Multiple Models for Recommending Temporal Aspects of Entities

Tu Ngoc Nguyen[1]([✉]) [iD], Nattiya Kanhabua[2]([✉]), and Wolfgang Nejdl[1]([✉])

[1] L3S Research Center/Leibniz Universität Hannover, Hannover, Germany
{tunguyen,nejdl}@L3S.de
[2] NTENT España, Barcelona, Spain
nkanhabua@NTENT.com

Abstract. Entity aspect recommendation is an emerging task in semantic search that helps users discover serendipitous and prominent information with respect to an entity, of which *salience* (e.g., popularity) is the most important factor in previous work. However, entity aspects are temporally dynamic and often driven by events happening over time. For such cases, aspect suggestion based solely on salience features can give unsatisfactory results, for two reasons. First, salience is often accumulated over a long time period and does not account for *recency*. Second, many aspects related to an event entity are strongly time-dependent. In this paper, we study the task of temporal aspect recommendation for a given entity, which aims at recommending the most relevant aspects and takes into account time in order to improve search experience. We propose a novel *event-centric* ensemble ranking method that learns from multiple time and type-dependent models and dynamically trades off salience and recency characteristics. Through extensive experiments on real-world query logs, we demonstrate that our method is robust and achieves better effectiveness than competitive baselines.

1 Introduction

Beyond the traditional "ten blue links", to enhance user experience with entity-aware intents, search engines have started including more semantic information, (1) suggesting related entities [4,9,30,31], or (2) supporting entity-oriented query completion or complex search with additional information or *aspects* [1,22,26]. These aspects cover a wide range of issues and include (but are not limited to) types, attributes/properties, relationships or other entities in general. They can change over time, as *public attention* shifts from some aspects to others. In order to better recommend such entity aspects, this temporal dimension has to be taken into account.

Exploiting collaborative knowledge bases such as Wikipedia and Freebase is common practice in semantic search, by exploiting anchor texts and inter-entity links, category structure, internal link structure or entity types [4]. More recently, researchers have also started to integrate knowledge bases with query logs for *temporal* entity knowledge mining [5,30]. In this work, we address *the*

temporal dynamics of recommending entity aspects and also utilize query logs, for two reasons. First, query logs are strongly entity related: more than 70% of Web search queries contain entity information [19,21]. Queries often also contain a short and very specific piece of text that represents users' intents, making it an ideal source for mining entity aspects. Second, different from knowledge-bases, query logs naturally capture temporal dynamics around entities. The intent of entity-centric queries is often triggered by a current event [17,18], or is related to "what is happening right now".

Previous work do not address the problem of temporal aspect recommendation for entities, often event-driven. The task requires taking into account the impact of temporal aspect dynamics and explicitly considering the relevance of an aspect with respect to the time period of a related event. To demonstrate the characteristics of these entity aspects, we showcase a real search scenario, where entity aspects are suggested in the form of query suggestion/auto-completion, given the entity name as a prior. Figure 1 shows the lists of aspect suggestions generated by a well-known commercial search engine for academy awards 2017 and australia open 2017. These suggestions indicate that the top-ranked aspects are mostly time-sensitive, and as the two events had just ended, the recommended aspects are timeliness-wise irrelevant (e.g., *live*, *predictions*).
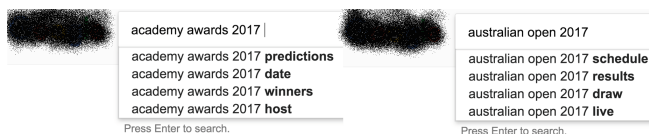

Fig. 1. [Screenshot] Recommendation generated by a commercial search engine for academy awards 2017 and australia open 2017, submitted on March 31th, 2017, on a clean history browser.

Although the exact techniques behind the search engine's recommendation are unknown, the mediocre performance might be caused by the effect of aspect salience (query popularity in this case) and the *rich get richer* phenomenon: the salience of an aspect is accumulated over a long time period. Figure 2 illustrates changes in popularity of relevant searches captured in the AOL (left) and Google (right) query logs (e.g., ncaa printable bracket, ncaa schedule, and ncaa finals) for the NCAA[1] tournament. The basketball event began on March 14, 2006, and concluded on April 3, 2006. In order to better understand this issue, we present two types of popularity changes, namely, (1) frequency or query volume (aggregated daily), and cumulative frequency. Frequencies of pre-event activities like printable bracket and schedule gain increased volume over time, especially in the *before* event period. On the other hand, up-to-date information about the event, such as, ncaa results rises in importance when the event has started

---

[1] A major sports competition in the US held annually by the National Collegiate Athletic Association (NCAA)- https://en.wikipedia.org/wiki/Ncaa.
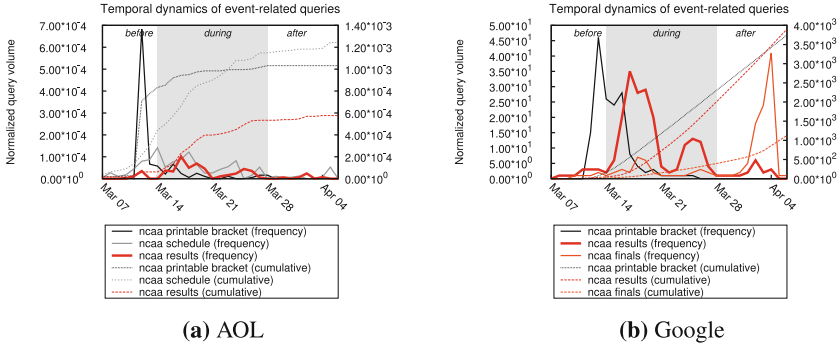
(a) AOL   **(b)** Google

Fig. 2. Dynamic aspect behaviors for entity `ncaa` in AOL and Google.

(on March 14), with very low query volume before the event. While the popularity of `results` or `finals` aspect exceeds that of `ncaa printable bracket` significantly in the periods *during* and *after* event, the cumulative frequency of the pre-event aspect stays high. We witness similar phenomenon with the same event in 2017 in the Google query logs. We therefore postulate that (1) long-term salience should provide good ranking results for the periods *before* and *during*, whereas (2) short-term or recent interest should be favored on triggers or when the temporal characteristics of an event entity change, e.g., from *before/during* to *after* phase. Different event types (`breaking` or `anticipated` events) may vary significantly in term of the impact of events, which entails different treatments with respect to a ranking model.

Our contributions can be summarized as follows.

- We present the first study of temporal entity aspect recommendation that explicitly models triggered event time and type.
- We propose a learning method to identify time period and event type using a set of features that capture temporal dynamics related to event diffusion.
- We propose a novel event-centric ensemble ranking method that relies on multiple time and type-specific models for different event entities.

To this end, we evaluated our proposed approach through experiments using real-world web search logs – in conjunction with Wikipedia as background-knowledge repository.

2   Related Work

Entity aspect identification has been studied in [22,26]. [26] focuses on salient ranking features in microblogs. Reinanda et al. [22] start from the task of mining entity aspects in the query logs, then propose salience-favor methods for ranking and recommending these aspects. When regarding an aspect as an entity, related work connected to temporal IR is [31], where they study the task of time-aware

entity recommendation using a probabilistic approach. The method also *implicitly* considers event times as triggering sources of temporal dynamics, yet relies on coarse-grained (monthly) granularity and does not recognize different phases of the event. It is therefore not really suitable for recommending fine-grained, temporal aspects. 'Static' entity recommendation was first introduced by the Spark [4] system developed at Yahoo! They extract several features from a variety of data sources and use a machine learning model to recommend entities to a Web search query. Following Spark, Sundog [9] aims to improve entity recommendation, in particular with respect to freshness, by exploiting Web search log data. The system uses a stream processing based implementation. In addition, Yu et al. [30] leverage user click logs and entity pane logs for global and personalized entity recommendation. These methods are tailored to ranking entities, and face the same problems as [31] when trying to generalize to 'aspects'.

It is also possible to relate these entity aspects to RDF properties/relations in knowledge bases such as FreeBase or Yago. [7,28] propose solutions for ranking these properties based on salience. Hasibi et al. [10] introduce dynamic fact based ranking (property-object pairs towards a sourced entity), also based on *importance* and *relevance*. These properties from traditional Knowledge Bases are often too specific (fact-centric) and temporally static.

3   Background and Problem Statement

3.1   Preliminaries

In this work, we leverage clues from entity-bearing queries. Hence, we first revisit the well-established notions of query logs and query-flow graphs. Then, we introduce necessary terminologies and concepts for entities and aspects. We will employ user log data in the form of queries and clicks.

Our datasets consist of a set of queries Q, a set of URLs U and click-through information S. Each query $q \in Q$ contains query terms $term(q)$, timestamps of queries $time(q)$ (so-called *hitting time*), and an anonymized ID of the user submitted the query. A clicked URL $u \in U_q$ refers to a Web document returned as an answer for a given query q. Click-through information is a transactional record per query for each URL clicked, i.e., an associated query q, a clicked URL u, the position on result page, and its timestamps. A co-clicked query-URL graph is a bipartite graph $G = (V, E)$ with two types of nodes: query nodes V_Q and URL nodes V_U, such that $V = V_Q \cup V_U$ and $E \subseteq V_Q \times V_U$.

3.2   Problem Definitions

We will approach the task of recommending temporal entity aspect as a ranking task. We first define the notions of an *entity query*, a *temporal entity aspect*, developed from the definition of entity aspect in [22], and an *event entity* . We then formulate the task of recommending temporal entity aspects.

Definition 1. *An entity query q_e is a query that is represented by one Wikipedia entity e. We consider q_e as the representation of e.*
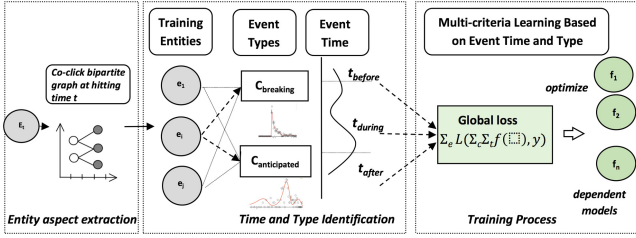
Fig. 3. Learning time and type-specific ranking models.

Definition 2. *Given a "search task" defined as an atomic information need, a temporal "entity aspect" is an entity-oriented search task with time-aware intent.* An entity-oriented search task is a set of queries that represent a common task in the context of an entity, grouped together if they have the same intent [22]. We will use the notion of query q to indicate an entity aspect a interchangeably hereafter.

Definition 3. *An entity that is related to a near event at time t_i is called an event-related entity, or event entity for short.* Relatedness is indicated by the observation that *public attention* of temporal entity aspects is triggered by the event. We can generalize the term *event entity* to represent any entity that is related to or influenced by the event. An event entity e that is associated to the event whose type $\mathscr{C}$ can be either *breaking* or *anticipated.* An event entity is also represented as a query with hitting time t. The association between t and the event time –defines e's time period $\mathscr{T}$– that can be either of the *before, during* or *after* phases of the event. When the entity is no longer event-related, it is considered a "static" entity.

Problem (Temporal Entity-Aspect Recommendation): *Given an event entity e and hitting time t as input, find the ranked list of entity aspects that most relevant with regards to e and t.*

Different from time-aware entity recommendation [27,31], for an entity query with exploratory intent, users are not just interested in related entities, but also entity aspects (which can be a topic, a concept or even an entity); these provide more complete and useful information. These aspects are very time-sensitive especially when the original entity is about an event. In this work, we use the notion of *event entity*, which is generalized to indicated related entities of any trending events. For example, Moonlight and Emma Stone are related entities for the 89th Academy Awards event. We will handle the aspects for such entities in a temporally aware manner.

4   Our Approach

As event entity identification has been well-explored in related work [14–16], we do not suggest a specific method, and just assume the use of an appropriate method. Given an event entity, we then apply our aspect recommendation

method, which is composed of three main steps. We summarize the general idea of our approach in Fig. 3. First, we extract suggestion candidates using a bipartite graph of co-clicked query-URLs generated at hitting time. After the aspect extraction, we propose a *two-step* unified framework for our entity aspect ranking problem. The first step is to identify event type and time in a joint learning approach. Based on that, in the second step, we divide the training task to different sub-tasks that correspond to specific event type and time. Our intuition here is that the timeliness (or short-term interest) feature-group might work better for specific subsets such as breaking and after events and vice versa. Dividing the training will avoid timeliness and salience competing with each other and maximize their effectiveness. However, identifying time and type of an event on-the-fly is not a trivial task, and breaking the training data into smaller parts limits the learning power of the individual models. We therefore opt for an ensemble approach that can utilize the whole training data to (1) supplement the uncertainties of the time-and-type classification in the first step and (2) leverage the learning power of the sub-models in step 2. In the rest of this section, we explain our proposed approach in more detail.

4.1   Aspect Extraction

The main idea of our approach for extracting aspects is to find related entity-bearing queries; then group them into different clusters, based on *lexical* and *semantic* similarity, such that each cluster represents a distinct aspect. The click-through information can help identifying related queries [25] by exploiting the assumption that any two queries which share many clicked URLs are likely to be related to each other.

For a given entity query e, we perform the following steps to find aspect candidates. We retrieve a set of URLs U_e that were clicked for e from the beginning of query logs until the hitting time t_e. For each $u_j \in U_e$, we find a set of distinct queries for which u_j has been clicked. We give a weight w to each query-URL by normalizing *click frequency* and *inverse query frequency* (CF-IQF) [6], which calculate the importance of a click, based on click frequency and inverse query frequency. $CF - IQF = cf \cdot log(N/(qf + 1))$, where N is the number of distinct queries. A high weight $CF - IQF$ indicates a high click frequency for the query-URL pair and a low query frequency associated with the URL in the whole query log. To extract aspect candidates from the click bipartite graph, we employ a personalized random walk to consider only one side of the query vertices of the graph (we denote this approach as **RWR**). This results in a set of related queries (aspects) to the source entity e, ranked by click-flow relatedness score. To this end, we refine these extracted aspects by clustering them using Affinity Propagation (AP) on the similarity matrix of *lexical* and *semantic* similarities. For semantic measure, we use a *word2vec* skip-gram model trained with the English Wikipedia corpus from the same time as the query logs. We pick one aspect with

highest frequency to represent each cluster, then select top-k aspects by ranking them using RWR relatedness scores[2].

4.2   Time and Type Identification

Our goal is to identify the probability that an event-related entity is of a specific event type, and in what time period of the event. We define these two targets as a joint-learning time-series classification task, that is based on event diffusion. In the following, we first present the feature set for the joint-learning task, then explain the learning model. Last we propose a light-weight clustering approach that leverages the learning features, to integrate with the ranking model in Sect. 4.3.

Features. We propose a set of time series features for our multi-class classification task. *seasonality* and *periodicity* are good features to capture the *anticipated* -recurrent events. In addition, we use additional features to model the temporal dynamics of the entity at studied/hitting time t_e. We leverage query logs and Wikipedia revision edits as the data sources for *short* and *long* span time series construction, denoted as $\psi_Q^{(e)}$ and $\psi_{WE}^{(e)}$ (for seasonal, periodical event signals) respectively[3]. The description of our features follows:

- **Seasonality** is a temporal pattern that indicates how periodic is an observed behavior over time. We leverage this time series decomposition technique for detecting not only seasonal events (e.g., Christmas Eve, US Open) [23] but also more fine-grained periodic ones that recurring on a weekly basis, such as a TV show program.
- **Autocorrelation**, is the cross correlation of a signal with itself or the correlation between its own past and future values at different times. We employ autocorrelation for detecting the trending characteristics of an event, which can be categorized by its predictability. When an event contains strong interday dependencies, the autocorrelation value will be high. Given observed time series values $\psi_1, ..., \psi_N$ and its mean $\bar{\psi}$, autocorrelation is the similarity between observations as a function of the time lag l between them. In this work, we consider autocorrelation at the one time unit lag only $(l = 1)$, which shifts the second time series by one day.
- **Correlation coefficient**, measures the dynamics of two consecutive aspect ranked lists at time t_e and $t_e - 1$, return by **RWR**. We use Goodman and

---

[2] About complexity analysis, the click bipartite graph construction costs $O(m + n)$ and RWR in practice, can be bounded by $O(m + n)$ for top-k proximity nodes. Note that m, n are the number of edges and nodes respectively. AP is quadratic $O(kn^2)$ time, (with k is the number of iterations), of our choice as we aim for a simple and effective algorithm and our aspect candidate sets are not large. A more efficient algorithm such as the Hierarchical AP can be used when candidate sets are large. The cost of constructing the similarity matrix is $O(n^2)$.

[3] Wikipedia page views is an alternative, however it is not publicly available for the time of our query logs, 2006.

Kruskal's gamma to account for possible new or old aspects appear or disappear in the newer list.

– **Level of surprise**, measured by the error margin in prediction of the learned model on the time series. This is a good indicator for detecting the starting time of *breaking* events. We use Holt-Winters as the predictive model.

– **Rising and falling signals.** The intuition behind time identification is to measure whether $\psi_Q^{(e)}$ is going up (*before*) or down (*after*) or stays trending (*during*) at hitting time. Given $\psi_Q^{(e)}$, we adopt an effective parsimonious model called SpikeM [20], which is derived from epidemiology fundamentals to predict the rise and fall of event diffusion. We use the *Levenberg-Marquardt* algorithm to learn the parameter set and use the parameters as features for our classification task.

Learning Model. We assume that there is a semantic relation between the event types and times (e.g., the before phase of *breaking* events are different from *anticipated*). To leverage the dependency between the ground labels of the two classification tasks, we apply a joint learning approach that models the two tasks in a cascaded manner, as a simple version of [11]. Given the same input instance $\mathscr{I}$, the 1st stage of the cascaded model predicts the event type $\mathscr{C}$ with all proposed features. The trained model $\mathscr{M}^1$ is used in the 2nd stage to predict the event time $\mathscr{T}$. We use the logistic regression model $\mathscr{M}_{LR}^2$ for the 2nd stage, which allows us to add additional features from $\mathscr{M}^1$. The feature vector of $\mathscr{M}_{LR}^2$ consists of the same features as $\mathscr{M}^1$, together with the probability distribution of $P(\mathscr{C}_k|e,t)$ (output of $\mathscr{M}^1$) of as additional features.

Ranking-Sensitive Time and Type Distribution. The output of an effective classifier can be directly used for determining a time and type probability distribution of entities; and thus dividing the training entities into subsets for our *divide-and-conquer* ranking approach. However, having a pre-learned model with separate and large training data is expensive and could be detrimental to ranking performance if the training data is biased. We therefore opt for effective on-the-fly *ranking-sensitive* time and type identification, following [3] that utilizes the 'locality property' of feature spaces. We adjust and refine the approach as follows. Each entity is represented as a feature vector, and consists of all proposed features with importance weights learned from a sample of training entities (for ranking). We then employ a Gaussian mixture model to obtain the centroids of training entities. In our case, the number of components for clustering are fixed before hand, as the number of event types multiplied by the number of event times. Hence the probability distribution of entity e at time t belonging to time and type $\mathscr{T}_l, \mathscr{C}_k$, $P(\mathscr{T}_l, \mathscr{C}_k|e,t)$ is calculated as $1 - \frac{\|\mathbf{x}^e - \mathbf{x}^{c_{\mathscr{T}_l, \mathscr{C}_k}}\|^2}{\max_{\forall T,C} \|\mathbf{x}^e - \mathbf{x}^{c_{\mathscr{T}_l, \mathscr{C}_k}}\|^2}$, or the distance between feature vector $\mathbf{x}^e$ and the corresponding centroid $c_{\mathscr{T}_l, \mathscr{C}_k}$.

4.3   Time and Type-Dependent Ranking Models

Learning a single model for ranking event entity aspects is not effective due to the dynamic nature of a real-world event driven by a great variety of multiple

factors. We address two major factors that are assumed to have the most influence on the dynamics of events at aspect-level, i.e., time and event type. Thus, we propose an adaptive approach based on the ensemble of multiple ranking models learned from training data, which is partitioned by entities' temporal and type aspects. In more detail, we learn multiple models, which are co-trained using data *soft* partitioning/clustering method in Sect. 4.2, and finally combine the ranking results of different models in an ensemble manner. This approach allows sub-models to learn for different types and times (where feature sets can perform differently), without hurting each other. The adaptive global loss then co-optimizes all sub-models in a unified framework. We describe in details as follows.

Ranking Problem. For aspect ranking context, a typical ranking problem is to find a function f with a set of parameters ω that takes aspect suggestion feature vector $\mathscr{X}$ as input and produce a ranking score $\hat{y}$: $\hat{y} = f(\mathscr{X}, \omega)$. In a learning to rank paradigm, it is aimed at finding the best candidate ranking model f^* by minimizing a given loss function $\mathscr{L}$ calculated as: $f^* = \arg\min_f \sum_{\forall a} \mathscr{L}(\hat{y_a}, y_a)$.

Multiple Ranking Models. We learn multiple ranking models trained using data constructed from different time periods and types, simultaneously, thus producing a set of ranking models $\mathbf{M} = \{M_{\mathscr{T}_1, \mathscr{C}_1}, \ldots, M_{\mathscr{T}_m, \mathscr{C}_n}\}$, where $\mathscr{T}_i$ is an event time period, $\in \mathscr{T}$, and $\mathscr{C} = \{\mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_n\}$ are the types of an event entity. We use an ensemble method that combines results from different ranking models, each corresponding to an identified ranking-sensitive query time $\mathscr{T}$ and entity type $\mathscr{C}$. The probabilities that an event entity e belongs to time period $\mathscr{T}_l$ and type $\mathscr{C}_k$ given the hitting time t is $P(\mathscr{T}_l, \mathscr{C}_k | e, t)$, and can be computed using the time and type identification method presented in Sect. 4.2.

$$f^* = \arg\min_f \sum_{\forall a} \mathscr{L}(\sum_{k=1}^{n} P(\mathscr{C}_k | a, t) \sum_{l=1}^{m} P(\mathscr{T}_l | a, t, \mathscr{C}_k) \hat{y_a}, y_a) \tag{1}$$

Multi-criteria Learning. Our task is to minimize the global relevance loss function, which evaluates the overall training error, instead of assuming the independent loss function, that does not consider the correlation and overlap between models. We adapted the L2R RankSVM [12]. The goal of RankSVM is learning a linear model that minimizes the number of discordant pairs in the training data. We modified the objective function of RankSVM following our global loss function, which takes into account the temporal feature specificities of event entities. The temporal and type-dependent ranking model is learned by minimizing the following objective function:

$$\min_{\omega,\xi,e,i,j} \frac{1}{2}||\omega||^2 + C\sum_{e,i,j}\xi_{e,i,j}$$

$$\text{subject to, } \sum_{k=1}^{n} P(\mathscr{C}_k|e,t)\sum_{l=1}^{m} P(\mathscr{T}_l|e,t,\mathscr{C}_k)\omega_{kl}^T X_i^e \tag{2}$$

$$\geq \sum_{k=1}^{n} P(\mathscr{C}_k|e,t)\sum_{l=1}^{m} P(\mathscr{T}_l|e,t,\mathscr{C}_k)\omega_{kl}^T X_j^e + 1 - \xi_{e,i,j},$$

$$\forall X_i^e \succ X_j^e, \xi_{e,i,j} \geq 0.$$

where $P(\mathscr{C}_k|e,t)$ is the probability the event entity e, at time t, is of type $\mathscr{C}_k$, and $P(\mathscr{T}_l|e,t,\mathscr{C}_k)$ is probability e is in this event time $\mathscr{T}_l$ given the hitting-time t and $\mathscr{C}_k$. The other notions are inherited from the traditional model ($X_i^q \succ X_j^e$ implies that an entity aspect i is ranked ahead of an aspect j with respect to event entity e. C is a trade-off coefficient between the model complexity $||\omega||$ and the training error $\xi_{a,i,j}$.

Ensemble Ranking. After learning all time and type-dependent sub models, we employ an unsupervised ensemble method to produce the final ranking score. Supposed $\bar{a}$ is a testing entity aspect of entity e. We run each of the ranking models in $\mathbf{M}$ against the instance of $\bar{a}$, multiplied by the time and type probabilities of the associated entity e at hitting time t. Finally, we sum all scores produced by all ranking models to obtain the ensemble ranking, $score(\bar{a}) = \sum_{m\in M} P(\mathscr{C}_k|e,t)P(\mathscr{T}_l|e,t,\mathscr{C}_k)\mathsf{f}^*{}_m(\bar{a})$.

4.4   Ranking Features

We propose two sets of features, namely, (1) *salience* features (taking into account the general importance of candidate aspects) that mainly mined from Wikipedia and (2) *short-term interest* features (capturing a trend or timely change) that mined from the query logs. In addition, we also leverage click-flow relatedness features computed using RWR. The features from the two categories are explained in details as follows.

***Salience* features** - or in principle, long-term prominent features.

– **TF.IDF** of an aspect a is the average $TF.IDF(w)$ of all terms $w \in a$; $TF.IDF(w)$ is calculated as $tf(w,D)\dot{}log\dfrac{N}{df(w)}$, whereas D is a section in the related Wikipedia articles C of entity e. To construct C, we take all in-link articles of the corresponding Wikipedia article of e; $tf(w,D)$ is the term frequency, $df(w)$ denotes the number of sections which w appears.
– **MLE-based**, where we reward the more (cumulated) frequently occurring aspects from the query logs. The maximum likelihood s_{MLE} is $\dfrac{sum_{w\in a}n(w,e)}{\sum_{a\prime}\sum_{t\in a\prime}f(w,e)}$, where $f(w,e)$ denotes the frequency a segment (word or phrase) $w \in a$ co-occurs with entity e.

– **Entropy-based**, where we reward the more "stable" aspects over time from the query logs. The entropy is calculated as: $\mathsf{s}_E = \sum_{t \in T} P(a|t,e) log P(a|t,e)$, where $P(a|t,e)$ is the probability of observing aspect a in the context of entity e at time t.
– **Language Model-based**, how likely aspects are generated by as statistical LM based on the textual representation of the entity $\mathsf{d}(e)$. We model $\mathsf{d}(e)$ as the corresponding Wikipedia article text. We use the unigram model with default Dirichlet smoothing.

Short-term interest features, are described as follows.

– **Temporal click entropy.** Click entropy [8] is known as the measurement of how much diversity of clicks to a particular query over time. In detail, the click entropy is measured as the query click variation over a set of URLs for a given query q. In this work, a temporal click entropy accounts for only the number of clicks on the time unit that the entity query is issued. The temporal click entropy TCE_t can be computed as $\sum_{u \in U_q} -P(u|q) \log P(u|q)$ where U_q is a set of clicked URLs for a given query q at time t. The probability of u being clicked among all the clicks of q, $P(u|q)$ is calculated as $\frac{|click(u,q)|}{\sum_{u_i \in U_q} |click(u_i,q)|}$.
– **Trending momentum** measures the trend of an aspect based on the query volume. The trending momentum at time t, Tm_t is calculated using the moving average (Ma) technique, i.e., $Tm_t = Ma(t,i_s) - Ma(t,i_l)$. Whereas, i_s,i_l denotes the short and long time window from the hitting time.
– **Cross correlation** or temporal similarity, is how correlated the aspect *wrt.* the main entity. The more cross-correlated the temporal aspect to the entity, the more influence it brings to the global trend. Given two time series ψ_t^e and ψ_t^a of the entity and aspect at time t, we employ the cross correlation technique to measure such correlation. Cross correlation $CCF(\psi_t^e, \psi_t^a)$ gives the correlation score at lagging times. Lagging time determines the time delay between two time-series. In our case, as we only interest in the hitting time, we take the maximum CCF in a lag interval of $[-1,1]$.
– **Temporal Language Model-based**, similar to the *salient* feature, only the textual representation $d(e)$ is the aggregated content of top-k most clicked URLs at time t.

5   Evaluation

In this section, we explain our evaluation for assessing the performance of our proposed approach. We address three main research questions as follows:

RQ1: How good is the classification method in identifying the most relevant event type and period with regards to the hitting time?

RQ2: How do long-term salience and short-term interest features perform at different time periods of different event types?

RQ3: How does the ensemble ranking model perform compared to the single model approaches?

In the following, we first explain our experimental setting including the description of our query logs, relevance assessment, methods and parameters used for the experiments. We then discuss experimental results for each of the main research questions.

5.1 Experimental Setting

Datasets. We use a real-world query log dataset from AOL, which consists of more than 30 million queries covering the period from March 1, to May 31, 2006. Inspired by the taxonomy of event-related queries presented in [13], we manually classified the identified events into two distinct subtypes (i.e., *Breaking* and *Anticipated*). We use Tagme[4] to link queries to the corresponding Wikipedia pages. We use the English Wikipedia dump of June, 2006 with over 2 million articles to temporally align with the query logs. The Wikipedia page edits source is from 2002 up to the studied time, as will be explained later. To count the number of edits, we measure the difference between consecutive revision pairs extracted from the Special:Export[5].

Identifying Event Entities. We reuse the event-related queryset from [14], that contains 837 entity-bearing queries. We removed queries that refer to past and future events and only chose the ones which occured in the period of the AOL dataset, which results in 300 distinct entity queries. Additionally, we construct a more recent dataset which consists of the volume of searches for 500 trending entity queries on Google Trend. The dataset covers the period from March to May, 2017. To extract these event-related queries, we relied on the Wikipedia Portal:Current events[6] as the external indicator, as we only access Google query logs via public APIs. Since the click logs are missing, the Google Trend queryset is used only as a supplementary dataset for *RQ1*.

Dynamic Relevance Assessment. There is no standard ground-truth for this novel task, so we relied on manual annotation to label entity aspects dynamically; with respect to the studied times according to each event period. We put a range of 5 days before the event time as *before* period and analogously for *after*. We randomly picked a day in the 3 time periods for the studied times. In our annotation process, we chose 70 popular and trending event entities focusing on two types of events, i.e., *Breaking* (30 queries) and *Anticipated* (40 queries). For each entity query, we make used of the top-k ranked list of candidate suggestions generated by RWR, cf. Sect. 4.1. Four human experts were asked to evaluate a pair of a given entity and its aspect suggestion (as relevant or non-relevant) with respect to the event period. We defined 4 levels of relevance: 3 (very relevant), 2 (relevant), 1 (irrelevant) and 0 (don't know). Finally, 4 assessors evaluated 1,250 entity/suggestion pairs (approximately 3,750 of triples), with approximately 17

---

[4] https://tagme.d4science.org/tagme/.
[5] https://en.wikipedia.org/wiki/Special:Export.
[6] https://en.wikipedia.org/wiki/Portal:Current_events.

suggestions per trending event on average. The average Cohen's Kappa for the evaluators' pairwise inter-agreement is k = 0.78. Examples of event entities and suggestions with dynamic labels are shown in Table 1. The relevance assessments will be made publicly available.

Table 1. Dynamic relevant assessment examples.

Entity	Suggestion	Dynamic Label		
		Before	During	After
kentucky derby	+ odds	VR	VR	R
kentucky derby	+ contenders	VR	R	R
kentucky derby	+ winner	NR	R	VR
kentucky derby	+ results	NR	VR	VR

Methods for Comparison. Our baseline method for aspect ranking is RWR, as described in Sect. 4.1. Since we conduct the experiments in a query log context, time-aware query suggestions and auto-completions (QACs) are obvious competitors. We adapted features from state-of-the-art work on time-aware QACs as follows. For the QACs' setting, entity name is given as prior. Instead of making a direct comparison to the linear models in [22] – that are tailored to a different variant of our target – we opt for the supervised-based approach, $SVM_{salient}$, which we consider a fairer and more relevant salient-favored competitor for our research questions.

Most popular completion (**MLE**) [2] is a standard approach in QAC. The model can be regarded as an approximate Maximum Likelihood Estimator (MLE), that ranks the suggestions based on past popularity. Let $P(q)$ be the probability that the next query is q. Given a prefix x, the query candidates that share the prefix $\mathcal{Q}_c$, the most likely suggestion $q \in \mathcal{Q}_c$ is calculated as: $MLE(x) = argmax_{q \in \mathcal{Q}_c} P(q)$. To give a fair comparison, we apply this on top of our aspect extraction cf. Section 4.1, denoted as $RWR + MLE$; analogously with recent MLE.

Recent MLE (**MLE-W**) [24,29] does not take into account the whole past query log information like the original MLE, but uses only recent days. The popularity of query q in the last n days is aggregated to compute $P(q)$.

Last N query distribution (**LNQ**) [24,29] differs from MLE and W-MLE and considers the last N queries given the prefix x and time x_t. The approach addresses the weakness of W-MLE in a time-aware context, having to determine the size of the sliding window for prefixes with different popularities. In this approach, only the last N queries are used for ranking, of which N is the trade-off parameter between *robust* (non time-aware bias) and *recency*.

Predicted next N query distribution (**PNQ**) employs the past query popularity as a prior for predicting the query popularity at hitting time, to use this prediction for QAC [24,29]. We adopt the prediction method proposed in [24].

Table 2. Example entities in May 2006.

Anticipated	may day, da vinci code, cinco de mayo, american idol,
	anna nicole smith, mother's day, danica patrick, emmy rossum,
	triple crown, preakness stakes, belmont stakes kentucky derby, acm awards
Breaking	david blaine, drudge report, halo 3, typhoon chanchu,
	patrick kennedy, indonesia, heather locklear

Table 3. Event type and time classification performance.

	Dataset	Model	Accuracy	Weighted F1
Event-type	AOL	Majority votes	63.7%	57.6%
		SVM	*78.7%*	*89.2%*
	GoogleTrends	Majority votes	60.8%	67.6%
		SVM	*82.7%*	*84.5%*
Event-time	AOL	Logistic regression	67.5%	71.5%
		Cascaded	*72.8%*	*83.4%*
	GoogleTrends	Logistic regression	70.8%	77.5%
		Cascaded	*74.5%*	*81.8%*

Parameters and Settings. The jumping probability for RWR is set to 0.15 (default). For the classification task, we use models implemented in Scikit-learn[7] with default parameters. For learning to rank entity aspects, we modify RankSVM. For each query, the hitting time is the same as used for relevance assessment. Parameters for RankSVM are tuned via grid search using 5-fold cross validation (CV) on training data, trade-off $c = 20$. For W-MLE, we empirically found the sliding window $W = 10$ days. The time series prediction method used for the PNQ baseline and the prediction error is Holt-Winter, available in R. In LNQ and PNQ, the trade-off parameter N is tuned to 200. The short-time window i_s for the trending momentum feature is 1-day and long i_l is 5-days. Top-k in the temporal LM is set to 3. The time granularity for all settings including hitting time and the time series binning is 1 day.

For RQ1, we report the performance on the *rolling* 4-fold CV on the whole dataset. To seperate this with the L2R settings, we explain the evaluating methodology in more details in Sect. 5.2. For the ranking on partitioned data (RQ2), we split *breaking* and *anticipated* dataset into 6 sequential folds, and use the last 4 folds for testing in a rolling manner. To evaluate the ensemble method (RQ3), we use the first two months of AOL for training (50 queries, 150 studied points) and the last month (20 queries as shown in Table 2, 60 studied points) for testing.

---

[7] http://scikit-learn.org/.

Metrics. For assessing the performance of classification methods, we measured accuracy and F1. For the retrieval effectiveness of query ranking models, we used two metrics, i.e., Normalized Discounted Cumulative Gain (NDCG) and *recall@k* (*r@k*). We measure the retrieval effectiveness of each metric at 3 and 10 (*m@3* and *m@10*, where $m \in \{NDCG, R\}$). $NDCG$ measures the ranking performance, while *recall@k* measures the proportion of relevant aspects that are retrieved in the top-k results.

5.2   Cascaded Classification Evaluation

Evaluating methodology. For **RQ1**, given an event entity e, at time t, we need to classify them into either *Breaking* or *Anticipated* class. We select a studied time for each event period randomly in the range of 5 days before and after the event time. In total, our training dataset for AOL consists of 1,740 instances of *breaking* class and 3,050 instances of *anticipated*, with over 300 event entities. For *GoogleTrends*, there are 2,700 and 4,200 instances respectively. We then bin the entities in the two datasets chronologically into 10 different parts. We set up 4 trials with each of the last 4 bins (using the history bins for training in a *rolling* basic) for testing; and report the results as average of the trials.
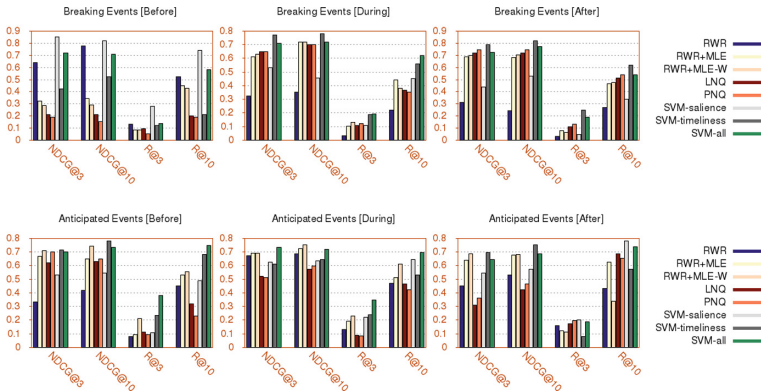

Fig. 4. Performance of different models for event entities of different types.

Results. The baseline and the best results of our 1st stage event-type classification is shown in Table 3-**top**. The accuracy for basic majority vote is high for imbalanced classes, yet it is lower at weighted F1. Our learned model achieves marginally better result at F1 metric.

We further investigate the identification of event time, that is learned on top of the event-type classification. For the gold labels, we gather from the studied times with regards to the event times that is previously mentioned. We compare the result of the cascaded model with non-cascaded logistic regression. The results are shown in Table 3-**bottom**, showing that our cascaded model, with

features inherited from the performance of SVM in previous task, substantially improves the single model. However, the overall modest results show the difficulty of this multi-class classification task.

5.3 Ranking Aspect Suggestions

For this part, we first focus on evaluating the performance of single L2R models that are learned from the pre-selected time (before, during and after) and types (*Breaking* and *Anticipate*) set of entity-bearing queries. This allows us to evaluate the feature performance i.e., *salience* and *timeliness*, with time and type specification (RQ2). We then evaluate our ensemble ranking model (results from the cascaded evaluation) and show it robustly improves the baselines for all studied cases (RQ3). Notice that, we do not use the learned classifier in Sect. 5.2 for our ensemble model, since they both use the same time period for training, but opt for the *on-the-fly* ranking-sensitive clustering technique, described in Sect. 4.2.

RQ2. Fig. 4 shows the performance of the aspect ranking models for our event entities at specific times and types. The most right three models in each metric are the models proposed in this work. The overall results show that, the performances of these models, even better than the baselines (for at least one of the three), vary greatly among the cases. In general, $SVM_{salience}$ performs well at the **before** stage of breaking events, and badly at the **after** stage of the same event type. Whereas $SVM_{timeliness}$ gives a contradictory performance for the cases. For anticipated events, $SVM_{timeliness}$ performs well at the **before** and **after** stages, but gives a rather low performance at the **during** stage. For this event type, $SVM_{salience}$ generally performs worse than $SVM_{timeliness}$. Overall, The SVM_{all} with all features combined gives a good and stable performance, but for most cases, are not better than the well-performed single set of features L2R model. In general, these results prove our assumption that *salience* and *timeliness* should be traded-off for different event types, at different event times. For feature importances, we observe regularly, stable performances of *same-group* features across these cases. *Salience* features from knowledge bases tend to perform better than from query logs for *short-duration* or less popular events. We leave the more in-depth analysis of this part for future work.

RQ3. We demonstrate the results of single models and our ensemble model in Table 4. As also witnessed in RQ2, SVM_{all}, will all features, gives a rather stable performance for both NDCG and Recall, improved the baseline, yet not significantly. Our *Ensemble* model, that is learned to trade-off between *salience* and *timeliness* achieves the best results for all metrics, outperforms the baseline significantly. As the testing entity queries in this experiment are at all event times and with all event types, these improvements illustrate the robustness of our model. Overall, we witness the low performance of adapted QAC methods. One reason is as mentioned, QACs, even time-aware generally favor already *salient* queries as follows the *rich-get-richer* phenomenon, and are not ideal for entity queries that are event-related (where aspect relevance can change abruptly).

Table 4. Performance of the baselines (RWR relatedness scores, RWR + MLE, RWR + MLE-W, LNQ, and PNQ) compared with our ranking models; $*, \dagger, \mp$ indicates statistical improvement over the baseline using t-test with significant at $p < 0.1$, $p < 0.05$, $p < 0.01$ respectively.

Methods	NDCG@3	NDCG@10	R@3	R@10
RWR	*0.3208*	*0.4137*	*0.1208*	*0.3749*
RWR + MLE	+29.94%	+9.73%	−21.09%	+5.15%*
RWR + MLE-W	+11.56%	+11.46%	−18.93%*	+3.28%
LNQ	+15.39%	−3.75%	−19.74%	−30.31%
PNQ	+13.19%	−9.95%	−23.46%	−33.53%
$SVM_{salience}$	+41.75%*	+9.18%	+23.32%*	+9.93%
$SVM_{timeliness}$	+15.19%	+17.53%	+14.77%	+11.3%
SVM_{all}	+52.65%*	+40.87%*	+9.73%†	+24.3%
Ensemble	**+85.12%∓**	**+45.34%†**	**+42.78%***	**+17.45%***

Time-aware QACs for partially long prefixes like entities often encounter sparse traffic of query volumes, that also contributes to the low results.

6 Conclusion

We studied the temporal aspect suggestion problem for entities in knowledge bases with the aid of real-world query logs. For each entity, we ranked its temporal aspects using our proposed novel time and type-specific ranking method that learns multiple ranking models for different time periods and event types. Through extensive evaluation, we also illustrated that our aspect suggestion approach significantly improves the ranking effectiveness compared to competitive baselines. In this work, we focused on a "global" recommendation based on public attention. The problem is also interesting taking other factors (e.g., *search context*) into account, which will be interesting to investigate in future work.

References

1. Balog, K., Dalton, J., Doucet, A., Ibrahim, Y.: Report on esair'15. In: ACM SIGIR Forum
2. Bar-Yossef, Z., Kraus, N.: Context-sensitive query auto-completion. In: WWW 2011 (2011)
3. Bian, J., Li, X., Li, F., Zheng, Z., Zha, H.: Ranking specialization for web search: a divide-and-conquer approach by using topical ranksvm. In: WWW 2010 (2010)

4. Blanco, R., Cambazoglu, B.B., Mika, P., Torzec, N.: Entity recommendations in web search. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) ISWC 2013. LNCS, vol. 8219, pp. 33–48. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41338-4_3

5. Chirigati, F., Liu, J., Korn, F., Wu, Y.W., Yu, C., Zhang, H.: Knowledge exploration using tables on the web. In: Proceedings of the VLDB Endowment (2016)

6. Deng, H., King, I., Lyu, M.R.: Entropy-biased models for query representation on the click graph. In: Proceedings of SIGIR 2009 (2009)

7. Dessi, A., Atzori, M.: A machine-learning approach to ranking RDF properties. Future Gener. Comput. Syst. **54**, 366–377 (2016)

8. Dou, Z., Song, R., Wen, J.-R.: A large-scale evaluation and analysis of personalized search strategies. In: Proceedings of WWW 2007 (2007)

9. Fischer, L., Blanco, R., Mika, P., Bernstein, A.: Timely semantics: a study of a stream-based ranking system for entity relationships. In: Arenas, M., Corcho, O., Simperl, E., Strohmaier, M., d'Aquin, M., Srinivas, K., Groth, P., Dumontier, M., Heflin, J., Thirunarayan, K., Staab, S. (eds.) ISWC 2015. LNCS, vol. 9367, pp. 429–445. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25010-6_28

10. Hasibi, F., Balog, K., Bratsberg, S.E.: Dynamic factual summaries for entity cards. In: SIGIR 2017 (2017)

11. Heitz, G., Gould, S., Saxena, A., Koller, D.: Cascaded classification models: combining models for holistic scene understanding. In: NIPS (2009)

12. Joachims, T.: Training linear svms in linear time. In: Proceedings of KDD 2006 (2006)

13. Kairam, S.R., Morris, M.R., Teevan, J., Liebling, D.J., Dumais, S.T.: Towards supporting search over trending events with social media. In: ICWSM (2013)

14. Kanhabua, N., Ngoc Nguyen, T., Nejdl, W.: Learning to detect event-related queries for web search. In: WWW 2015 Companion. ACM (2015)

15. Kanhabua, N., Ren, H., Moeslund, T.B.: Learning dynamic classes of events using stacked multilayer perceptron networks. CoRR, abs/1606.07219 (2016)

16. Karmaker Santu, S.K., Li, L., Park, D.H., Chang, Y., Zhai, C.: Modeling the influence of popular trending events on user search behavior. In: WWW 2017 (2017)

17. Kong, W., Li, R., Luo, J., Zhang, A., Chang, Y., Allan, J.: Predicting search intent based onl pre-search context. In: SIGIR 2015 (2015)

18. Kulkarni, A., Teevan, J., Svore, K.M., Dumais, S.T.: Understanding temporal query dynamics. In: Proceedings of WSDM 2011 (2011)

19. Lin, T., Pantel, P., Gamon, M., Kannan, A., Fuxman, A.: Active objects: actions for entity-centric search. In: WWW 2012 (2012)

20. Matsubara, Y., Sakurai, Y., Prakash, B.A., Li, L., Faloutsos, C.: Rise and fall patterns of information diffusion: model and implications. In: Proceedings of KDD. ACM (2012)

21. Pound, J., Mika, P., Zaragoza, H.: Ad-hoc object retrieval in the web of data. In: WWW 2010 (2010)

22. Reinanda, R., Meij, E., de Rijke, M.: Mining, ranking and recommending entity aspects. In: Proceedings of SIGIR, pp. 263–272. ACM (2015)

23. Shokouhi, M.: Detecting seasonal queries by time-series analysis. In: SIGIR 2011 (2011)

24. Shokouhi, M., Radinsky, K.: Time-sensitive query auto-completion. In: IGIR 2012 (2012)

25. Silvestri, F.: Mining query logs: turning search usage data into knowledge. Found. Trends Inf. Retrieval **4**(1–2), 1–174 (2010)
26. Spina, D., Meij, E., De Rijke, M., Oghina, A., Bui, M.T., Breuss, M.: Identifying entity aspects in microblog posts. In: SIGIR 2012 (2012)
27. 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
28. Vadrevu, S., Tu, Y., Salvetti, F.: Ranking relevant attributes of entity in structured knowledge base. US Patent 9,229,988, 5 Jan 2016
29. Whiting, S., Jose, J.M.: Recent and robust query auto-completion. In: WWW 2014 (2014)
30. Yu, X., Ma, H., Hsu, B.-J.P., Han, J.: On building entity recommender systems using user click log and freebase knowledge. In: Proceedings of WSDM, pp. 263–272. ACM (2014)
31. Zhang, L., Rettinger, A., Zhang, J.: A probabilistic model for time-aware entity recommendation. In: Groth, P., Simperl, E., Gray, A., Sabou, M., Krötzsch, M., Lecue, F., Flöck, F., Gil, Y. (eds.) ISWC 2016. LNCS, vol. 9981, pp. 598–614. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46523-4_36