0.009	racketeering	0.084	futures	0.084	bribery	0.091	market	0.093

Table 1: Example word translations, along with translation probabilities, using TLM-MI (left), NTLM with cbow embeddings (centre) and with skipgram embeddings (right). The terms for which we are seeking translations are those of query 55, "insider trading", from TREC 1.

embedding); the mappings are learnt through the optimisation of an objective function. Along with the probabilistic model learnt by training neural network language models, a distributed word representation (or word embedding) is also learnt and can be further exploited.

An objective function that is often used for training word embeddings is to learn a vector for a target word which predicts the vectors for words occurring near to it; this is the intuition behind the continuous skipgram model.

3.1 Continuous bag-of-word model

The continuous bag-of-words model (cbow) constructs term representations by optimising the ability of context words to predict the representations of the current target word (i.e., predict a word given its context), and is based on a standard feed-forward neural language model without the intermediate projection layer [21].

Given a target word w_t and a sequence of training words $\mathcal{W} = \{w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}\}$ (where w_{t-k} precedes and w_{t+k} follows w_t by k positions), the objective of the cbow model is to maximise the likelihood of correctly predicting the target word w_t, where weights for different positions in the sequence are shared. Because all words share the projection layer in the neural network, all words are projected into the same position and thus their vectors are averaged. Formally, the cbow model generates a vector v_t, corresponding to the target word w_t, as the average of the vectors of words within the training sequence $\mathcal{W}$, i.e., $v_t = \frac{1}{|\mathcal{W}|}\sum_{i=1}^{|\mathcal{W}|} v_i$. Averaging vector representations of input words implies that the order of the words in the context does not matter.

3.2 Skipgram model

The skipgram model constructs term representations by optimising their ability to predict the representations of surrounding terms.

At initialisation, the vector representations of the words are assigned random values; these vector representations are then optimised using gradient descent with decaying learning rate by iterating through sentences observed in the training corpus. Specifically, given a sequence of training words $\mathcal{W} = \{w_1, \ldots, w_t, \ldots, w_n\}$, the objective of the skipgram model is to maximise the following average log probability

$$\frac{1}{|\mathcal{W}|}\sum_{i=1}^{|\mathcal{W}|}\sum_{-|\mathcal{W}|\leq j\leq|\mathcal{W}|, j\neq 0} \log p(w_{t+j}|w_t) \tag{8}$$

The context window size, $|\mathcal{W}|$ determines which words surrounding the target word w_t are considered for the computation of the log probability (where the window is centred around the target word). However, note that in the skipgram model $|\mathcal{W}|$ is in effect the maximal window radius: in practical implementations[3], for each word in the corpus, a window size $|\mathcal{W}|' \leq |\mathcal{W}|$ is sampled uniformly from $[1, |\mathcal{W}|]$ [15].

The probability of an output word is computed according to the softmax function:

$$p(w_O|w_I) = \frac{\exp(v_{w_O}^\top v_{w_I})}{\sum_{w=1}^{|V|}\exp(v_w{}^\top, v_{w_I})} \tag{9}$$

where the v_{w_I} and v_{w_O} are the vector representations of the input and output vectors, respectively, and $\sum_{w=1}^{|V|}\exp(v_w^\top, v_{w_I})$ is the normalisation factor, whose role is to normalise the inner product results across all vocabulary words ($|V|$ is the vocabulary size). Negative sampling is used to reduce computational complexity, where the objective function is modified to force the model to distinguish the target word w_O from one that is drawn from a noise distribution [23].

3.3 Estimating Translation Probabilities with Neural Language Models

The use of neural language models based on continuous bag-of-words or skipgram gives rise to two different word embeddings. Word embeddings can be used to estimate translation probabilities in translation language models; specifically, cosine similarity can be used as a proxy for $p(u|w)$:

$$p_{cos}(u|w) = \frac{cos(u,w)}{\sum_{u'\in V} cos(u',w)} \tag{10}$$

where $cos(u,w)$ is the cosine between the vector representation of word u and w and this is normalised to obtain a probability distribution over all possible translations. These estimations are then plugged in Equation 3 to derive the neural translation language models (NTLM) based on cbow (NTLM-cbow) and skipgram (NTLM-skipgram).

Table 1 reports example translations for query 55 in TREC 1, "insider trading", obtained using the cbow (centre) and skipgram models (right). As for mutual information (see Section 2.2), the methods based on word embeddings suggest translations that appeared highly related to the topic expressed by the query. However, there were a number of key differences between the neural language models and the model based on mutual information. First, syntactic variations of the query term that refer to the same meaning (e.g., u="traders" for w ="trading") were often assigned a higher translation probability (and ranked higher in general among the possible translations). (Similar findings are observed across many other queries.) Second, with neural translation language models, there was generally less difference in magnitude when comparing the self translation probability $P(w|w)$ with the next highest translation probability $P(w|u)$. For example, with cbow and skipgram, the

[3] e.g., in the widely adopted *word2vec* implementation of skipgram models, which is used in this paper.

self translation probability $p(\text{trading}|\text{trading})$ was between 1.6 and 2 times larger than the next translation probability, while with TLM-MI the same self translation probability was more than 3 times larger than that of the next translation. The effect that these differences between mutual information and neural language models have on retrieval effectiveness is what we investigate in our empirical evaluation of Section 4.

4. EMPIRICAL EVALUATION

Our evaluation was conducted to answer the following research questions: **RQ1**: Do neural language models provide translation probability estimates between words that lead to improvement in retrieval effectiveness when compared to state-of-the-art translation language models? **RQ2**: How sensitive are neural translation language models to the different ways word embeddings can be constructed: latent space dimensions, window size and type of word embedding (cbow or skipgram)? **RQ3**: Does the choice of corpus used to induce word embeddings influence retrieval effectiveness, and specifically, what is the effect of embeddings constructed on a different corpus to that used in retrieval?

4.1 Experiment Settings

4.1.1 Data Sets

Evaluation was done using four standard TREC collections for ad-hoc retrieval: 1) news articles from AP88-89 (TREC disk 1 and 2) with topics from TREC 1, 2 and 3 ad-hoc (topics 51-200); 2) news articles from WSJ87-92 (TREC disk 1) with topics from TREC 1, 2 and 3 ad-hoc (topics 51-200); 3) webpages from the crawl of the .gov domain from DOTGOV with topics from TREC 2002 (topics 551-600); and 4) TREC Medical Records Track (MedTrack) collection (2011 and 2012). The use of AP and WSJ is in line with previous work on translation language models [16, 17], although we consider more documents (AP88-89) and more topics (TREC 1, 2 and 3). The motivation for the use of AP88-89 and WSJ87-92 was that manual runs were included to form the pools of documents for assessment. Thus, the judged documents did not necessarily contain the query keywords. Therefore, these collections would be more suited for the evaluation of translation methods that exploit more semantic relationships. The MedTrack collection was chosen as medical search is a particular domain known to suffer from issues of vocabulary mismatch [14, 18], which may be alleviated by the translation language models. Translation language models have never been evaluated before on the DOTGOV collection (and in general on collections larger than few hundred thousands documents). Deriving translation probabilities on large collections poses computational challenges if particular attention is not paid to optimising the operations involved in calculating these estimations[4], although this is generally not a problem for NTLM. For example, the estimation of probabilities based on mutual information requires computing co-occurrence statistics for every term in the vocabulary with respect to every other term – and then to normalise these statistics over all the vocabulary to form the probability distribution $p_{mi}(w|u)$.

Statistics of collections are reported in Table 2.

[4]Our experiments were performed on the full DOTGOV.

Desc.	# Docs	Topics	Avg. query size	Vocab. size
AP88-89	164,597	51-200 AdHoc	5.2	247,350
WSJ87-92	173,252	51-200 AdHoc	5.2	216,539
DOTGOV	1,247,442	551-600	3.3	3,051,601
MedTrack	100,866	101–185	8.9	55,065

Table 2: Statistics for the TREC collections used.

4.1.2 Software Implementations

Both indexing and retrieval was implemented using the *Terrier* IR toolkit [24]. Both documents and queries were stopped using the stopping list distributed with Terrier.

Word embeddings were computed using the *word2vec* software package released by Mikolov et al. [21, 23]. Embeddings were then loaded at retrieval time, pairwise similarity was computed and normalised to form the probability distribution $p(u|w)$. While our use of the embeddings was not necessarily efficient, it did allowed us to better control for different settings and to make the embeddings themselves available to others. A more efficient solution is to precompute the $p(u|w)$ distribution offline and then lookup the relevant values at runtime. Other efficiency improvements can be adopted; for example, Blanco et al. [6] provide techniques for improving efficiency by compressing vector embeddings.

The source code used for all methods is made available at `https://github.com/ielab/adcs2015-NTLM`, with word embeddings, parameter files, result files and evaluation files produced in our empirical experiments. These embeddings can be used beyond the scope of this work; e.g., to study differences in language use across corpora, or the impact of parameters settings on the produced word embeddings.

4.1.3 Baseline — Dirichlet Language Model

The Dirichlet language model constituted the baseline method (LM), in line with the previous experiments on translation language models of Karimzadehgan and Zhai [16, 17]. We mimic their experimental methodology: parameters were tuned per-collection and per-topic set (but not on a per-query basis) with MAP as objective measure[5]. For example, for the Dirichlet language model on AP88-89 (and TREC topics 51-200), we report the results for the single value of the smoothing parameter μ that achieved the highest effectiveness on that collection; while for the same retrieval method on WSJ87-92 (and TREC topics 51-200) we report the results for the single value of μ that achieved the highest effectiveness on that collection/topic set combination, i.e., the value of μ for WSJ87-92 may be different from that for AP88-89. Values of μ considered were 100 and those in the range [500, 4000].

4.1.4 Benchmark — Translation Language Model

For the translation language models, we set μ to the same value that provided the highest effectiveness on the baseline language model (thus this value may not have been the optimal for the translation model). Translation language models parameters (i.e., α, s) were tuned with the same methodology used for the baseline language model, i.e., per collection (as done by [16, 17]). Only the top 10 translation terms were considered for retrieval as translations terms beyond the top 10 have been shown to have little influence for TLM-MI based approaches [16].

4.1.5 Neural Translation Language Model

The neural translation language models based on cbow and skipgram estimations rely on the parameters that con-

[5]Except for Medtrack where bpref was used as this was the primary measure for that task.

Corpus	Latent Dimensions	Window Size
AP88-89	100 – 1,000, step = 100	5, 10
WSJ87-92	100 – 1,000, step = 100	5, 10
DOTGOV	200 – 1,000, step = 200	5, 10
MedTrack	100 – 1,000, step = 100	5, 10

Table 3: Parameter values explored in the training of cbow and skipgram word embeddings.

trol the construction of the word embeddings. In the experiments, we tuned the latent dimensionality of the embeddings and the size of the window used to capture contextual information around target terms following the same methodology used for the baseline and the benchmark approaches. Details of the range of values considered during the tuning process are provided in Table 3. Typical values used in computational linguistics tasks are 100-600 for the latent dimensionality of the word embeddings and 5-10 for the context window [21, 23, 12, 25]. The effect of parameter values other than those leading to the best effectiveness for each collection were investigated separately in the sections that follow. We will, however, notice that the parameter values of the neural translation language models have limited effect on their retrieval effectiveness (at least they do not change the main trends observed in the results).

We did not experiment with other parameters of the word embeddings, such as the sampling technique used to dampen the effect of frequent words. Specifically, we used negative sampling (as opposed to hierarchical softmax) with 20 samples and a subsampling of frequent words that discards a word w in the training set with probability $1 - \sqrt{\frac{10^{-4}}{c(w,C)}}$. The number of iterations or epochs used in the stochastic gradient descent optimisation was set to 5. These values are all in line with those used by previous work that experimented with neural language models [21, 23, 6, 12, 25]. We leave the experimentation of alternative settings of parameters for future work. For the neural translation language models, the Dirichlet smoothing parameter μ was set to the same value that provided the highest effectiveness on the baseline language model: as for the benchmark models, this value may not be the optimal for NTLM. In line with the translation language model benchmark [16], the number of translation terms was set to 10. (The influence of the number of translation terms in NTLM is out of the scope of this evaluation and will be subject of future work.)

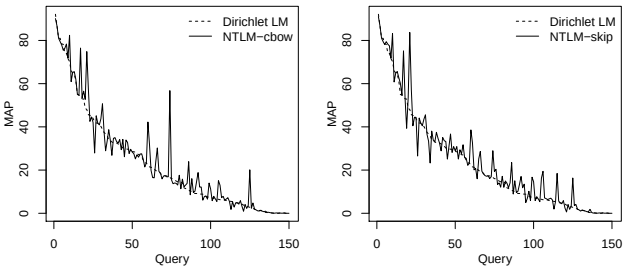
Finally, to answer RQ3 (the effect of different training corpora), we built the word embeddings using three different corpora, AP88-89, WSJ87-92 and Wikipedia1B[6]; these word embeddings were then used in the retrieval system and evaluated against AP88-89 and WSJ87-92.

4.2 RQ1 — Comparing Neural Translation LM with State-of-the-art Translation LM

Retrieval results comparing baseline LM, benchmark TLM and the NTLM are show in Table 4. The per-collection setting of μ are provided in the first row; the best overall results are provided in bold, while the best mutual information based translation model results are provided in italics.

In line with previous work on translation models [16, 17], the benchmark TLM does provide some improvements over the baseline LM. However, these are not always statistically significant and no single TLM is best across test collections: TLM-MI is best for AP88-89 but TLM-MI-s is best for

[6]We provide more details about this dataset in Section 4.4.



(a) LM vs. NTLM-cbow (b) LM vs. NTLM-skipgram

Figure 1: Per-query performance on AP88-89 of NTLM (skigram and cbow) in comparison with Dirichlet language model. Queries are ordered by descending MAP of the LM baseline.

WSJ87-92 (with s=0.9), DOTGOV (with s=0.8) and MedTrack (with s=0.9). Note, these findings differ somewhat from those of [16, 17]. This may be due to two differences in our evaluation: 1) we used 150 topics each for AP88-89 and WSJ87-92, whereas they used 50 each; and 2) we evaluated on two additional test collections (DOTGOV & MedTrack).

For three collections the neural translation model provides the best performance. For AP88-89, the improvements of both the skipgram (dimension=200, window=5) and the cbow (dimension=900, window=10) are statistically significant compared to the LM baseline. For WSJ87-92, the improvements of both the skipgram (dimension=500, window=5) and the cbow (dimension=100, window=10) are statistically significant compared to the LM baseline and the TLM benchmarks. For DOTGOV, the improvements of both the skipgram (dimension=1000, window=5) and the cbow (dimension=200, window=5) have the highest MAP but are not found to be statistically significant. For MedTrack, the improvements on P@10 of both the skipgram (dimension=1000, window=10) and the cbow (dimension=1000, window=10) are statistically significant compared to the LM baseline; however, the best method according to bpref is TLM-MI-s (s=0.9).

While the results for the NTLM were obtained with word embeddings parameter settings that maximised MAP (bpref for MedTrack), we observed that different parameter settings still produced similar improvements over other methods. Sections 4.3 provides a more detailed analysis of this.

4.2.1 *Per-query Analysis of Neural Translation Language Models*

To better understand the trend in gain provided by the neural translation model we consider the performance of individual queries. Figure 1 shows the per-query MAP for the NTLM models against the LM baseline (for brevity, we report only AP88-89, although the same trend was found for other collections); the figure is sorted by descending MAP of the LM baseline. Both subfigures show that the NTLM provides modest improvements on a large number of queries (rather than large differences on only a few queries).

While we do not have space to provide a complete analysis of all cases of success and failure, it is interesting to comment on a couple of example queries. One of the queries that exhibited the largest gains over both the baseline and the benchmark method was query 55 from TREC 1 (gains both in AP88-89 and WSJ87-92). For example, for AP88-89 NTLM-cbow and NTLM-skipgram obtained an average precision of 52.72 and 52.22 respectively, while LM achieved

	AP88-89 ($\mu = 1,000$)		WSJ87-92 ($\mu = 1,500$)		DOTGOV ($\mu = 500$)		MedTrack ($\mu = 3,500$)	
Method	*MAP*	*P@10*	*MAP*	*P@10*	*MAP*	*P@10*	*bpref*	*P@10*
Dirichlet LM	22.69	39.60	21.71	40.80	18.73	24.60	37.69	43.95
TLM-MI	*23.83*^{d}	*41.67*^{d}	20.75	40.73	17.06	22.40	37.02	46.42
TLM-MI-α	22.55	39.73	21.32	40.33	17.15	22.60	37.23	43.70
TLM-MI-s	22.53	39.13	*22.08*	*41.33*	*18.76*	*24.80*	**38.93**	*49.26*^{d}
NTLM-skipgram	**24.27**^{d}	41.00	**22.66**^{d,m}	**42.40**^{d}	**19.32**	**25.00**	38.83	**49.75**^{d}
NTLM-cbow	24.18^{d}	**41.93**^{d}	22.62^{d,m}	42.27^{d}	19.16	24.80	38.77	49.51^{d}

Table 4: Effectiveness of language models with Dirichlet smoothing baseline (*Dirichlet LM*), translation language models with Mutual Information estimates (*TLM-MI*, *TLM-MI-α*, *TLM-MI-s*) and neural translation models, with continues bag-of-words (*NTLM-cbow*) and skipgram (*NTLM-skipgram*). Statistically significant differences, using paired t-test, indicated by ^{d} against *Dirichlet LM* and ^{m} against best *TLM*.

48.61 and TLM-MI 33.07, suggesting that the NTLM models provided high quality translations while those of TLM-MI led to poor estimations and consequently losses in retrieval effectiveness. The translations provided by these methods were already examined in Table 1 and Sections 2.1 and 3.3, and it was shown that cbow and skipgram assigned more probability mass to syntactic variations of the query terms, in contrast to TLM-MI. It was further noted that the difference in magnitude between self-translation probabilities and other probabilities is less for NTLM-cbow and NTLM-skipgram than for TLM-MI.

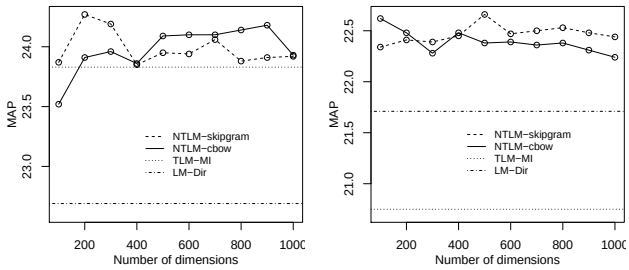
One of the queries where NTLM-cbow and NTLM-skipgram exhibited the largest losses over both the baseline and benchmark methods is query 195 from TREC 3, "stock market perturbations attributable to computer initiated trading". For this query, on AP88-89, NTLM-cbow and NTLM-skipgram obtained an average precision of 14.03 and 12.65 respectively, while LM achieved 15.43 and TLM-MI 21.89. The translations obtained for this query using cbow were characterised by the strong presence of the term "dollar" as translation of stock, market (with high probability estimate), and trading. The word did not appear to be a good discriminator of relevance for documents associated to this topic, and its presence as a translation for three of the original query terms may have induced excessive weight to be assigned to non relevant documents that contained this non-discriminating term. Indeed, when dollar was removed from the translations provided by cbow the effectiveness of NTLM-cbow did improve. While the skipgram embeddings shared a number of translation terms with both TLM-MI and the cbow embeddings (but not "dollar"), some of the translations that were derived resulted in documents with irrelevant content being retrieved. For example, "video" and "chip" were provided as translations of "computer" (while translations provided by TLM-MI and cbow were somewhat more "corporate oriented"). These translations, although valid in general, were certainly not related to the topic of the query.

4.3 RQ2 — Sensitivity to how Word Embeddings are Built

4.3.1 The Effect of Embedding Dimensionality

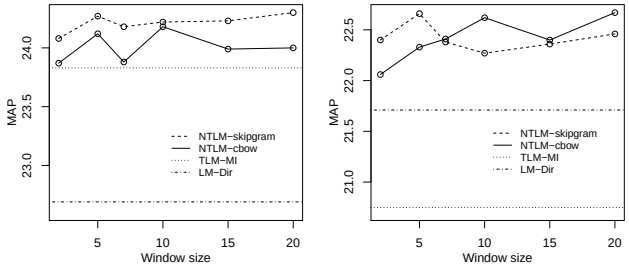
Each word in the neural language model is represented by a vector of a certain dimensionality in the latent space. To study the effect of the number of dimensions, the window size was fixed (to the best setting) and dimensionality altered to determine its effect on MAP. Figure 2 shows the effect of dimensionality on MAP for AP88-89 and WSJ87-92 (other collections excluded for brevity). NTLM was robust according to number of dimensions, with only a single setting of NTLM-cbow on AP88-89, where dimensions=100, produced a MAP below that of the TLM benchmark.

4.3.2 The Effect of Window Size



(a) AP88–89 (b) WSJ87–92

Figure 2: Effect on MAP of different dimension size used to build word embeddings. Choice in dimension size did not significantly affect effectiveness.



(a) AP88–89 (b) WSJ87–92

Figure 3: Effect on MAP of different window size settings used to build word embeddings. Choice in window size did not significantly affect effectiveness.

Word embeddings were created by considering co-occurrence statistics according to a context window of a specific size. To study the effect of window size, the dimensionality was fixed (to the best setting) and window size altered to determine its effect on MAP. Figure 3 shows the effect of window sizes 2, 5, 7, 10, 15 and 20 on MAP. Window size 2 was used to test wether very tight co-occurrence (bigrams) generates word embeddings that focus more on the local context; window sizes 15 and 20 were built to study the influence of distant relationships in the construction of the embeddings and ultimately on retrieval effectiveness. As with dimensionality, the NTLMs were robust according to window size. The NTLM-skipgram is more effective with smaller window sizes (w=5) than the NTLM-cbow (w=10). Overall, window size has more of an effect on cbow than skipgram.

4.3.3 The Effect of Word Embedding Type

Word embeddings can be built in two ways: cbow and skipgram. Overall, the retrieval results showed no statistical significant differences between the two (some collections are better with one model and some with the other, although the differences are minor). Everything else being equal, cbow is

computationally less expensive than skipgram in terms of time (space is the same). However, the previous sections revealed that skipgram generally performed better with less dimensions and smaller context windows; thus skipgram with these parameter settings was computationally more efficient (time and space).

Considering the effect of both number of latent dimensions and window size, the neural translation language models are robust to the choice of parameter settings. In light of this, other forms of parameter tuning, e.g. cross-validation, would not have influenced largely the results and trends observed in our empirical analysis.

4.4 RQ3 — The Effect of Training Corpus

Word embeddings can be built prior to both indexing and retrieval of a corpus and, in fact, do not have to be built using the corpus used in retrieval. The choice of corpus used to build the word embeddings may affect retrieval results. To study this effect, we conducted two sets of experiments.

First, word embeddings were derived from AP88-89 but were evaluated on WSJ87-92, and vice versa. The two corpora contain newswire articles that refer to the same period and thus are comparable. In addition, they are evaluated on the same set of topics, both in the original TREC campaigns and in our experiments.

Second, word embeddings were derived from a dump of Wikipedia[7], containing the first 1 billion words that are commonly used in experiments in computational linguistics. These experiments provided an initial evaluation of the feasibility of using a general purpose corpus (like Wikipedia) to generate word embeddings, rather than creating embeddings on each specific corpus.

The results of this cross-corpus evaluation are shown in Table 5. The grey shaded cells indicate when the same corpus is used for building the word embeddings and for retrieval; bold values indicate the corpus that produced the best performance for each of the NTLM model. The results show that the choice of corpus used to construct word embeddings had little effect on retrieval results. Even when evaluating on newswire collections, the NTLM models were effective when word embeddings are built from Wikipedia articles. This demonstrates the general purpose nature of the distributed word representations provided by these types of neural language models. Such representations could, potentially, be built from general sources of language (e.g., Wikipedia) and be made available as a service, independent of any particular retrieval experiment or model.

5. RELATED WORK

Translation language models were first proposed by Berger and Lafferty, inspired by approaches in automatic machine translations [5]. Karimzadehgan and Zhai have subsequently investigated translation language models based on mutual information, showing their superiority with respect to both standard smoothed language models and the original estimations of Berger and Lafferty [16].

The retrieval mechanism of translation language models initiates a transfer of probability mass from non-query terms to query terms. This probability kinematics resembles the underlying idea introduced by Crestani et al. in their Logical Imaging retrieval model [10]. However, in Logical Imaging probability mass is transferred from terms not in the document to terms in the document, as opposed to the transfer from non-query terms to query terms that takes place in translation language models. Logical Imaging has been shown to be ineffective for modern ad-hoc IR tasks [31].

Beyond translation language models, numerous other approaches have been investigated to incorporate semantic information within the retrieval process and thus go beyond retrieval based on simple (query) keyword matching. While it is impossible to provide a fair and complete account of all methods, we briefly mention some notable pointers to important and recent milestones in this context.

Query expansion [8] is the process of adding terms to a seed query to improve its retrieval effectiveness: conceptually, the automatic expansion process shares the same key intuition as translation language models. Automatic query expansion based on crafted linguistic resources such as WordNet have not led to substantial improvements in effectiveness [29]. Data-driven automatic query expansion has, however, been effective (e.g., [2]), although research has highlighted that such techniques are often optimised to perform well on average, yet their effectiveness is unstable across queries and, for a portion of queries, automatic query expansion may be detrimental [9].

Graph-based models, such as Turtle and Croft's [28] inference network, have been used to define inference mechanisms to augment the retrieval process. Extensions of this approach have progressively increased the amount of semantic information leveraged to extend search beyond keyword matching (e.g., [3]). The recent work of Dalton et al. [11] demonstrates how query representations can be enriched with features from semantic annotations and their links to knowledge bases, such as FreeBase.

The neural translation language models proposed in this paper exploit recent advances in word representations, in particular continuous bag-of-words and skipgram models [21]. Levy & Goldberg have noted similarities between these models and matrix factorisation [19]. However, [21] indicates that linear relationships between word vectors derived from the embeddings generated by cbow and skipgram do not hold for simpler models like latent semantic analysis [13], latent Dirichlet allocation [7] or vectors using tf-idf features.

The process used to construct word embeddings with cbow and skipgram also bear some resemblance with the process used in the hyperspace analogue to language (HAL) [20] and probabilistic HAL [1] models. These have been also used for retrieval with certain success [1, 27]. However, in HAL, co-occurrences within a window centred around a target term are accumulated to form the vector representations; while in cbow and skipgram the representation of a target term is fitted to predict the representations of its lexical context (skipgram), and vice versa (cbow). Despite this, it is not clear yet whether these neural inspired models are generally better than traditional distributional semantic methods.

6. KEY CONTRIBUTIONS AND FINDINGS

1. Theoretically, we provide a means of incorporating neural language models within a retrieval model based on the translation language framework — a neural translation language model. The model captures implicit semantic relationships — via word embeddings — between terms.
2. Empirically, the neural translation language model is statistically significantly better than baseline language mod-

[7] http://mattmahoney.net/dc/enwik9.zip

NTLM	Word Embeddings Corpus	AP88-89 MAP	AP88-89 P@10	AP88-89 P@20	AP88-89 Rel Retr	WSJ87-92 MAP	WSJ87-92 P@10	WSJ87-92 P@20	WSJ87-92 Rel Retr
skipgram	AP88-89	24.27	41.00	37.53	**9483**	22.43	42.20	**37.70**	8744
	WSJ87-92	**24.31**	**41.87**	37.20	9329	**22.66**	**42.40**	37.67	**8967**
	Wikipedia 1B	24.09	41.67	**36.87**	9250	22.50	42.13	37.33	8762
cbow	AP88-89	**24.18**	**41.93**	**37.63**	**9331**	22.05	41.33	36.73	8542
	WSJ87-92	23.65	40.80	36.87	9097	**22.62**	**42.27**	37.37	**8728**
	Wikipedia 1B	23.91	41.00	36.37	9208	22.19	**42.27**	**37.57**	8565

Table 5: Retrieval results when embeddings were constructed on a different corpus to that used in retrieval. The results show that no there are no statistically significant differences when a different corpus was used to construct word embeddings, even when different types of corpora (e.g., newswire vs. Wikipedia) were used.

els (Dirichlet) and comparable to (when not better than) benchmark translation language models.

3. We contribute an understanding of the impact that choices made when building word embeddings have on retrieval effectiveness. Specifically, retrieval is robust with respect to choices in embedding dimensionality and window size.
4. We empirically show that word embeddings do not even need to be produced from the same corpus used for retrieval. Even when different types of corpus (e.g., newswire vs. Wikipedia articles) are used, there is no statistically significant degradation in retrieval effectiveness.
5. We contribute back to the research community both our implementation of neural translation language models and the large number of word embeddings computed to explore our research questions. These embeddings could be further used by the community, including applications outside information retrieval.

7. WIDER IMPACT AND FUTURE WORK

The neural language models investigated here do provide valid and useful translations. However, these are done independently of the specific context of the query. Indeed, our analysis of certain queries highlighted that retrieval could be harmed when translating to valid yet out-of-context terms (e.g., translating "stock" to "dollar"). Clearly taking into account the context of the query would be desirable. Neural language models do in fact provide a possible mechanism to do this via effective representations of phrases [23]. Through simple arithmetic operations on vectors, translation can be considered not only for single terms but also by including term compositions, up to treating the entire query as a single phrase vector. Indeed, one of the advantages of these neural language models is that they provide simple, efficient methods to operate on terms based on simple arithmetic operations on vectors. These powerful yet simple semantic operations can be incorporated into retrieval to better model, among others, query terms dependencies, named-entities and expressions.

8. REFERENCES

[1] L. Azzopardi, M. Girolami, & M. Crowe. Probabilistic hyperspace analogue to language. In *SIGIR'05*, pg. 575–576, 2005.

[2] J. Bai, D. Song, P. Bruza, J.-Y. Nie, & G. Cao. Query expansion using term relationships in language models for information retrieval. In *CIKM'05*, pg. 688–695, 2005.

[3] M. Bendersky, D. Metzler, & W. B. Croft. Learning concept importance using a weighted dependence model. In *WSDM'10*, pg. 31–40, 2010.

[4] Y. Bengio, R. Ducharme, P. Vincent, & C. Jauvin. A neural probabilistic language model. *JMLR*, 3:1137–1155, 2003.

[5] A. Berger & J. Lafferty. Information retrieval as statistical translation. In *SIGIR'99*, pg. 222–229, 1999.

[6] R. Blanco, G. Ottaviano, & E. Meij. Fast & space-efficient entity linking in queries. In *WSDM'15*, pg. 179–188, 2015.

[7] D. M. Blei, A. Y. Ng, & M. I. Jordan. Latent Dirichlet Allocation. *JMLR*, 3:993–1022, 2003.

[8] C. Carpineto & G. Romano. A survey of automatic query expansion in information retrieval. *ACM CSUR*, 44(1):1, 2012.

[9] K. Collins-Thompson. Reducing the risk of query expansion via robust constrained optimization. In *CIKM'09*, pg. 837–846, 2009.

[10] F. Crestani & C. J. van Rijsbergen. Information retrieval by logical imaging. *J. Doc.*, 51(1):3–17, 1995.

[11] J. Dalton, L. Dietz, & J. Allan. Entity query feature expansion using knowledge base links. In *SIGIR'14*, pg. 365–374, 2014.

[12] L. De Vine, G. Zuccon, B. Koopman, L. Sitbon, & P. Bruza. Medical semantic similarity with a neural language model. In *CIKM'14*, pg. 1819–1822, 2014.

[13] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, & R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[14] T. Edinger, A. M. Cohen, S. Bedrick, K. Ambert, & W. Hersh. Barriers to retrieving patient information from electronic health record data: failure analysis from the TREC Medical Records Track. In *AMIA*, volume 2012, pg. 180–188. 2012.

[15] Y. Goldberg & O. Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[16] M. Karimzadehgan & C. Zhai. Estimation of statistical translation models based on mutual information for ad hoc information retrieval. In *SIGIR'10*, pg. 323–330. ACM, 2010.

[17] M. Karimzadehgan & C. Zhai. Axiomatic analysis of translation language model for information retrieval. In *ECIR'12*, pg. 268–280, 2012.

[18] B. Koopman & G. Zuccon. Why assessing relevance in medical IR is demanding. In Workshop on Medical Information Retrieval (MedIR), Gold Coast, Australia, July 2014.

[19] O. Levy & Y. Goldberg. Neural word embedding as implicit matrix factorization. In *NIPS*, pg. 2177–2185, 2014.

[20] K. Lund & C. Burgess. Hyperspace analogue to language (hal): A general model semantic representation. In *Brain & Cognition*, volume 30. Academic Press Inc, 1996.

[21] T. Mikolov, K. Chen, G. Corrado, & J. Dean. Efficient estimation of word representations in vector space. In *Workshop at ICLR*, 2013.

[22] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, & S. Khudanpur. Recurrent neural network based language model. In *InterSpeech*, pg. 1045–1048, 2010.

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, & J. Dean. Distributed representations of words & phrases & their compositionality. In *NIPS*, pg. 3111–3119, 2013.

[24] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, & C. Lioma. Terrier: A High Performance & Scalable Information Retrieval Platform. In *OSIR'06*, 2006.

[25] J. Pennington, R. Socher, & C. D. Manning. Glove: Global vectors for word representation. *EMNLP'14*, 2014.

[26] H. Schwenk & J.-L. Gauvain. Training neural network language models on very large corpora. In EMNLP'05, pg. 201–208, 2005.

[27] D. Song & P. Bruza. Discovering information flow using high dimensional conceptual space. In *SIGIR'01*, pg. 327–333. 2001.

[28] H. Turtle & W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM TOIS*, 9(3):187–222, 1991.

[29] E. M. Voorhees. Query expansion using lexical-semantic relations. In *SIGIR'94*, pg. 61–69, 1994.

[30] C. Zhai & J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM TOIS*, 22(2):179–214, 2004.

[31] G. Zuccon, L. Azzopardi, & C. J. van Rijsbergen. Revisiting logical imaging for information retrieval. In *SIGIR'09*, pg. 766–767, 2009.