Generating Semantic Aspects for Queries

Dhruv Gupta[1,2(✉)], Klaus Berberich[1,3], Jannik Strötgen[4], and Demetrios Zeinalipour-Yazti[5]

[1] Max Planck Institute for Informatics, Saarbrücken, Germany
{dhgupta,kberberi}@mpi-inf.mpg.de
[2] Graduate School of Computer Science, Saarbrücken, Germany
[3] htw saar, Saarbrücken, Germany
[4] Bosch Center for Artificial Intelligence, Renningen, Germany
jannik.stroetgen@de.bosch.com
[5] University of Cyprus, Nicosia, Cyprus
dzeina@cs.ucy.ac.cy

Abstract. Large document collections can be hard to explore if the user presents her information need in a limited set of keywords. Ambiguous intents arising out of these short queries often result in long-winded query sessions and many query reformulations. To alleviate this problem, in this work, we propose the novel concept of semantic aspects (e.g., $\langle\{\text{michael-phelps}\}, \{\text{athens, beijing, london}\}, [2004, 2016]\rangle$ for the ambiguous query *olympic medalists*) and present the xFactor algorithm that generates them from annotations in documents. Semantic aspects uplift document contents into a meaningful structured representation, thereby allowing the user to sift through many documents without the need to read their contents. The semantic aspects are created by the analysis of semantic annotations in the form of temporal, geographic, and named entity annotations. We evaluate our approach on a novel testbed of over 5,000 aspects on Web-scale document collections amounting to more than 450 million documents. Our results show the xFactor algorithm finds relevant aspects for highly ambiguous queries.

1 Introduction

When querying large document collections or the Web, it is challenging to guide the user to relevant documents. This is because short and ambiguous keyword queries posed to information retrieval (IR) systems represent many possible information needs [15]. This is a known acute problem; it has been shown that around 46% users issue reformulated queries [22]. To assist users in refining their search, existing approaches use related terms [31], named entities in knowledge graphs (KGs) [13] or hand-crafted knowledge-panels [24]. Still with these aids, the user must read and consult individual documents in the ranked list to check for their relevance. What is therefore needed is a way of uplifting the unstructured text in documents to a structured representation that exposes its key *aspects*. To this end, we propose the novel concept of *semantic*

P. Hitzler et al. (Eds.): ESWC 2019, LNCS 11503, pp. 162–178, 2019.
https://doi.org/10.1007/978-3-030-21348-0_11

aspects (e.g., $\langle\{\text{michael-phelps}\}, \{\text{athens, beijing, london}\}, [2004, 2016]\rangle$) that help users posing ambiguous queries (e.g., *olympic medalists*) explore document collections without reading their contents.

Table 1. Generated semantic aspects for the query *olympic medalists* from the New York Times document collection (covering 1987–2007). Each row in the table corresponds to one semantic aspect.

Query: *olympic medalists*		
Time	**Entities**	**Entity Type**
[1980,1988]	*yago:sergei-grinkov, yago:maya-usova, yago:marina-klimova, yago:evgeni-platov, yago:oksana-grishuk, yago:sergei-ponomarenko, yago:ekaterina-gordeeva*	*wiki-category: olympic medalists in figure skating*
[1992,1992]	*yago:leroy-burrell, yago:jon-drummond, yago:dennis-mitchell, yago:michael-marsh-(athlete)*	*wiki-category: american sprinters*
[1996,1996]	*yago:dominique-dawes, yago:shannon-miller, yago:kerri-strug*	*wiki-category: olympic medalists in gymnastics*
[1998,1998]	*yago:jenni-meno, yago:kyoko-ina, yago:todd-eldredge, yago:todd-sand, yago:nicole-bobek*	*wiki-category: figure skaters at the 1998 winter olympics*

To generate semantic aspects for ambiguous keyword queries, we turn to natural language processing (NLP) tools that can help us enrich text with annotations. In particular, we make use of annotations in the form of named entities (persons and organizations), geographic locations, and temporal expressions. These are extremely important annotations in the domain of IR [20,36]: 71% of Web queries were found to mention named entities, while 30.9% of Web queries were either explicitly or implicitly temporal in nature. Generation of meaningful semantic aspects and their evaluation is challenging. To generate them, we must first model and interpret the semantics underlying the annotations. For example, temporal expressions can be highly uncertain (e.g., *90s*) and two locations or named entities in a KG can be related by many facts, e.g., 'Maria Sharapova lives in US but represents Russia in sports' [4]. Moreover, queries can signify different kinds of ambiguities: temporal ambiguity (e.g., *tokyo summer olympics* - 1964 or 2020), location ambiguity (e.g., *rome* - many US cities are named after European cities), or entity ambiguity (e.g., *spitz* - Mark or Elisa Spitz). Moreover, since semantic aspects are more than "related terms" or facts in KGs there currently exists no benchmark for their automatic evaluation.

Approach Outline. To solve the above challenges, we propose the following solutions in this work. To generate semantic aspects, we describe the xFactor algorithm (Sect. 4). xFactor takes as an input a large set of annotated documents retrieved for a keyword query and outputs a set of semantic aspects. xFactor generates the semantic aspects in three steps. First, it partitions the input document set by identifying salient sets of annotations in formal models that capture the semantics of an annotation type (e.g., named entities). Second, xFactor additionally considers salient co-occurrences of annotations with

different semantics (e.g., named entities and temporal expressions) by virtue of them being present in the same document partition. Third, it outputs all possible ways of analyzing the initial ambiguity behind the query. This is done by permuting the order in which the annotations are considered for partitioning the initial set of documents. Table 1 shows examples of generated semantic aspects for the ambiguous query *olympic medalists*. To perform automated evaluation of the generated semantic aspects, we provide a novel evaluation benchmark compiled from Wikipedia with new measures to the research community (Sect. 6).

2 Preliminaries

We first cover the required terminology to describe our method.

Document Model. Consider, a large annotated document collection $\mathcal{D} = \{d_1, d_2, \ldots, d_{|\mathcal{D}|}\}$. Each document $d \in \mathcal{D}$ is processed with natural language processing (NLP) annotators that tag sequences of words in the document with annotations from a given type $\mathcal{X}$ (e.g., temporal expressions). Formally, let each document be represented by n bags-of-annotations:

$$d = \{d_{\mathcal{X}_1}, d_{\mathcal{X}_2}, \ldots, d_{\mathcal{X}_n}\}. \tag{1}$$

Specifically, for this work, we consider the following semantic annotations: temporal expressions, geographic locations, and other named entities (persons and organizations). Thus, our document model refers to the bags-of-annotations for entities ($d_{\mathcal{E}}$), locations ($d_{\mathcal{G}}$), and temporal expressions ($d_{\mathcal{T}}$): $d = \{d_{\mathcal{E}}, d_{\mathcal{G}}, d_{\mathcal{T}}\}$.

Aspect Model. Let an ambiguous query q reflect the *information need* of a user. An information retrieval (IR) method in response to the query q returns a set of pseudo-relevant documents $\mathcal{R} \subset \mathcal{D}$. The document set $\mathcal{R}$ conveys many implicit information needs. The user's information need may be a subset of those reflected by the documents [15]. Our aim is to extract an ordered set of semantic aspects $\mathcal{A}$ that make these implicit information needs explicit, thereby pointing the user to the relevant documents:

$$\mathcal{A} = \langle a_1, a_2, \ldots, a_{|\mathcal{A}|} \rangle. \tag{2}$$

An aspect a (e.g., $\langle \{\text{michael-phelps}\}, \{\text{athens, beijing, london}\}, [2004, 2016] \rangle$) is determined by considering the salience of annotations sharing the same semantics (e.g., temporal expressions) as well as co-occurrence with annotations of different semantics (e.g., temporal expressions and named entities). An aspect $a \in \mathcal{A}$ is modeled as: $a = \langle x_1, \ldots, x_n \rangle$, where x (e.g., time interval) corresponds to salient annotation(s) from type $\mathcal{X}$ (e.g., temporal expressions). For this work, the aspects are modeled as: $a = \langle a_{\mathcal{E}}, a_{\mathcal{G}}, a_{\mathcal{T}} \rangle$.

Annotation Models. For large-scale enrichment of text documents with semantic annotations we utilize two different NLP tools. First, we make use of a named entity recognition and disambiguation (NERD) tool Aida [26] to

annotate and disambiguate mentions of named entities to canonical entries in KGs (i.e., Yago [34]). Second, to resolve expressions of time in text we leverage a temporal tagger HeidelTime [33] that can annotate them with high precision. We next explain the formal models for each of the annotation types.

Named Entities and Entity Model. Named entities in text are modeled as canonical entries of a KG (e.g., Yago [34]). These annotations are obtained by using NERD tools (e.g., Aida [26]). We differentiate between locations and other named entities, by the presence of geographic coordinates in their KG entry. Let, $\mathcal{G}$ and $\mathcal{E}$ denote the type information associated with locations and other entities, respectively. Named entities may share common relationships that convey a degree of their relatedness. For example, *tokyo* and *beijing* are related as they both lie in *asia*. These relationships in Wikipedia are encoded in the form of an explicit link structure. Concretely, each named entity mention disambiguated by Aida, is linked to its Wikipedia article. Each Wikipedia article contains links to other Wikipedia articles, indicating their semantic relatedness. We model each named entity by its Wikipedia link structure. Formally, each named entity e can be described by the links ℓ its article W_e has to other articles in Wikipedia W:

$$W_e = \{\ell_1, \ell_2, \ldots, \ell_{|W|}\}. \tag{3}$$

Temporal Expressions and Time Model. Temporal expressions in documents can be annotated using temporal taggers (e.g., HeidelTime [33]). Such annotators are able to identify and resolve explicit, implicit, relative, and underspecified temporal expressions using metadata such as publication dates [33]. Let, $\mathcal{T}$ denote the type information associated with temporal expressions. Each annotation in $\mathcal{T}$ (i.e., dates) is represented by their UNIX time epochs (i.e., number of milliseconds since 01-January-1970). Temporal expressions are challenging to analyze as they can be present at different levels of granularity, e.g., day, month, or year granularity. Furthermore, temporal expressions can indicate an uncertain time interval, e.g., *1990s*. In such cases, the begin and end of the time interval conveyed is not clear. An uncertain temporal expression can therefore refer to infinitely many time intervals. The uncertainty in temporal expressions can be modeled by analyzing when the time interval could have begun and ended [8]. That is, the temporal expression *1990s* can refer to any time interval that can begin (b) in $[1990, 1999]$ and end (e) in $[1990, 1999]$. In other words, $b \in [b_\ell, b_u]$ and $e \in [e_\ell, e_u]$ (with $b \leq e$) giving the uncertainty-aware time model [8]:

$$T = \langle b_\ell, b_u, e_\ell, e_u \rangle. \tag{4}$$

For example, *1990s* can now be represented as: $\langle 1990, 1999, 1990, 1999 \rangle$.

3 Generating Factors

To generate semantic aspects for a given query, we first need to compute salience of annotations in models informed of their semantics. Thus, we are not simply

counting annotations but rather considering the salience of entities, locations, and temporal expressions in their respective semantic models. We denote the methods that compute salience as **factor methods** and the resulting salient annotations as **factors** (e.g., sets of locations or time intervals). We next describe how to find factors associated with each annotation type $\mathcal{X}$ in a document set $\mathcal{R}$ by using its factoring method, factor($\mathcal{X}, \mathcal{R}$). Algorithm 1 outlines how factor methods use the salience computation for a given document partition and annotation type.

Factoring Named Entities - factor($\mathcal{X}_\mathcal{G}, \mathcal{R}$) and factor($\mathcal{X}_\mathcal{E}, \mathcal{R}$). The factor method for named entities outputs sets of entities and locations where each entity in the set is related to the others in highly relevant documents. Concretely, to create the factors (i.e., sets of locations and entities) we first compute: sim(e, e'), semantic relatedness between entities e and e'. This is done by calculating the Jaccard coefficient of links shared by the Wikipedia entries of e and e':

$$\text{sim}(e, e') = \frac{|W_e \cap W_{e'}|}{|W_e \cup W_{e'}|}, \tag{5}$$

We make use of Jaccard coefficient as entity relatedness measure, as it has shown good performance over other relatedness measures [14]. Second, we weight the entity-entity relatedness by the document relevance $s(d, \mathcal{R})$ (given by the IR method during retrieval) that contains them to compute entity salience s($e, \mathcal{R}$). That is,

$$\text{s}(e, \mathcal{R}) = \sum_{d \in \mathcal{R}} \text{s}(d, \mathcal{R}) \cdot \sum_{e' \in d_\mathcal{E}} \text{sim}(e, e'). \tag{6}$$

Factoring Time - factor($\mathcal{X}_\mathcal{T}, \mathcal{R}$). Temporal expressions are challenging to analyze. For instance, uncertain temporal expressions such as *the 90s* can refer to an infinite number of time intervals. Thus, it becomes quite difficult to identify salient time intervals. To overcome these limitations, we use the approach by Gupta and Berberich [21] to generate factors for time. In brief, salient time intervals (time factors) in $\mathcal{R}$ (i.e., s($[b, e], \mathcal{R}$)) can be found by generating overlaps of the temporal expressions in the uncertainty-aware time model and weighting them by the document's relevance, which contains the temporal expressions:

$$\text{s}([b, e], \mathcal{R}) = \sum_{d \in \mathcal{R}} \text{s}(d, \mathcal{R}) \cdot \text{sim}([b, e], d_\mathcal{T}), \tag{7}$$

where, s($d, \mathcal{R}$) denotes the document relevance and sim($[b, e], d_\mathcal{T}$) denotes the salience of the time interval $[b, e]$ in the document's bag of temporal expressions $d_\mathcal{T}$. The value of sim($[b, e], d_\mathcal{T}$) is computed as follows:

$$\text{sim}([b, e], d_\mathcal{T}) = \frac{1}{|d_\mathcal{T}|} \cdot \sum_{T \in d_\mathcal{T}} \frac{\mathbb{1}([b, e] \in T)}{|T|}. \tag{8}$$

In Eq. 8, the cardinality $|T|$ denotes the number of time intervals T can generate and the indicator function $\mathbb{1}(\bullet)$ tests the membership of $[b, e]$ in T.

4 The xFactor Algorithm

In addition to annotation salience, we consider the co-occurrence salience of annotations from different types to generate semantic aspects. To this end, we propose the xFactor algorithm. Our xFactor algorithm is inspired by the Apriori algorithm for frequent itemset mining [6]. The Apriori algorithm, however, is not informed of annotation semantics. Thus, its direct application, will not capture any semantic co-occurrence among different annotation types.

Consider, a document set $\mathcal{R}$ and its n annotation types $\{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n\}$. A set of salient aspects $\mathcal{A}$ can be derived by iteratively partitioning $\mathcal{R}$ for different annotation factors:

$$\text{Basis Step} : \{x_1\} = \text{factor}(\mathcal{X}_1, \mathcal{R}) \tag{9}$$

$$\text{Inductive Step} : \{x_k\} = \text{factor}(\mathcal{X}_k, \mathcal{R}^{(k-1)\ldots(1)}). \tag{10}$$

First and foremost, the salience of a factor (e.g., time interval) in each aspect is obtained by the factor method (Sect. 3) that considers a semantic model corresponding to its annotation type (e.g., temporal expressions). Second, each factor for an annotation type allows us to partition the document set $\mathcal{R}$ into documents that contain annotations that helped derive the factor and those documents which did not help. Thus, by iteratively applying the factor methods for the different annotation types, we can identify the salience between factors by virtue of their co-occurrence in the same partition. Mathematically given by:

$$\text{factor}(\mathcal{X}_1, \mathcal{R}) : \{d_{\mathcal{X}_1} \in d \mid \forall d \in \mathcal{R}\} \rightarrow 2^{\mathcal{X}_1}, \tag{11}$$

$$\text{factor}(\mathcal{X}_k, \mathcal{R}^{(k-1)\ldots(1)}) : \{d_{\mathcal{X}_k} \in d \mid \forall d \in \mathcal{R}^{(k-1)\ldots(1)}\} \rightarrow 2^{\mathcal{X}_k}. \tag{12}$$

Third and finally, we create a partition index that keeps track of factors that were generated by a particular partition of the pseudo-relevant set of documents $\mathcal{R}$. The partition index for iteration i of the recursive algorithm keeps track of:

$$\text{Partition Index: } \langle \{x_k\}_i, R_i^{(k-1)\ldots(1)} \rangle. \tag{13}$$

By concatenating the factors of different annotation types, from the same document partition, we can generate the aspects. Figure 1 exemplifies the recursive xFactor algorithm and how the aspects are generated. The xFactor algorithm thereby allows us to extract a subset of aspects that contain salient relationships among their factors from all possible combinations of different annotation types: $\mathcal{A} \subset 2^{\mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_n}$. Algorithm 2 illustrates a tail-recursive version of the xFactor algorithm.

Minimum Salience and Aspect Ranking. The xFactor algorithm is still computationally expensive if we were to consider every factor for each annotation type. To prune the recursion depth, we utilize a minimum salience criteria. For a given value of minimum salience $\sigma \in [0, 1]$, a factor is deemed salient if and only if: $\text{s}(x, \mathcal{R}) \geq \sigma$. Using the salience we furthermore rank the aspects presented to the user as: $\text{s}(a, d) = \prod_{x_i \in a} \text{s}(x_i, d)$.

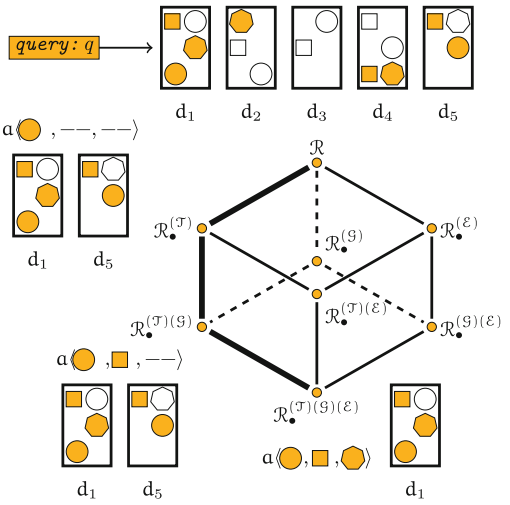

Fig. 1. The lattice structure for aspect generation by the xFactor algorithm is shown. Shapes in documents d represent annotation types. While colors represent different annotation values for same annotation type. Each element in the lattice corresponds to the partition of documents that arises by applying the factor method for that annotation type. For example, $\mathcal{R}_{\bullet}^{(\mathcal{T})}$ is generated by factoring $\mathcal{R}$ along time. The time factor $a = \langle \bullet, -, - \rangle$ is generated by documents $\{d_1, d_5\} \in \mathcal{R}_{\bullet}^{(\mathcal{T})}$. Continuing in this recursive manner over the geographic annotation type $\mathcal{G}$ we get $a = \langle \bullet, \blacksquare, - \rangle$. The sequence of factoring operations can be permuted to obtain different partitions; $\mathcal{R}_{\bullet}^{(\mathcal{T})(\mathcal{G})(\mathcal{E})}$ corresponds to time → geography → entity (traversing the bold edges).

Algorithm 1: Generate Factors.

Function `factor`$(\mathcal{X}, \mathcal{R}, \sigma)$
  $\bigcup\langle x, \mathcal{R}'\rangle \leftarrow$ `generate pairs of: factors using the semantic model for` $\mathcal{X}$ `and the originating document partition.`
  factors $\leftarrow \varnothing$
  foreach $(\langle x, \mathcal{R}'\rangle \in \bigcup\langle x, \mathcal{R}'\rangle)$ **do**
    if $(\mathtt{s}(x, \mathcal{R}') \geq \sigma)$ **then**
      factors.`add`(x)
      `PartitionIndex.put`$(\langle x, \mathcal{R}'\rangle)$
  return factors

Algorithm 2: The xFactor Algorithm.

Function `xFactor`$(\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n, \mathcal{R}, \sigma)$
  `// The set of aspects to return`
  $\mathcal{A} \leftarrow \varnothing$
  x_nFactors $\leftarrow$ `factor` $(\mathcal{X}_n, \mathcal{R}, \sigma)$
  foreach (x_nFactor $\in x_n$Factors) **do**
    $\mathcal{R}' \leftarrow$ `PartitionIndex.get`$(x_n$Factor$)$
    x_{n-1}Factors $\leftarrow$ `factor`$(\mathcal{X}_{n-1}, \mathcal{R}', \sigma)$
    ...
    foreach (x_2Factor $\in x_2$Factors) **do**
      $\mathcal{R}' \leftarrow$ `PartitionIndex.get`$(x_2$Factor$)$
      x_1Factors $\leftarrow$ `factor`$(\mathcal{X}_1, \mathcal{R}', \sigma)$
      `// Generate aspects`
      foreach (x_1Factor $\in x_1$Factors) **do**
        $\mathcal{A} \leftarrow$
        $\mathcal{A} \cup \langle x_1$Factor, ..., x_nFactor$\rangle$
    ...
  return $\mathcal{A}$

5 Properties of the xFactor Algorithm

Structured Representation of Documents. The aspects which are assimilated from multiple documents can be used to transform the semi-structured documents with annotations (i.e., $d = \{d_{\mathcal{E}}, d_{\mathcal{G}}, d_{\mathcal{T}}\}$) into a structured representation of aspects (i.e., $d = \langle a_1, a_2, \ldots, a_n \rangle$). This structured representation of documents using aspects is then immediately useful for applications in search tasks, such as result diversification. To represent documents using aspects, we can obtain the inverse mapping of documents to aspects by looking for all $a \in \mathcal{A}$ associated with a particular document d in the partition index.

Query Pivoting. If the order in which the annotation types are factored are permuted then the xFactor algorithm will produce different sets of aspects. This is because the factor methods rely on a given document partition to generate the factors. For instance, for three annotation types, we can realize six different sets of aspects by permutation of the different factor methods. This in turn provides us different ways of analyzing three different kinds of initial ambiguity underlying the query: temporal ambiguity, e.g., *tokyo summer olympics*; geographical ambiguity, e.g., *springfield*; and named entity related ambiguity, e.g., *george bush*. If the sequence of factor methods is time $\rightarrow$ entity $\rightarrow$ geography, then the resulting set of aspects will be denoted by $\mathcal{A}_{\langle \mathcal{T},\mathcal{E},\mathcal{G} \rangle}$. The other five possibilities are: $\mathcal{A}_{\langle \mathcal{T},\mathcal{G},\mathcal{E} \rangle}$, $\mathcal{A}_{\langle \mathcal{G},\mathcal{T},\mathcal{E} \rangle}$, $\mathcal{A}_{\langle \mathcal{G},\mathcal{E},\mathcal{T} \rangle}$, $\mathcal{A}_{\langle \mathcal{E},\mathcal{T},\mathcal{G} \rangle}$, and $\mathcal{A}_{\langle \mathcal{E},\mathcal{G},\mathcal{T} \rangle}$. Using the illustration in Fig. 1, these six factor sequences can be obtained by following different paths in the lattice.

Summarizing Entity Sets. For each of the generated aspects, we can summarize the resulting factors (e.g., named entities) using background knowledge (e.g., KG) into broader semantic classification types (e.g., categories from Wikipedia). For instance, in Table 1, we have summarized all the named entities into categories from Wikipedia. Concretely, to arrive at the types for named entities, we look up all the types that an entity belongs to (following rdfs:type and rdfs:subClassOf links). Thereafter, we select the summary type as the one that covers most of the entities in the entity factor.

6 Evaluation

We next describe the setup and results of our experimental evaluation.

6.1 Annotated Document Collections

Document Collections. We test our algorithm on two different types of document collections. The first category of document collections consists of news articles. News archives have the benefit of being accompanied by rich metadata in the form of accurate publication dates and well-written text. This can aid NLP tools to provide more accurate annotations. For example, temporal taggers can resolve relative temporal expressions (e.g., yesterday) and implicit temporal

expressions (e.g., good friday) with respect to the publication date. We consider two document collections in this category. One of them is a collection of approximately two million news articles published in the New York Times between 1987 and 2007. It is publicly available as the New York Times Annotated Corpus [5]. The other one is a collection of approximately four million news articles collected from various online sources during the period of 2013 to 2016, called Stics [25].

The second category of document collection consists of web pages. Web crawls unlike news articles have unreliable metadata and ill-formed language. This hampers us in obtaining high-quality semantic annotations for them. For example, we cannot resolve relative and underspecified temporal expressions, as the document creation time for Web pages may not reflect their true publication dates. We consider two web crawls [1,2] from 2009 and 2012, which are publicly available as ClueWeb'09 and ClueWeb'12 document collections, respectively. Statistics for the document collections are summarized in Table 2.

Annotating Documents. Semantic annotations are central to our approach. To obtain them, we utilize publicly available annotations for the document collections or automatically generate them using NLP tools. For the news archives and for ClueWeb'09, we utilized Aida [26], which performs named entity recognition and disambiguation. Each disambiguated named entity is linked to its canonical entry in the Yago KG and Wikipedia. As a subset of these named entities, we can obtain geographic locations. For ClueWeb'12, we utilized the FACC annotations [17] provided by Google. The FACC annotations contain the offsets of high precision entities spotted in the web pages. Temporal expressions for all the document collections were obtained using the HeidelTime temporal tagger [33]. In Table 2, we additionally report the average counts of the three types of semantic annotations found in at most 10,000 documents retrieved for each query in our testbed.

Table 2. Collection statistics.

Collection	#Documents	Avg. Time	Avg. Entities	Avg. Locations
New York Times	1,679,374	12.50	16.25	8.65
Stics	4,075,720	10.09	10.89	5.93
ClueWeb'09	50,220,423	30.59	8.23	9.49
ClueWeb'12	408,878,432	5.80	7.74	5.61

6.2 Ground Truth Semantic Aspects and Queries

To evaluate our system, we extracted 5,122 aspects from Wikipedia. This was done considering their diversity along annotation types of time, locations, and other named entities for a set of twenty-five keyword queries. The broad topics of the aspects along with the specific keyword queries and the number of aspects generated are listed in Table 4.

For each query, we constructed a set of ground-truth aspects by considering the table of events present on the Wikipedia page corresponding to the query [3,9]. For the table, we considered each row consisting of time, locations, and other entities as an aspect. If no locations or entities were mentioned, we extracted them from the associated event page of the row, by running Aida on the introductory paragraph of the event's Wikipedia page. For instance, consider Table 3 as an example Wikipedia table for Olympic medalists. Treating each row as a ground truth aspect, we look for temporal expressions, e.g., [2008, 2016] as a time factor; locations, e.g., Beijing, London, and Rio de Janeiro as a location factor; and other named entities, e.g., Usain Bolt as an entity factor. Similarly, for the second row in Table 3, the extracted aspect is: $\langle[2004, 2016], \{\text{athens}, \text{beijing}, \text{london}, \text{rio-de-janeiro}\}, \{\text{michael-phelps}\}\rangle$. The testbed is publicly available at the following URL: http://resources.mpi-inf.mpg.de/dhgupta/data/eswc2019/.

Table 3. An example table of events for generating ground truth.

Years	Description	Locations
2008 to 2016	Usain Bolt won total of 9 Olympic medals during the Summer Olympic games in the years he was active.	Beijing, London, and Rio de Janeiro
2004 to 2016	Michael Phelps has won a record number of 23 gold medals at various Olympic games during his career.	Athens, Beijing, London, and Rio de Janeiro

6.3 Measures

The two key characteristics for evaluating aspects are: their correctness with respect to a ground truth and their novelty with respect to other aspects in the set. These two characteristics taken together ensure that our aspect sets are meaningful and non-redundant. We next describe the measures of correctness and novelty.

Similarity computation between aspects is central to both the correctness and novelty. To compute the similarity between the two aspects, a (system generated) and b (ground truth), we consider their similarity dimension-wise:

$$\text{sim}(a, b) = \frac{1}{3}\left(\frac{|a_{[b,e]} \cap b_{[b,e]}|}{|a_{[b,e]}|} + \frac{|a_{\mathcal{E}} \cap b_{\mathcal{E}}|}{|a_{\mathcal{E}}|} + \frac{|a_{\mathcal{G}} \cap b_{\mathcal{G}}|}{|a_{\mathcal{G}}|}\right),$$

where, for temporal similarity we coarsen the time intervals at year granularity to make them comparable. The temporal overlaps are computed using the uncertainty-aware time model [8] by converting the time intervals to the four-tuple notation. While for the other two dimensions the similarity is akin to computing the overlap between bag-of-locations and bag-of-entities. Note that the similarity computation is done with respect to the system generated aspect

Table 4. Query categories with factor operation sequences and aspect counts (in brackets).

Entity - $\mathcal{A}_{\langle \mathcal{E}, \bullet, \bullet \rangle}$: *nobel prize* [114] |*oscars* [1, 167] |*space shuttle missions* [155] |*olympic medalists* [48] |*paralympic medalists* [24]
Location - $\mathcal{A}_{\langle \mathcal{G}, \bullet, \bullet \rangle}$: *aircraft accidents* [513] | *avalanches* [56] | *epidemics* [211] | *famines* [133] | *genocides* [35] | *volcanic eruptions* [171] | *hailstorms* [39] | *landslides* [85] | *earthquakes* [39] | *nuclear accidents* [26] | *oil spills* [140] | *tsunamis* [88]
Time - $\mathcal{A}_{\langle \mathcal{T}, \bullet, \bullet \rangle}$: *assassinations* [130] | *cold war* [81] | *corporate scandals* [44] | *proxy wars* [34] | *united states presidential elections* [57] | *terror attacks* [316] | *treaties* [1, 057] | *wars* [359]

(a in the denominator). This is done to avoid matching (and thereby not rewarding) those system aspects a with a very large time interval, bag-of-locations or bag-of-entities, to every ground truth aspect (b).

Correctness. Given a set of aspects $\mathcal{A}$ generated by our algorithm for a query q and the set of aspects $\mathcal{B}$ corresponding to the ground truth derived from Wikipedia page for the same query, correctness is given by:

$$\text{correctness}(\mathcal{A}, \mathcal{B}) = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \text{sim}(a, b).$$

Novelty for the set of aspects $\mathcal{A}$ can be intuitively thought of measuring the dissimilarity with respect to $\mathcal{A}$ itself:

$$\text{novelty}(\mathcal{A}) = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \frac{1}{|\mathcal{A}|} \sum_{(a' \in \mathcal{A} / \{a\})} \Big(1 - \text{sim}(a, a')\Big).$$

We can additionally conform the correctness measure to the standard information retrieval measures such as **precision** and **recall** as follows:

$$\text{precision} = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \max_{b \in \mathcal{B}} \Big(\text{sim}(a, b)\Big) \quad \& \quad \text{recall} = \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \max_{a \in \mathcal{A}} \Big(\text{sim}(a, b)\Big).$$

6.4 Evaluation Setup

Baselines and Systems. We consider two baselines to compare our proposed approach. As a naïve baseline, we treated each document in the pseudo-relevant set to represent an aspect. This is equivalent to presenting the user a ranked list of documents to satisfy her information need. The equivalent aspect for a document is constructed by considering the earliest and latest time point in the document as its time interval and bag-of-locations and bag-of-entities to represent the other two dimensions. As a second baseline, we consider latent Dirichlet allocation (LDA) [12]. With this baseline, we want to cluster together those documents that are semantically similar using only text. Using LDA, we discover k topics from the pseudo-relevant set of documents. From each topic's partition of documents,

Table 5. Results for news archives.

(a) Results for correctness (C) & novelty (N).

	New York Times			Stics		
	$\mu_{\|\mathcal{A}\|}$	C	N	$\mu_{\|\mathcal{A}\|}$	C	N
BM25	3,096	**.0237**	**.4269**	3,797	**.0204**	**.4227**
LDA-50	50	.0067	.2634	50	.0102	.2910
LDA-100	100	.0053	.2107	100	.0080	.2404
LDA-200	200	.0046	.1497	200	.0063	.1639
xFactor	2,261	.0161	.4201	503	.0190	.3862

(b) Results for precision (P) & recall (R).

	New York Times			Stics		
	$\mu_{\|\mathcal{A}\|}$	P	R	$\mu_{\|\mathcal{A}\|}$	P	R
BM25	3,096	.1577	**.2749**	3,797	.1414	**.2828**
LDA-50	50	.0471	.0160	50	.0634	.0311
LDA-100	100	.0432	.0102	100	.0483	.0207
LDA-200	200	.0400	.0070	200	.0456	.0129
xFactor	2,261	**.2804**	.1777	503	**.2400**	.1427

we derive the corresponding semantic aspect by considering the earliest and latest time point in the partition as its time interval and bag-of-entities and bag-of-locations to represent the two remaining dimensions. We refer to this baseline as LDA-k. For the xFactor algorithm, we considered the specific sequence of factor operations that were deemed meaningful for that query (as shown in Table 4). For instance, since the query ***earthquakes*** is oriented towards locations we considered the factor sequence operations $\mathcal{A}_{\langle\mathcal{G},\mathcal{E},\mathcal{T}\rangle}$ and $\mathcal{A}_{\langle\mathcal{G},\mathcal{T},\mathcal{E}\rangle}$.

Parameters. For each query in Table 4, we retrieve at most 10,000 documents with disjunctive operator using Okapi BM25 as the retrieval method. We used the standard parameters, $b = 0.75$ and $k_1 = 1.20$, for its configuration. For the LDA baseline, we followed Griffiths and Steyvers [19] for setting its parameters. Specifically, β was set to 0.1 and α was set to $^{50}/_{|\text{topics}|}$. We considered three topic set sizes for LDA namely, $|\text{topics}| \in \{50, 100, 200\}$ and the same number of top-k documents for each topic, e.g., for $|\text{topics}| = 50$, we picked top-50 documents for each topic as its generating partition. For our method, the minimum salience was set to $\sigma = 0.001$.

6.5 Results for Quality

Results for News Archives. We first consider the results of the systems in terms of correctness and novelty as reported in Table 5a. We additionally report the average number of aspects $\mu_{|\mathcal{A}|}$ for each system under comparison. Note that the Okapi BM25 baseline gives us an upper bound for the value of correctness that can be obtained against the ground-truth. As, ultimately we generate the LDA topics and aspect from this set of documents. For the New York Times collection, our method identifies the most correct aspects with respect to the ground truth as compared to the LDA baselines. Despite the observation that Okapi BM25 wins in terms of novelty by considering all pseudo-relevant documents, our method still achieves a high degree of novelty, thereby identifying the most non-redundant set of aspects and is able to partition the set of pseudo-relevant documents to the greatest degree. For the Stics news collection, our method outperforms the LDA baselines in terms of correctness and is close to the upper bound that can be achieved from the given set of pseudo-relevant documents. Okapi BM25 achieves a higher novelty value, however, the increase compared

Table 6. Results for web collections.

(a) Results for correctness (C) & novelty (N).

	ClueWeb'09			ClueWeb'12		
	$\mu_{\|\mathcal{A}\|}$	C	N	$\mu_{\|\mathcal{A}\|}$	C	N
BM25	9,580	**.0155**	.3958	9,742	**.0266**	**.4531**
LDA-50	50	.0123	.3275	50	.0177	.3478
LDA-100	100	.0092	.2880	100	.0126	.3025
LDA-200	200	.0064	.2441	200	.0097	.2554
xFactor	1,822	.0146	**.4239**	616	.0173	.4176

(b) Results for precision (P) & recall (R).

	ClueWeb'09			ClueWeb'12		
	$\mu_{\|\mathcal{A}\|}$	P	R	$\mu_{\|\mathcal{A}\|}$	P	R
BM25	9,580	.1151	**.3595**	9,742	.1494	**.4018**
LDA-50	50	.0734	.0999	50	.0930	.1049
LDA-100	100	.0560	.0808	100	.0718	.0770
LDA-200	200	.0433	.0502	200	.0585	.0500
xFactor	1,822	**.2218**	.1880	616	**.2433**	.1648

to our method is not significant. Observing both correctness and novelty our method excels in providing both relevant and non-redundant sets of aspects when compared to the LDA baselines which can only achieve high novelty.

Now, consider precision and recall for the systems, as reported in Table 5b. For the New York Times, while considering precision, our system consists of more relevant aspects compared to the baselines. With respect to recall, the Okapi BM25 baseline wins by considering the entire pseudo-relevant set of documents. Note that the LDA baselines and our algorithm xFactor can not achieve this value as they discard many annotations, favoring precision over recall. Therefore, considering both precision and recall together, our system presents a balanced performance: high precision and recall. While the baselines achieve high recall only. For the Stics collection, when considering precision and recall, our method again shows significant improvements over the baselines. Thus, by taking all the four measures, correctness, novelty, precision, and recall, our method allows us to distill interesting aspects which can guide the user to navigate through a large number of documents.

Results for Web Collections. Web archives give us more challenging documents to test the effectiveness of our approach. Particularly, since they are not well-formed, they have a lower average number of annotations per document, the annotations in them are prone to more errors, and the size of the web archives is magnitudes larger than news archives. Hence, they present a challenging real-world scenario to test our methods. We first consider the results for the web archives when measuring correctness and novelty that are reported in Table 6a. For ClueWeb'09, our method outperforms both baselines in terms of novelty. In particular, for correctness our method comes close to the upper bound established by Okapi BM25. For ClueWeb'12 our method performs at par with baselines in terms of novelty. When considering the measures in isolation, for correctness the LDA baseline wins over our method and Okapi BM25 baseline has higher novelty than our method. However, when considering both correctness and novelty together, xFactor is consistent in providing more correct and novel aspects as opposed to the LDA baselines.

Next, we consider the second set of experimental results for web collections when measuring precision and recall that are reported in Table 6b. For ClueWeb'09, our method in terms of precision and recall outperforms the LDA

Table 7. Results for precision and recall at $k = \{10, 20, 50\}$.

(a) Results on news archives.

	New York Times			Stics		
	$P@10$	$P@25$	$P@50$	$P@10$	$P@25$	$P@50$
BM25	.1874	.1863	.1909	.1380	.1394	.1461
LDA-50	.0453	.0464	.0471	.0592	.0590	.0634
LDA-100	.0426	.0431	.0430	.0487	.0481	.0476
LDA-200	.0396	.0397	.0400	.0440	.0434	.0440
xFactor	**.2283**	**.2450**	**.2366**	**.1690**	**.1869**	**.1740**
	$R@10$	$R@25$	$R@50$	$R@10$	$R@25$	$R@50$
BM25	**.2593**	**.2672**	**.2510**	**.2732**	**.2834**	**.2700**
LDA-50	.0172	.0148	.0133	.0310	.0294	.0270
LDA-100	.0130	.0104	.0092	.0216	.0192	.0174
LDA-200	.0101	.0075	.0062	.0148	.0126	.0113
xFactor	.1729	.1691	.1633	.1274	.1295	.1237

(b) Results on Web collections.

	ClueWeb'09			ClueWeb'12		
	$P@10$	$P@25$	$P@50$	$P@10$	$P@25$	$P@50$
BM25	.1111	.1183	.1217	.1436	.1451	.1521
LDA-50	.0792	.0770	.0734	.0982	.0961	.0930
LDA-100	.0714	.0634	.0601	.0876	.0828	.0756
LDA-200	.0591	.0526	.0485	.0697	.0644	.0602
xFactor	**.1771**	**.1909**	**.1919**	**.1999**	**.1990**	**.2018**
	$R@10$	$R@25$	$R@50$	$R@10$	$R@25$	$R@50$
BM25	**.3632**	**.3973**	**.3380**	**.3867**	**.4357**	**.3727**
LDA-50	.1000	.1036	.0894	.0963	.1153	.0940
LDA-100	.0864	.0856	.0750	.0722	.0832	.0681
LDA-200	.0524	.0571	.0450	.0446	.0534	.0421
xFactor	.1523	.1649	.1532	.1469	.1607	.1362

baselines significantly. For ClueWeb'12, our method outperforms both baselines with respect to precision. However, in terms of recall Okapi BM25 outperforms our method when considering all the pseudo-relevant documents. Despite of this, our method provides a balanced performance with high precision and moderate recall as compared to the baselines which have high recall but very low precision.

6.6 Results for Ranking

Results for News Archives. The results of precision and recall at $k = \{10, 25, 50\}$ for the news archives are shown in Table 7a. As we can observe, the ranking provided by xFactor surpasses both the Okapi BM25 and LDA baselines in terms of precision for both the New York Times and Stics archives. While, our proposed algorithm provides a high level of recall for both the news archives when compared to LDA baseline.

Results for Web Collections. The results of precision and recall at $k = \{10, 25, 50\}$ for the Web collections are shown in Table 7b. Similar to the observations made in the case of news archives, we see that the aspects extracted by xFactor at increasing ranks result in higher precision than both the baselines. Our algorithm further provides high recall at increasing rank positions as compared to the LDA baselines, while the Okapi BM25, considering all the annotations in each document outperform our system and the LDA baseline.

Overall Summary

Our experiments on two large news archives show that annotations in the form of temporal expressions, locations, and other entities can be used to identify

semantic aspects that are correct and novel for document exploration. On Web-scale corpora, where quality annotations are few, xFactor can also identify precise aspects for information consumption. Moreover, using annotation and co-occurrence salience, xFactor shows it can produce ranked list of highly-relevant aspects.

7 Related Work

Structuring Text for Search. [23] proposed TextTiling, an algorithm for identifying coherent passages (subtopics) in text documents. [28] utilized LDA to identify topics in documents for their structured representation. However, both approaches were not informed of semantic annotations, which we leverage to identify aspects for structuring text for search.

Faceted Search. Faceted Search systems allow a user to navigate document collections and prune irrelevant documents by displaying important features about them. [16,27] rely only on text to mine keyword lists present in pseudo-relevant documents for generating aspects. [7] discussed various algorithms that allowed business intelligence aggregations and advanced dynamic discovery of correlated facets across multiple dimensions. [29] leveraged semantic metadata present in Wikipedia such as entities and their associated category for automated generation of facets for exploring Wikipedia articles. [18] considered the use of named entities and their relationships in graphs for generating facets in DBpedia abstracts. Our approach, in contrast, considers the underlying semantics for each annotation during the generation of aspects. We additionally consider temporal expressions and geographic locations as additional annotations. Furthermore, we model the co-occurrences between different annotation types for generating aspects.

Temporal Search. [35] analyzed annotated documents for recommending related entities given an entity and an associated text. For this, the authors used temporal expressions, KGs and word embeddings. In a similar vein, [10] looked into incorporating temporal knowledge into embeddings for KGs. [11] on the other hand, analyze query logs to recommend time intervals for queries about events. In contrast, our approach models the uncertainty behind temporal expressions when generating query aspects. Our xFactor algorithm is additionally extensible to other annotation types (e.g., locations, numbers and sentiment). Moreover, xFactor allows query pivoting, thereby disambiguating query intent using different annotation types.

Entity Search. [30] leverage search-engine query logs to mine and suggest entities of relevance given an entity-oriented query. They consider metadata associated with the queries in the query log for their approach e.g., user clicks, user sessions, and query issue timestamps. [32] propose a method for recommending related entities for entity-centric queries using pseudo relevance feedback from retrieved documents and KGs. Their work, however, does not tap into the document contents or temporal expressions for generating aspects.

8 Conclusions

In this work, we discussed the xFactor algorithm that leverages semantic annotations such as temporal expressions, geographic locations, and other named entities to generate semantic aspects. The xFactor algorithm consists of factor methods that model the semantics of annotations in order to compute their salience. xFactor additionally considers the co-occurrence salience of annotations of different types to generate semantic aspects. Furthermore, the factor methods can be applied in different orders to disambiguate different types of ambiguities underlying the query and thereby identifying the most relevant set of semantic aspects for it. Our experiments on two types of document collections that include news archives and Web collections, show that the xFactor algorithm allows the user to navigate through messy unstructured text in a structured manner.

References

1. The ClueWeb09 dataset. http://lemurproject.org/clueweb09/
2. The ClueWeb12 dataset. http://lemurproject.org/clueweb12/
3. List of lists of lists. https://en.wikipedia.org/wiki/List_of_lists_of_lists
4. Maria Sharapova. https://en.wikipedia.org/wiki/Maria_Sharapova
5. The New York Times Annotated Corpus. https://catalog.ldc.upenn.edu/LDC2008T19
6. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: VLDB 1994, pp. 487–499 (1994)
7. Ben-Yitzhak, O., et al.: Beyond basic faceted search. In: WSDM 2008, pp. 33–44 (2008)
8. Berberich, K., Bedathur, S., Alonso, O., Weikum, G.: A language modeling approach for temporal information needs. In: Gurrin, C., et al. (eds.) ECIR 2010. LNCS, vol. 5993, pp. 13–25. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12275-0_5
9. Bhagavatula, C.S., Noraset, T., Downey, D.: TabEL: entity linking in web tables. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9366, pp. 425–441. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25007-6_25
10. Bianchi, F., Palmonari, M., Nozza, D.: Towards encoding time in text-based entity embeddings. In: Vrandečić, D., et al. (eds.) ISWC 2018. LNCS, vol. 11136, pp. 56–71. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00671-6_4
11. Nguyen, T.N., Kanhabua, N., Nejdl, W.: Multiple models for recommending temporal aspects of entities. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 462–480. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_30
12. Blei, D.M., et al.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
13. Bordino, I., et al.: Beyond entities: promoting explorative search with bundles. Inf. Retr. J. **19**(5), 447–486 (2016)
14. Ceccarelli, D., et al.: Learning relatedness measures for entity linking. In: CIKM 2013, pp. 139–148 (2013)
15. Clarke, C.L.A., et al.: Novelty and diversity in information retrieval evaluation. In: SIGIR 2008, pp. 659–666 (2008)

16. Dou, Z., et al.: Finding dimensions for queries. In: CIKM 2011, pp. 1311–1320 (2011)
17. Gabrilovich, E., et al.: FACC1: freebase annotation of ClueWeb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0), June 2013
18. Grau, B.C. et al.: SemFacet: faceted search over ontology enhanced knowledge graphs. In: ISWC 2016 (2016)
19. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proc. Natl. Acad. Sci. **101**(Suppl. 1), 5228–5235 (2004)
20. Guo, J., et al.: Named entity recognition in query. In: SIGIR 2009, pp. 267–274 (2009)
21. Gupta, D., Berberich, K.: Identifying time intervals of interest to queries. In: CIKM 2014, pp. 1835–1838 (2014)
22. Hearst, M.A.: Search User Interfaces, 1st edn. Cambridge University Press, New York (2009)
23. Hearst, M.A., Plaunt, C.: Subtopic structuring for full-length document access. In: SIGIR 1993. pp. 59–68 (1993)
24. Henry, J.: Providing knowledge panels with search results, 2 May 2013. https://www.google.com/patents/US20130110825. US Patent App. 13/566,489
25. Hoffart, J., et al.: STICS: searching with strings, things, and cats. In: SIGIR 2014, pp. 1247–1248 (2014)
26. Hoffart, J., et al.: Robust disambiguation of named entities in text. In: EMNLP 2011, pp. 782–792 (2011)
27. Kong, W., Allan, J.: Extracting query facets from search results. In: SIGIR 2013, pp. 93–102 (2013)
28. Koutrika, G., et al.: Generating reading orders over document collections. In: ICDE 2015, pp. 507–518 (2015)
29. Li, C., et al.: Facetedpedia: Dynamic generation of query-dependent faceted interfaces for Wikipedia. In: WWW 2010, pp. 651–660 (2010)
30. Reinanda, R., et al.: Mining, ranking and recommending entity aspects. In: SIGIR 2015, pp. 263–272 (2015)
31. Santos, R.L.T., et al.: Search result diversification. Found. Trends® Inf. Retr. **9**(1), 1–90 (2015)
32. Schuhmacher, M., et al.: Ranking entities for web queries through text and knowledge. In: CIKM 2015, pp. 1461–1470 (2015)
33. Strötgen, J., Gertz, M.: Multilingual and cross-domain temporal tagging. Lang. Resour. Eval. **47**(2), 269–298 (2013)
34. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a large ontology from wikipedia and wordnet. Web Semant. **6**(3), 203–217 (2008)
35. Tran, N.K., Tran, T., Niederée, C.: Beyond time: dynamic context-aware entity recommendation. In: Blomqvist, E., Maynard, D., Gangemi, A., Hoekstra, R., Hitzler, P., Hartig, O. (eds.) ESWC 2017. LNCS, vol. 10249, pp. 353–368. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58068-5_22
36. Zhang, R., et al.: Learning recurrent event queries for web search. In: EMNLP 2010, pp. 1129–1139 (2010)