Event-Driven Query Expansion

Guy D. Rosin
Technion
Haifa, Israel
guyrosin@cs.technion.ac.il

Ido Guy
eBay Research
Netanya, Israel
idoguy@acm.org

Kira Radinsky
Technion
Haifa, Israel
kirar@cs.technion.ac.il

ABSTRACT

A significant number of event-related queries are issued in Web search. In this paper, we seek to improve retrieval performance by leveraging events and specifically target the classic task of query expansion. We propose a method to expand an event-related query by first detecting the events related to it. Then, we derive the candidates for expansion as terms semantically related to both the query and the events. To identify the candidates, we utilize a novel mechanism to simultaneously embed words and events in the same vector space. We show that our proposed method of leveraging events improves query expansion performance significantly compared with state-of-the-art methods on various newswire TREC datasets.

CCS CONCEPTS

• **Information systems → Query reformulation**; *Digital libraries and archives*; • **Computing methodologies** → *Lexical semantics.*

KEYWORDS

query expansion; temporal semantics; word embeddings

ACM Reference Format:
Guy D. Rosin, Ido Guy, and Kira Radinsky. 2021. Event-Driven Query Expansion. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441833

1 INTRODUCTION

One of the common practices in Information Retrieval (IR) is the enrichment of search queries with additional terms [3, 7]. This approach is called query expansion (QE) and is intended to narrow the gap between the query and the corpus language, to increase retrieval's recall. In this work, we focus on the task of QE for event-related queries.

A significant number of event-related queries are issued in Web search. Based on our analysis over an AOL query log [30], 36% of Web search queries are related to events (evaluation method described in Section 5.1). While several studies have focused on detecting such queries [15, 19, 46], no attempt has been made to leverage event representations for QE.

While some event-related queries explicitly include a temporal expression [28] (e.g., "2020 pandemic" or "US election 2016"), most are implicitly associated with a particular time period [20] (e.g., "Black Monday" or "Tiananmen Square protesters"). In addition, queries can be relevant to either a single event ("Schengen agreement") or multiple events ("computer viruses").

We suggest adding an intermediate step in query expansion of identifying related events. Our hypothesis is that it can help determine the right time span and context for search results, and thus help guide the query expansion. Detecting the right events is a challenging task for several reasons. First, queries are usually very short and may contain ambiguous terms. For example, given the query "Bank Failures", our method detected the event "2000 Camp David Summit", which discussed the West Bank territory and is considered a failure.[1] Second, events can be relevant to entities in the query, but not to the query intent. As an example, given the query "Greenpeace", our method detected "Sinking of the Rainbow Warrior" (a bombing of Greenpeace's flagship). This event is indeed relevant to Greenpeace, but it does not fit the query intent and thus it does not provide a relevant context for query expansion.

We propose a method to expand an event-related query by first detecting the events related to it. Then, based on the detected events, we derive the candidates for expansion as terms semantically related to both the query and the events. Figure 1 shows an example: given the query "African civilian deaths", our method detects the following events: *War in Darfur*, *Somali Civil War*, and *South Sudanese Civil War.* Then, it suggests these top expansions: 'armed', 'sudan', 'insurgent', 'rebel', and 'uganda'. These are all highly relevant events and expansions for the query. On the other hand, a baseline method [17] produced as top expansions names of African nationalities: 'sudanese', 'rhodesian', and 'ugandan'.

We present a novel technique for embedding both words and events in the same vector space (Section 3.2). Previous work has often used large static corpora, such as Wikipedia, to represent words [14, 25, 40]. Similarly, events can be represented by the embedding of their corresponding article in Wikipedia [9, 14]. However, events occur at a specific time, hence their semantics is true only for that time [35]. Static representations, such as Wikipedia, capture a description of the event in perspective to today, thus losing their temporal representation. For example, a major part of World War II's Wikipedia entry[2] is dedicated to periods before or after the war itself (i.e., pre-war events, its aftermath and impact). We therefore conjecture that the use of temporal embeddings would enable to better capture the interaction between the event and the QE candidates. We suggest to project static event embeddings onto a temporal corpus (38 years of the *New York Times*). This allows tracking words and events dynamics over time (Section 3.1), to

WSDM '21, March 8–12, 2021, Virtual Event, Israel

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441833

[1] https://en.wikipedia.org/wiki/2000_Camp_David_Summit
[2] https://en.wikipedia.org/wiki/World_War_II

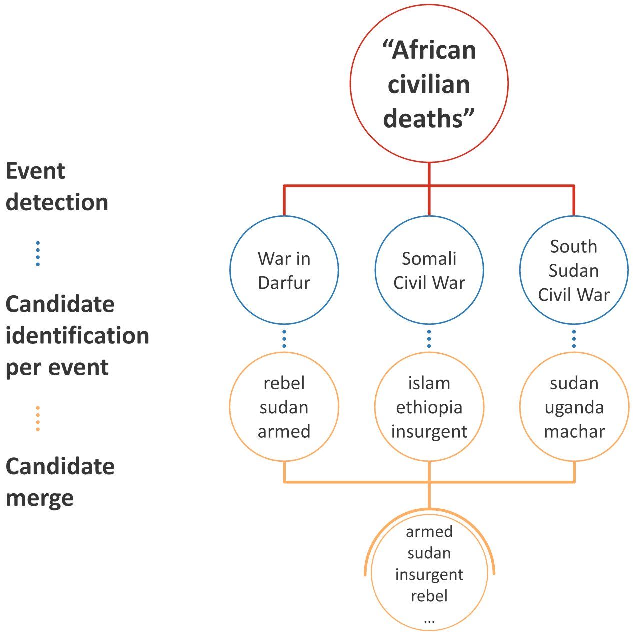

Figure 1: Pipeline of event-driven query expansion, with the example query of "African civilian deaths".

identify potential candidates for expansion. Intuitively, we identify words whose semantics is influenced by the event and hypothesize that they serve as better expansion candidates for the query. Since the proposed projection method is applied as a preprocessing step, our QE method is suitable for a real-time search scenario. Our empirical results over the global analysis approach for automatic query expansion [3], show high performance gains compared with state-of-the-art query expansion algorithms [17, 21, 29, 36].

The contributions of this paper are threefold: First, we propose an approach for query expansion by adding an intermediate step of identifying events related to the query (Section 4.1). Those, in turn, are used to identify more precise query expansion candidates (Section 4.2). Second, we present a novel mechanism to simultaneously embed words and events in the same space (Section 3.1). The embeddings are used to identify expansion candidates whose semantics was influenced by the events related to the query. Third, we show that our proposed method of leveraging events improves QE performance significantly (Section 6). We also publish our code and data.[3]

2 RELATED WORK

2.1 Query Expansion

Query expansion (QE) is the process of adding relevant terms to a query to improve retrieval performance. In recent years, most QE techniques employed word embeddings [3, 7]. Kuzi et al. [21], Roy et al. [36], and Zamani and Croft [42, 43] used word embeddings to select terms semantically related to the query as expansions. Diaz et al. [12] proposed to train embeddings on the top retrieved documents for each query. Rosin et al. [34] leveraged the changing relatedness between words to improve QE for temporal queries. Zamani and Croft [44] proposed relevance-based word embeddings and thus matched the embedding algorithm's objective to that of IR tasks (i.e., to capture relevance). External data sources (e.g., Wikipedia and WordNet) have also been utilized for QE [2, 27]. Recently, Imani et al. [17] used a deep neural network classifier to guide the expansion selection process, and Padaki et al. [29] showed that using the topic descriptions of TREC collections can improve pretrained reranking models.

In this paper, we perform QE on a specific type of queries, i.e., event-related queries. Moreover, this is the first attempt to leverage events for QE, to the best of our knowledge.

2.2 Temporal IR and Events

Temporal information retrieval is a subfield of IR whose purpose is to improve IR by exploiting temporal information in documents and queries [6, 18, 31]. Temporal information has been utilized for query expansion [34] and query auto-completion [38], among other tasks. To leverage events for IR tasks, it is often necessary to represent events as continuous vectors (i.e., embeddings). Most approaches treat events as textual data and apply text embedding methods [25, 35, 39], while others involve leveraging knowledge graph data [13], or using networks and random walks techniques [37]. The latter has also been used for creating news embeddings [22].

In this work, we embed events using text embedding methods, together with a novel method to combine events and words in a joint vector space.

2.3 Leveraging Events for Search

It has been reported that 11% of web search queries are categorized as news [4]. There have been several attempts to focus on the news domain for search and specifically to leverage events. They all employed supervised learning to detect event-related queries. Kanhabua et al. [19] found query-event candidates using a query log and then determined whether each one is an event using a classifier. Ghoreishi and Sun [15] used a classifier with 20 features based on top-ranked search results and a query log. Finally, Zhang et al. [46] published a follow-up study using a similar method with additional features.

All the above studies utilized a query log as the main (if not the only) source of features. In contrast, in our work, we do not rely on query logs but use Wikipedia and DBpedia as sources of events that are open and publicly accessible.

3 EVENT AND WORD EMBEDDING

To expand an event-related query, we detect the events related to that query, and then, based on the detected events, we derive the candidates for expansion. Both steps require a representation of the words and events. Previous work has often represented words using a large static corpus, such as Wikipedia [25, 40]. Similarly, events can be represented by the embedding of their corresponding article in Wikipedia [9]. We refer to those embeddings as *static embeddings*. Alternatively, using *temporal embeddings* allows us to compare words and events over time and focus on the semantics of specific time periods (e.g., given an event, focus on the time period

[3] https://github.com/guyrosin/event_driven_qe

in which it occurred). We conjecture that utilizing these embeddings would enable us to capture better the interaction between the event and the query expansion candidates.

3.1 Temporal Embeddings

Temporal word embeddings were presented in prior work [11, 16, 34, 41]. The authors created the embeddings using data from a large temporal corpus (e.g., the New York Times (NYT) archive, or Google News). We specifically use the NYT archive. The embeddings are generated for each time period (i.e., year) and enable us to examine how words meanings and relatedness between words change over time (see Section 5.3).

However, these temporal models may not have meaningful embeddings of events [35]. We found only 30% of the event names in our dataset (Section 5.2) to appear (at least once) in their corresponding temporal corpus, and only 12% of them to have at least 10 occurrences. For example, the string "Prestige oil spill" appears once in the NYT corpus of 2002 (when the spill occurred). The search was performed with case ignored, and explicit temporal expressions removed from the event names (e.g., "1989 Tiananmen Square protests" was replaced with "Tiananmen Square protests"). Moreover, it may take some time until an event's name is determined and referred to in newspapers. For example, the name "World War I" was used only after WWII started. To conclude, most events do not appear enough times in the temporal corpus to allow their embedding in a vector space. As a result, we are unable to compare events and words.

We therefore present a methodology of representing events in temporal embeddings. We transform each temporal model into a joint vector space for words and events, representing a specific time period. We refer to this process as *Event Projection*.

In this work, we use this procedure offline, to enrich the temporal models with events (see implementation details in Section 5.3). We henceforth refer to "temporal embeddings" as the enriched temporal embeddings, so that each model of time t would include embeddings of events that occurred in t.

3.2 Event Projection

Static embeddings allow creating a common latent space for *both* words and entities (and specifically, events), allowing us to compare both. However, they lack the ability to compare words and events *over time* and focus on the semantics of specific time periods. We therefore suggest projecting the event static embeddings into a temporal snapshot. The event projection method enables us to embed events inside the temporal models.

We present a method based on trilateration—a method used to locate an object in space, using measurements to three known objects. For example, consider the event *World War II*. To project it into a temporal model, we leverage the fact that it is embedded in the static model. In that model, we can identify *World War II*'s neighbors (e.g., 'germany', 'hitler', and 'war'). Intuitively, the relationship between *World War II* and its neighbors in the static model should be similar to that in the temporal model. Exploiting this intuition enables us to position it in the temporal model (Figure 2).

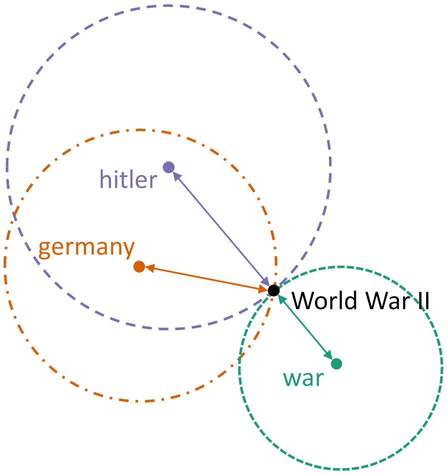

Figure 2: Trilateration illustration. The position of the event *World War II* can be found by using its distance to the neighboring words *war*, *germany*, and *hitler*.

Technically, let M_t be a temporal word embedding model of time t, and $e \notin M_t$ be an event. Our goal is to embed e in M_t. Algorithm 1 formally describes the method.

Let *Wiki* be the static Wikipedia model. It contains embeddings of events, so $e \in Wiki$. In line 1, we take as *anchors* e's top k neighbors in *Wiki*, which also exist in M_t. To identify the neighbors, in line 2 we use a distance function *dist* to calculate the distance of each anchor n to e in *Wiki*. Specifically, we use cosine distance as *dist*. Intuitively, since the anchors also exist in M_t, we assume that e should be located somewhere between them also in M_t, according to the calculated distances.

Technically, finding the best position of e in M_t is an optimization problem, where we seek a position $\hat{v}$ whose distances to the anchors are similar to those of e. Specifically, we minimize the mean squared error (MSE) of the distances of the anchors to e in *Wiki*, and to v in M_t (line 3). We use L-BFGS [5] as the optimization algorithm.

Algorithm 1: Event projection algorithm

```
Input: $M_t$ (source embedding model)
Input: $Wiki$ (embedding model of Wikipedia)
Input: $e$ (event, $e \notin M_t$)
1 $anchors \leftarrow \{n : n \in kNN_{Wiki}(e) \ \wedge \ n \in M_t\}$
2 $D \leftarrow \{dist(Wiki(e), Wiki(n)) : n \in anchors\}$
3 $\hat{v} \leftarrow \arg\min_v \{MSE(D, \{dist(v, M_t(n)) : n \in anchors\})\}$
4 $M_t(e) \leftarrow \hat{v}$
```

Finally, although we described the projection method specifically for our use case for readability, in fact it is a generic method for projecting embeddings from a source model to a target model; it can be applied to any two embedding models, not necessarily Wikipedia and temporal models. In addition, the projection method does not require the two embedding models to have the same dimensionality, unlike existing mapping methods such as Orthogonal Procrustes [8].

4 EVENT-RELATED QUERY EXPANSION

Let q be a query defined by a bag of terms $q = \{w_1, w_2, \ldots, w_n\}$. Our goal is to expand it with semantically related words. Our expansion method is composed of several steps, as depicted in Figure 1. We first detect related events to q (Section 4.1), then find expansion candidates based on each event (Section 4.2), and finally merge them into an expansion for q.

We use the following notations throughout the paper:

Notation 4.1. cos is cosine similarity, which we use as a similarity function between embeddings.

Notation 4.2. $\cos(w_1, w_2)$ is the similarity between w_1 and w_2 using the static model.

Notation 4.3. $\cos_t(w_1, w_2)$ is the similarity between w_1 and w_2 during time t (i.e., using the temporal model of time t).

Notation 4.4. $\mathit{tfidf}(w, e)$ is the TF-IDF score of the word w in the event e's Wikipedia entry.

4.1 Event Detection

To detect events related to a query q, we start by detecting events for each query term $w{\in}q$ separately, as a set of events, E_w. The set is composed of the top-scoring events above a predefined threshold *min_score*. We consider two scoring functions for a candidate event e. The first is embedding-based (called *Similarity*): we calculate the similarity between w and e using a word embedding model. Technically, it can be either $\cos(w, e)$ if we use the static model, or $\cos_t(w, e)$ if we use the temporal model of e's time t. The second score is text-based (called *Frequency*): we calculate the frequency of w in e's Wikipedia entry: $\frac{\#w\text{'s occurrences in } e}{\#\text{words in } e}$. We stem the words and require w itself to appear at least twice in e's entry.

Now, the set of events for q is defined as the set of events that were detected for most query terms:

$$E_q = \{e : e{\in}E_w \text{ for most of } w{\in}q\}$$

Finally, to retain only significant events, for every time t we remove from E_q events with a relatively low score. Technically,

$$E_q = \bigcup_t \left\{e \in E_q^t : \mathit{score}(e) > \mu \cdot \max(\{\mathit{score}(e) : e \in E_q^t\})\right\}$$

where *score* is a scoring function, μ is a parameter, and E_q^t is the set of events that occurred in time t. All parameter values are reported in Section 5.4.

4.2 Event-Driven Expansion

Once related events E_q have been detected, we turn to find expansion terms based on them, as a set C_q. First, for each detected event $e \in E_q$, we create a set of candidate terms, C_e. We examine two types of candidates: terms similar to e, and terms similar to q. We use a combination of them to compose C_e: λk top terms by $\mathit{tfidf}(c, e)$, and $(1 - \lambda)k$ top terms by $\cos(c, q)$ (where c is a term and λ is a parameter).

Second, each candidate term $c \in C_e$ is given a score. We examine two scoring options:

Static variant. We use the static embeddings and the following equation to score each candidate:

$$\mathit{score}_S(c) = \alpha \cdot \mathit{tfidf}(c, e) + \beta \cdot \cos(c, e) + \gamma \cdot \cos(e, q)$$

where α, β, and γ are parameters. We refer to our QE method that uses this variant as *SED* (for ***S**tatic **E**vent-**D**riven Method*).

Temporal variant. We utilize the temporal embeddings instead of the static embeddings. This variant works similarly to the static one, with two differences. First, the similarities are calculated using temporal embeddings (i.e., given an event e we use the temporal model of e's time t). Second, we add a new feature, called *temporal relevance* (*TempRel*) that prioritizes terms that got closer to e's top neighbors during e's time t. It is calculated as the average amount a candidate c got closer to e's neighborhood:

$$\mathit{TempRel}(c, e) = \frac{1}{k} \sum_{n \in \mathit{kNN}(e)} \frac{\cos_t(c, n)}{\cos_{t-1}(c, n)}$$

where $n \in \mathit{kNN}(e)$ and k is a parameter. Intuitively, it helps us identify terms whose semantics was influenced by the event. The equation to score each candidate c is:

$$\mathit{score}_T(c) = \alpha \cdot \mathit{tfidf}(c, e) + \beta \cdot \cos(c, e) + \gamma \cdot \cos(e, q) + \delta \cdot \mathit{TempRel}(c, e)$$

where α, β, γ, and λ are parameters. We refer to our QE method that uses this variant as *TED* (for ***T**emporal **E**vent-**D**riven Method*).

Finally, the candidates are merged over all events: $C_q = \bigcup_{e \in E_q} C_e$ and the top-scoring terms (by the chosen score from above) are chosen as the actual expansion terms.

5 EXPERIMENTAL SETUP

All retrieval experiments were carried out using the Terrier search toolkit [23], with 100 expansion terms (selected based on a separate validation set, from $\{10, 20, \ldots, 150\}$).

5.1 Event-Related Query Classification

We perform our experiments on event-related queries. To define event relatedness, we consider both explicit relatedness (e.g., "tropical storms") and implicit relatedness (e.g., "police deaths", due to violent protests). We present an automated method for identifying such queries. In practice, this will allow search engines in real time to identify queries for which the event-driven QE should be applied.

A query is classified as event-related if most of its terms w have a frequency $\frac{\#w\text{'s occurrences in } e}{\#\text{words in } e} > 0.001$ in any event's Wikipedia entry e. In addition, we randomly sampled 25% of the queries and manually annotated them by a group of five annotators; they were given various examples of both cases and instructed to mark a query as event-related if they could think of an event that is related to the query, either explicitly or implicitly. Inter-annotator agreement over all queries, measured using Fleiss's kappa, was 0.76. This annotation process deemed that the precision of the method is 68%–76% and the recall is 94%–98% (performance varies among the datasets).

5.2 Data

We explore the effectiveness of our proposed QE method on the standard ad-hoc retrieval task using several TREC collections. Since events are mostly relevant to news, we use newswire datasets. First,

Table 1: TREC datasets used for experiments.

Collection	TREC Disks	# of Docs	# of Topics
Robust	Disks 4,5-CR	528,155	250
TREC12	Disks 1–2	741,856	150
AP	Disks 1–3	242,918	150
WSJ	Disks 1–2	173,252	150

we use Robust (TREC Robust Track 2004 collection, which consists of Tipster disks 4 and 5) and TREC12 (consists of Tipster disks 1 and 2). Second, we use two smaller and more focused datasets, each containing documents from a single newspaper: AP (Associated Press) and WSJ (Wall Street Journal). See Table 1 for details. Finally, we include in the evaluation an artificial collection of queries (*ALL*), composed of all the queries in the above datasets combined. We use the TREC topic titles as queries to the retrieval models.

Event data. We used DBpedia[4] (an open, structured knowledge base derived from Wikipedia) as a source for events and followed the methodology of Rosin and Radinsky [35] to mine events. We mined DBpedia entities whose type is 'event'[5] and that have an associated Wikipedia entry and year of occurrence. We retained events corresponding to our focus time period of 1981 to 2018. Finally, to reduce the noise of insignificant events, we retained events with over 5000 monthly views (on average, between 2018–2020) and over 15 external references. Our final dataset contains 2354 events, most related to armed conflicts, politics, disasters, and sports.

5.3 Indexing and Word Embedding Models

We indexed the collections using the Terrier search toolkit, with its default setting. We used Porter stemmer for stemming and removed stop-words using the stopping list distributed with Terrier.

For the static embeddings, we used a pre-trained Wikipedia model from Yamada et al. [39]. It was created based on the English Wikipedia (dump of April 2018) using word2vec [26], with a dimensionality of 100 and a window size of 5. It was designed to include embeddings of both words and entities (and specifically, events) [40]. In particular, the model contains embeddings of all the events in our dataset (Section 5.2).

For the temporal embeddings, we used the NYT archive[6], with articles from 1981 to 2018 (11GB of text in total). For each year of content, we created embeddings using word2vec's skip-gram with negative sampling, with a window size of 5 and a dimensionality of 140, using the Gensim library [32]. We filtered out words with fewer than 50 occurrences during that year. The temporal embeddings were then enriched with events using the projection method described in Section 3.1. Each temporal model was enriched with events from our dataset (Section 5.2) that occurred in its time. For the projection process, we use L-BFGS [5] as the optimization algorithm, and $k = 30$ anchors per event (selected empirically from $\{10, 20, 30\}$, based on a separate validation set).

We preferred to use general-purpose corpora (i.e., Wikipedia and NYT) for word embedding, rather than creating embeddings on each specific corpus (as in Diaz et al. [12]), for the sake of generality.

We created a TF-IDF model of the English Wikipedia based on a dump from March 2020, using the Gensim library. This model was used by our event-driven QE method (Section 4.2).

5.4 Compared Methods

Baseline methods. We consider several baselines to evaluate our methods. First, we compare with three state-of-the-art QE methods (all reimplemented): Imani et al. [17], Roy et al. [36], and Kuzi et al. [21]. All these methods leverage word embeddings to yield expansion terms that are semantically close to the query terms. Our implementations of these methods use the static Wikipedia embeddings, for a fair comparison. Imani et al. [17] create a deep neural network for classifying expansion terms based on their effectiveness for query expansion, and use it to re-weight expansion terms. Roy et al. [36] and Kuzi et al. [21] look for expansion terms close to the query in a vector space. Second, we compare with a BERT [10] query expansion method, inspired by Padaki et al. [29], where the BERT-Base pre-trained model [10] is used to find expansion terms close to the query. Finally, we compare with a standard BM25 approach for IR [33], which does not exploit query expansion techniques.

We compare the above baselines with our two event-driven methods, i.e., SED and TED (Section 4). Both use a standard language model with fixed coefficient interpolation [45]: $P(w|q) = \alpha P_{ED}(w|q) + (1-\alpha)P_{ML}(w|q)$ where P_{ED} is the normalized score of w as calculated by the event-driven method, P_{ML} is the maximum likelihood estimation of w, and the interpolation coefficient α is set to 0.6 (as per the setting prescribed in Roy et al. [36]). We retrieve the documents for the expanded query language model using the TF-IDF weighting model (we also checked BM25 which showed similar results and trends, so we do not report its results for brevity).

Parameter setting. The proposed event-driven methods have several hyperparameters. To detect events, we set $min\_score = 0.003$, and $\mu = 0.5$ (Section 4.1). To generate candidate terms per event e, we selected λ from $\{0, 0.1, \ldots, 1\}$. (Section 4.2; also see the relevant analysis in Section 6.1). To score the expansion candidates, we set $\alpha = 3, \beta = 1, \gamma = 1, \delta = 1$. For the *TempRel* feature, we selected $k = 5$ nearest neighbors from $\{5, 10, 20\}$ (Section 4.2). All parameter values are determined based on a separate validation set, where MAP serves as the optimization criterion. To ensure the repeatability of the results, we have released our code and instructions.[7]

Evaluation metrics. Retrieval effectiveness is measured and compared using several standard metrics [24]: precision of the top 10 retrieved documents (P@10), normalized discounted cumulative gain calculated for the top 10 retrieved documents (NDCG@10), and mean average precision (MAP) of the top-ranked 1000 documents. Statistical significance tests were performed using paired t-test at a 95% confidence level.

[4] https://wiki.dbpedia.org

[5] i.e., `yago/Event100029378` or `Ontology/Event`

[6] https://spiderbites.nytimes.com

Table 2: Results of query expansion evaluation. Statistically significant differences with BM25, Roy et al. [36], Kuzi et al. [21], Imani et al. [17], and BERT baselines are marked with ‘b’, ‘r’, ‘k’, ‘i’, and ‘B’, respectively.

Method	Robust			TREC12			AP			WSJ			ALL		
	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP
BM25	0.40^{i}	0.39	0.19	0.42	0.42	0.25	0.29	0.30	0.20	0.36	0.37	0.23	0.41	0.40	0.26
Roy [36]	0.38	0.38	0.19	0.49^{b}	0.50	0.27^{b}	0.33	0.34	0.22^{b}	0.37	0.38	0.25^{b}	0.41	0.41	0.27
Kuzi [21]	0.36	0.37	0.19	0.49	0.49	0.27^{b}	0.35	0.33	0.21	0.37	0.37	0.25^{b}	0.39	0.40	0.27
Imani [17]	0.36	0.37	0.19	0.49	0.49	0.27^{b}	0.35	0.33	0.21	0.37	0.38	0.25^{b}	0.40	0.40	0.27
BERT	0.42	0.41	0.20	0.46	0.46^{b}	0.26^{b}	**0.37**$^{\mathrm{b}}$	0.35	0.21	0.39	0.38	0.24	0.42	0.43^{b}	0.27^{b}
TED	**0.45**$^{\mathrm{rki}}$	**0.45**$^{\mathrm{rki}}$	**0.23**$^{\mathrm{brkiB}}$	**0.53**$^{\mathrm{b}}$	**0.54**$^{\mathrm{b}}$	**0.31**$^{\mathrm{brkiB}}$	**0.37**$^{\mathrm{b}}$	**0.37**	**0.23**	**0.41**	**0.41**	**0.27**$^{\mathrm{bB}}$	**0.47**$^{\mathrm{brki}}$	**0.48**$^{\mathrm{brki}}$	**0.31**$^{\mathrm{brkiB}}$

6 RESULTS

In this section, we outline the results of our empirical evaluation. In all tables throughout the section, the best result in each column is boldfaced, and statistically significant results are marked with ‘*’, unless otherwise specified. Table 2 presents the main results of our experiments, comparing our best expansion method TED with the five baseline methods. The baselines perform similarly, while TED outperforms all of them significantly.

6.1 Analysis

We present a detailed analysis focused on our best performing method TED.

6.1.1 Embedding type. We analyze the contribution of various embedding types to event-driven QE. We compare our methods SED and TED, as the former uses static embeddings, and the latter uses temporal embeddings. Table 3 shows the results. The temporal embeddings have a clear advantage in all datasets and metrics.

6.1.2 Query type. We compare the performance of TED with SED and the baselines on different types of queries. As mentioned in Section 5.1, event-related queries can be divided into two types: single-event related queries (e.g., “Schengen agreement”, or “Tiananmen Square protesters”), and multiple-event related queries (e.g., “computer viruses” and “wrongful convictions”). We manually created sets of events of either type, from all the TREC datasets used in the evaluation, combined (Section 5.2). The queries were labeled by a group of three annotators, who were asked to mark the queries as *single-event related*, *multiple-event related* or *N/A*. The annotators were given guidelines with examples of the different types. Specifically, they were instructed to mark a query as single (multiple)-event related if they can think of a direct connection between the query and a single (multiple) event; and to mark as *N/A* in case the relatedness is vague or non-existing, e.g., for the query “robotics”. 20% and 49% of the event-related queries were labeled as *single-* and *multiple-event related*, respectively; 31% were labeled as *N/A*. Inter-annotator agreement, measured using Fleiss’ kappa, was 0.78.

Table 4 shows the comparison results. TED is superior for both types of queries. Comparing the event-driven methods, TED outperforms SED significantly for multiple-event queries, while for single-event queries it has a much smaller advantage. This is reasonable, as queries that are related to a single event are more specific; their intent is often more focused on the event, so it is relatively easy to identify expansion terms matching the intent. Comparing the two methods with the baselines strengthens this hypothesis; TED outperforms the baselines consistently for both types of queries, while SED is outperformed by the baselines for multiple-event queries. To conclude, TED shows its advantage on the more difficult type of queries, which is related to multiple events.

6.1.3 Individual components. Finally, we analyze the contribution of each of our expansion method’s main components: event detection, expansion candidate composition, and expansion candidate scoring.

Event detection method. In Section 4.1, we suggest two methods for related event detection, namely *Similarity* and *Frequency*. For comparing them, we execute TED with each. Table 5 shows the results: the *Frequency* method performs better for all datasets (significantly, in most cases). We believe the reason for this difference is that word2vec similarity is not precise enough for this task: we observed many false positive events using the *Similarity* method. In contrast, the *Frequency* method guarantees that the detected events include the query words (frequently) in their Wikipedia entry, and is therefore more precise.

Expansion candidate composition. We compare different compositions of expansion candidates, given a query q and a related event e (Section 4.2). Figure 3 plots the sensitivity of TED to the interpolation parameter λ for the Robust dataset (other datasets behave similarly). λ controls the balance between candidates similar to q ($\lambda = 0$) and candidates related to e ($\lambda = 1$). The plot has a positive trend, and the optimal λ is 0.8. We deduce that: (1) terms related to the event are more beneficial to the query expansion than terms similar to the query; (2) using a small number of terms similar to the query is helpful; it diversifies the expansion and acts as a fallback in case of a false positive (i.e., an unrelated event was detected).

Expansion candidate scoring. Table 6 presents a leave-one-out comparison of the features used to score expansion candidates by TED: term-query similarity, term-event similarity, event-query similarity, term-event TF-IDF score, and temporal relevance (Section 4.2). We observe an almost equal performance, i.e., no single feature is more important than any other. Term-query similarity is relatively strong, and on the other hand term-event similarity is relatively weak, but the differences are not statistically significant.

[7] https://github.com/guyrosin/event_driven_qe

Table 3: Comparison of embedding types.

Method	Robust			TREC12			AP			WSJ			ALL		
	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP
SED	0.39	0.38	0.20	0.47	0.48	0.26	0.33	0.36	0.22	0.36	0.37	0.25	0.42	0.41	0.27
TED	**0.45***	**0.45**	**0.23***	**0.53***	**0.54**	**0.31**	**0.37**	**0.37**	**0.23**	**0.41**	**0.41**	**0.27**	**0.47***	**0.48**	**0.31***

Table 4: Comparison of query types. Statistically significant differences with BM25, Roy et al. [36], Kuzi et al. [21], Imani et al. [17], BERT, and SED are marked with 'b', 'r', 'k', 'i', 'B', and 's', respectively.

Method	Single-Event Queries			Multiple-Event Queries		
	P@10	NDCG	MAP	P@10	NDCG	MAP
BM25	0.46	0.37	0.38	0.43	0.41	0.26
Roy [36]	0.46	0.40^{b}	0.39^{b}	0.49^{b}	0.49^{b}	0.29^{b}
Kuzi [21]	0.45	0.39	0.39	0.48	0.50^{b}	0.29^{b}
Imani [17]	0.45	0.39	0.39	0.50	0.50^{b}	0.30^{b}
BERT	0.51	0.44^{b}	0.41^{b}	0.44	0.44	0.28
SED	0.50	0.48^{k}	0.47^{bi}	0.46	0.46^{bB}	0.25
TED	**0.52**^{ki}	**0.52**^{bk}	**0.48**^{brkiB}	**0.55**^{bsB}	**0.53**^{b}	**0.31**^{brkisB}

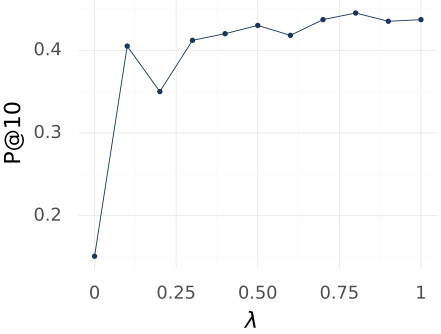

Figure 3: Sensitivity of TED to the interpolation parameter λ (Section 4.2) for the Robust dataset.

To conclude, the contributing factors to TED's performance are detecting the right events and using the right embeddings. In contrast, the choice of candidate scoring features is insignificant.

6.2 Qualitative Examples

To gain more intuition on when and why event-driven QE works, we provide examples with queries from the Robust and TREC12 datasets. For each query, we show the top detected events by TED, top five expansions produced by it, and for comparison—top five expansions produced by a baseline [17].

Positive examples. Table 7 shows successful examples of expansions made by TED. For the query "African civilian deaths", the baseline produced as expansions names of African nationalities, while TED successfully detected several civil wars that occurred in Africa and produced terms related to them.

Similarly, for the query "U.S. invasion of Panama", the baseline produced mainly countries or terms similar to Panama. In contrast, TED produced expansion terms that are more related to the invasion itself, which is closer to the query intent.

The third example, "Tiananmen Square protesters", is easier to expand. The expansions of both methods are similarly focused on the word 'protesters'.

Negative examples. Table 8 demonstrates errors made by TED. The first type of error is detecting wrong events. For example, given the query "Bank Failures", the detected event "2000 Camp David Summit" is not relevant to the query, but was detected nonetheless because (1) the Summit is considered a failure, and (2) the phrase "West Bank" appears frequently in the event's Wikipedia entry. It is interesting to note that despite the wrong detection, the expansions produced by TED are mostly reasonable; just one term out of the top five expansions is not related to the query ('Israel').

Another error TED can make is to produce wrong expansions, based on a relevant event. For the query "Death from Cancer", the baseline produced reasonable cancer-related expansions. TED detects Chernobyl Disaster, which is a relevant event. The expansions are reasonable, but since only one event was detected, the focus on it is considerable; two out of the top five expansions are more relevant to the event than to the query ('reactor' and 'dose').

Similarly, for the query "Greenpeace", the baseline produced reasonable expansions related to the environmental organization. TED detected a relevant event (a bombing operation of Greenpeace's flagship), but the expansions are too focused on the event, and so they are less relevant to the query (e.g., 'bomb', 'yacht', and 'warrior').

7 CONCLUSIONS

In this paper, we presented a novel approach of event-driven query expansion. Our approach identifies events related to the query and then suggests expansion terms that are semantically related to those events or influenced by them. We studied different embedding types for words and events. We identified that temporal embeddings coupled with a mechanism to simultaneously embed words and events in the same space significantly improve QE performance. The mechanism operates by projecting event embeddings from an auxiliary model (Wikipedia) to temporal models. This enables us to compare words and events at specific times. We analyzed different event-related queries and conclude that temporal embeddings significantly improve multiple-event queries. Additionally, we studied several methods for event detection, expansion candidate composition, and expansion candidate scoring. We showed that

Table 5: Comparison of event detection methods.

Method	Robust			TREC12			AP			WSJ			ALL		
	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP
Sim.	0.25	0.26	0.11	0.36	0.34	0.12	0.22	0.22	0.10	0.31	0.34	0.15	0.28	0.25	0.13
Freq.	**0.45***	**0.45***	**0.23***	**0.53***	**0.54***	**0.31***	**0.37**	**0.37**	**0.23***	**0.41**	**0.41**	**0.27***	**0.47***	**0.43***	**0.31***

Table 6: Comparison of expansion candidate scoring features in a leave-one-out setting.

Left-Out Feature	Robust			TREC12			AP			WSJ		
	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP	P@10	NDCG	MAP
$\cos(c, q)$	0.44	0.45	**0.24**	0.49	0.51	0.29	0.35	0.35	0.22	0.35	0.36	0.25
$\cos(c, e)$	**0.46**	**0.46**	0.23	**0.53**	**0.54**	**0.31**	**0.38**	**0.38**	**0.23**	**0.41**	0.41	**0.27**
$\cos(e, q)$	0.45	0.45	0.23	0.51	0.53	**0.31**	0.35	0.36	**0.23**	**0.41**	0.41	**0.27**
$\mathit{tfidf}(c, e)$	0.45	0.45	**0.23**	**0.53**	**0.54**	**0.31**	0.37	0.37	0.23	**0.41**	**0.42**	**0.27**
$\mathit{TempRel}(c, e)$	0.45	0.45	**0.23**	0.52	0.53	**0.31**	0.35	0.36	**0.23**	**0.41**	0.41	**0.27**
All features	0.45	0.45	0.23	0.53	**0.54**	**0.31**	0.37	0.37	**0.23**	**0.41**	0.41	**0.27**

Table 7: Positive examples of expansions created by TED, compared with a baseline [17].

Query	Top Detected Events	Top Expansions	Baseline's Top Expansions
African civilian deaths	War in Darfur, Somali Civil War, Sudanese Civil War	uganda, sudan, military, force, tutsi	sudanese, rhodesian, ugandan, rwandan, namibian
U.S. invasion of Panama	US invasion of Panama	invade, air, operation, occupation, force	nicaragua, reoccupation, guatemala, ryukyus, hispaniola
Tiananmen Square protesters	1989 Tiananmen Square protests	demonstration, crowd, riot, beijing, protests	protesters, demonstrators, protesting, marchers, protests

Table 8: Negative examples of expansions created by TED, compared with a baseline [17].

Query	Top Detected Events	Top Expansions	Baseline's Top Expansions
Bank Failures	2000 Camp David Summit	loan, deposit, israel, banks, lending	banks, collapse, depositors, failures, savings
Death from Cancer	Chernobyl Disaster	disease, illness, lymph, reactor, dose	leukemia, prostate, pancreatic, melanoma, tumour
Greenpeace	Sinking of the Rainbow Warrior	bomb, environmentalist, yacht, auckland, warrior	environmentalists, avaaz worldwatch, actionaid, adbusters

our proposed method improves QE performance significantly on various newswire TREC collections, compared with state-of-the-art baselines. For future work, we intend to leverage the event-driven methodology for other IR tasks, such as reranking and query classification.

REFERENCES

[1] Gianni Amati and Cornelis Joost Van Rijsbergen. 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 357–389.

[2] Hiteshwar Kumar Azad and Akshay Deepak. 2019. A new approach for query expansion using Wikipedia and WordNet. *Information Sciences* 492 (2019), 147–163.

[3] Hiteshwar Kumar Azad and Akshay Deepak. 2019. Query expansion techniques for information retrieval: A survey. *Information Processing & Management* 56, 5 (2019), 1698–1735.

[4] Judit Bar-Ilan, Zheng Zhu, and Mark Levene. 2009. Topic-Specific Analysis of Search Queries. In *Proceedings of the 2009 Workshop on Web Search Click Data* (Barcelona, Spain) *(WSCD '09)*. Association for Computing Machinery, New York, NY, USA, 35–42. https://doi.org/10.1145/1507509.1507515

[5] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. 1995. A limited memory algorithm for bound constrained optimization. *SIAM Journal on scientific computing* 16, 5 (1995), 1190–1208.

[6] Ricardo Campos, Gaël Dias, Alípio M Jorge, and Adam Jatowt. 2014. Survey of temporal information retrieval and related applications. *ACM Computing Surveys (CSUR)* 47, 2 (2014), 1–41.

[7] Claudio Carpineto and Giovanni Romano. 2012. A Survey of Automatic Query Expansion in Information Retrieval. *ACM Comput. Surv.* 44, 1 (Jan. 2012).

[8] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word Translation Without Parallel Data. *arXiv preprint arXiv:1710.04087* (2017).

[9] Supratim Das, Arunav Mishra, Klaus Berberich, and Vinay Setty. 2017. Estimating event focus time using neural word embeddings. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management.* 2039–2042.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[11] Valerio Di Carlo, Federico Bianchi, and Matteo Palmonari. 2019. Training temporal word embeddings with a compass. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6326–6334.

[12] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. 2016. Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891* (2016).

[13] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2016. Knowledge-driven event embedding for stock prediction. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers.* 2133–2142.

[14] Evgeniy Gabrilovich, Shaul Markovitch, et al. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis.. In *IJcAI*, Vol. 7. 1606–1611.

[15] Seyyedeh Newsha Ghoreishi and Aixin Sun. 2013. Predicting event-relatedness of popular queries. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management.* 1193–1196.

[16] William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1489–1501.

[17] Ayyoob Imani, Amir Vakili, Ali Montazer, and Azadeh Shakery. 2019. Deep neural networks for query expansion using word embeddings. In *European Conference on Information Retrieval.* Springer, 203–210.

[18] Nattiya Kanhabua and Avishek Anand. 2016. Temporal information retrieval. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval.* 1235–1238.

[19] Nattiya Kanhabua, Tu Ngoc Nguyen, and Wolfgang Nejdl. 2015. Learning to detect event-related queries for web search. In *Proceedings of the 24th International Conference on World Wide Web.* 1339–1344.

[20] Nattiya Kanhabua and Kjetil Nørvåg. 2010. Determining time of queries for re-ranking search results. In *International Conference on Theory and Practice of Digital Libraries.* Springer, 261–272.

[21] Saar Kuzi, Anna Shtok, and Oren Kurland. 2016. Query expansion using word embeddings. In *Proceedings of the 25th ACM international on conference on information and knowledge management.* 1929–1932.

[22] Ye Ma, Lu Zong, Yikang Yang, and Jionglong Su. 2019. News2vec: News Network Embedding with Subnode Information. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).* 4845–4854.

[23] Craig Macdonald, Richard McCreadie, Rodrygo LT Santos, and Iadh Ounis. 2012. From puppy to maturity: Experiences in developing Terrier. *Proc. of OSIR at SIGIR* (2012), 60–63.

[24] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval.* Cambridge university press.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems.* 3111–3119.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems.* 3111–3119.

[27] Jamal Abdul Nasir, Iraklis Varlamis, and Samreen Ishfaq. 2019. A knowledge-based semantic framework for query expansion. *Information Processing & Management* 56, 5 (2019), 1605–1617.

[28] Sérgio Nunes, Cristina Ribeiro, and Gabriel David. 2008. Use of temporal expressions in web search. In *European Conference on Information Retrieval.* Springer, 580–584.

[29] Ramith Padaki, Zhuyun Dai, and Jamie Callan. 2020. Rethinking Query Expansion for BERT Reranking. In *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II 42.* Springer, 297–304.

[30] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *Proceedings of the 1st international conference on Scalable information systems.* 1–es.

[31] Kira Radinsky, Fernando Diaz, Susan Dumais, Milad Shokouhi, Anlei Dong, and Yi Chang. 2013. Temporal web dynamics and its application to information retrieval. In *Proceedings of the sixth ACM international conference on Web search and data mining.* 781–782.

[32] Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks.* Citeseer.

[33] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp* 109 (1995), 109.

[34] Guy D Rosin, Eytan Adar, and Kira Radinsky. 2017. Learning Word Relatedness over Time. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.* 1168–1178.

[35] Guy D Rosin and Kira Radinsky. 2019. Generating Timelines by Modeling Semantic Change. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL).* 186–195.

[36] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. 2016. Using word embeddings for automatic query expansion. *arXiv preprint arXiv:1606.07608* (2016).

[37] Vinay Setty and Katja Hose. 2018. Event2Vec: Neural embeddings for news events. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 1013–1016.

[38] Milad Shokouhi and Kira Radinsky. 2012. Time-sensitive query auto-completion. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval.* 601–610.

[39] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. *arXiv preprint 1812.06280v3* (2020).

[40] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning.* Association for Computational Linguistics, 250–259. https://doi.org/10.18653/v1/K16-1025

[41] Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the eleventh acm international conference on web search and data mining.* 673–681.

[42] Hamed Zamani and W Bruce Croft. 2016. Embedding-based query language models. In *Proceedings of the 2016 ACM international conference on the theory of information retrieval.* 147–156.

[43] Hamed Zamani and W Bruce Croft. 2016. Estimating embedding vectors for queries. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval.* 123–132.

[44] Hamed Zamani and W Bruce Croft. 2017. Relevance-based word embedding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 505–514.

[45] Chengxiang Zhai and John Lafferty. 2017. A study of smoothing methods for language models applied to ad hoc information retrieval. In *ACM SIGIR Forum*, Vol. 51. ACM New York, NY, USA, 268–276.

[46] Xiaojuan Zhang, Shuguang Han, and Wei Lu. 2018. Automatic prediction of news intent for search queries. *The Electronic Library* (2018).

A COMPARISON TO PSEUDO-RELEVANCE FEEDBACK

Adding pseudo-relevance feedback (PRF) to an existing word embedding based QE method has been found to bring significant improvements to retrieval performance [21, 42], and is thus consistently used by state-of-the-art methods [17, 29]. In this work, we presented a general model that is independent of PRF. We leave its integration with PRF for future work (i.e., PRF can be considered as an additional component on top of our model).

Nevertheless, to extend our analysis, in this appendix we compare our method TED (which does not involve PRF) with a PRF method called Bo1, which is the default query expansion method in Terrier. Bo1 is a generalization of Rocchio's method and is based on Divergence from Randomness [1]. Note that PRF requires an additional retrieval round and has access to more information about the query. TED achieves better or equal performance compared to Bo1 on all the TREC datasets used in this work, combined (though the differences are not statistically significant). On the P@10, NDCG, and MAP metrics, TED achieves 0.47, 0.48, and 0.31, compared to 0.42, 0.42, and 0.31 by Bo1, respectively.