An Event-Based Framework for the Semantic Annotation of Locations

Anh Le, Michael Gertz, and Christian Sengstock

Database Systems Research Group, Heidelberg University, Germany
{anh.le.van.quoc,gertz,sengstock}@informatik.uni-heidelberg.de

Abstract. There is an increasing number of Linked Open Data sources that provide information about geographic locations, e.g., GeoNames or LinkedGeoData. There are also numerous data sources managing information about events, such as concerts or festivals. Suitably combining such sources would allow to answer queries such as '*When and where do live-concerts most likely occur in Munich?*' or '*Are two locations similar in terms of their events?*'. Deriving correlations between geographic locations and event data, at different levels of abstraction, provides a semantically rich basis for location search, topic-based location clustering or recommendation services. However, little work has been done yet to extract such correlations from event datasets to annotate locations.

In this paper, we present an approach to the discovery of semantic annotations for locations from event data. We demonstrate the utility of extracted annotations in hierarchical clustering for locations, where the similarity between two locations is defined on the basis of their common event topics. To deal with periodic updates of event datasets, we furthermore give a scalable and efficient approach to incrementally update location annotations. To demonstrate the performance of our approach, we use real event datasets crawled from the Website *eventful.com*.

1 Introduction

The main difference between a '*place*' and a position is that a place is represented as a human-readable description of a geographic location rather than just a geographic coordinate. Such descriptive information about locations is essential for location-based services (LBS), for instance, location recommendation or social event recommendation [5,6]. Typically, a data source managing information about locations provides various attributes of a location for an LBS application, including the name, address, description, and metadata such as tags. From a semantic perspective, such description or tags associated with a location are useful in semantic location search.

Unfortunately, such descriptive attributes detailing location information tend to be poor in many data sources. For example, based on our analysis, there are about one million locations in a dataset of events crawled for the years 2011 and 2012 from the Website *eventful.com*, but only 10% of them contain descriptions or tags. Moreover, querying based on simple text matching of descriptions and tags cannot take into account concept hierarchies that might exist for locations, time, or event topics. For example, using suitable concept hierarchies '*live jazz on Saturdays*' may be considered a match for '*live music on weekends*'. Therefore, enriching information about events at different levels of granularity and abstraction is necessary and useful.

Several methods have been proposed to extract semantic annotations for locations. However, some of them highly depend on the location data provided by external sources such as Wikipedia or the Google Maps API [2]. Other approaches exploit either user-tags (e.g., in the context of Flickr data) [11,12] or the user behavior (e.g., check-in data or user-interest-profiles) extracted from online social networks [6,15]. Such types of user generated content are often sparse, noisy and sometimes even inaccurate.

On the other hand, numerous data sources managing information about events are available on the Internet. This includes popular Websites such as *last.fm*, *eventim.de*, or *eventful.com.* Although in these sources the event data are less noisy (and more accurate) than in other georeferenced social media, there are still challenges in fully exploiting such information, as concept hierarchies, either explicitly or implicitly, exist for event topics, locations, and time.

Intuitively, some events occur more likely at some place/time than at other places/times. For example, events related to the topics '*live music*' or '*dance*' likely occur at a bar or club at weekends, whereas events related to '*conference*' or '*talk*' likely occur at a university on working days. Following this, we aim at extracting semantic annotations for locations from event data on the basis of exploiting correlations among geographic locations, time and event topics.

In this paper, we propose a framework to extract location annotations from event data. Our framework is based on the concept of a Location-Time-pair Class (LTC) to describe a group of location-time pairs that have the same location and time concepts, e.g., ['*Stadium*','*Weekend*']. We define a measure to identify significant event topics with respect to an LTC, based on *Pointwise Mutual Information* [14]. A set of significant topics with respect to an LTC is called a Location-Time-pair Profile (LTP). LTPs are utilized to derive semantic annotations for locations, where an annotation is a pair of an event topic and a time concept, e.g., ['*Live-music*','*Weekend*']. Figure 1 shows the components of our framework. The *LTProfile-Miner* component extracts LTPs from event datasets. To efficiently deal with (periodic) updates to event data, the component *LTProfile-Updater* updates the current set of location annotations. With the latter component, we provide a scalable and efficient approach to deal with large datasets that do not fit in main memory.

Based on external sources such as Wikipedia, extracted annotations of famous locations, e.g., stadiums or theatres, can be manually validated. Since there is no pre-existing ground-truth to validate all results obtained from a given dataset, we indirectly measure how good the extracted annotations are with location clustering. In summary, the contributions of this paper are as follows:

- We model semantic annotations for locations based on the concepts of events and event topics.

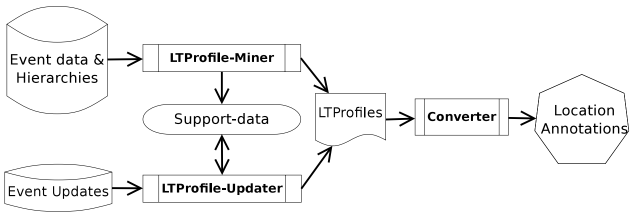


Fig. 1. Conceptual framework for annotating event locations

- We propose a measure based on *Pointwise Mutual Information* to identify significant event topics from a dataset of events.
- We develop two approaches: *LTProfile-Miner* to derive location annotations from an event dataset, and *LTProfile-Updater* to deal with periodic updates of that dataset.
- We demonstrate the utility of extracted annotations in semantic location search and clustering by using real event data.

In the following section, we discuss related work. In Section 3, we introduce the basic concepts and notations. We describe our method to extract semantic annotations for locations in Section 4. After presenting some experimental results in Section 5, we summarize the paper in Section 6.

2 Related Work

Basically, the term '*annotation*' means to attach information (metadata) to existing data. An example is that Flickr users add tags to photos to describe the photos. Since such human effort-based annotation systems are often noisy and incomplete, there have been many approaches to *automatically* annotate objects in different formats such as textual documents or photos, e.g., [4,7]. However, extracting annotations from spatio-temporal data like event data raises many challenges, e.g., annotations might differ among regions as well as over time, as discussed in [5]. Thus, such approaches cannot be directly applied to extract location annotations from event datasets. Nevertheless, the idea of *word-context matrices* and a statistical measure successfully used in annotating textual documents, called *Pointwise Mutual Information* [10,14], can be utilized to estimate correlations among locations, time, and event topics. This will be described in more detail in Section 3.2.

Several approaches are similar to our work in extracting annotations from spatio-temporal data. One direction of research relies on location information from external sources, such as the Google Maps API to annotate locations [2,3]. These approaches first extract points of interest (e.g., *stops* from trajectory data), and then annotate them with place categories (e.g., '*hotel*' or '*museum*') using external sources. Different from these approaches, we aim at extracting not only place categories but also relevant event topics w.r.t. a given location, important information that cannot be obtained from the above data sources.

Another direction of research aims at exploiting georeferenced social media to describe and annotate *geographic space*. Rattenbury and colleagues proposed several spatial clustering methods to identify Flickr tags corresponding to places and/or events [11,12]. Such tags can then be used to annotate geographic space on the basis of discovered clusters. Similarly, the approach in [13] aims at extracting latent geographic place semantics from Flickr data. Geographic space can then be annotated using spatial distributions and coefficients of extracted features. Although the above approaches focus on extracting annotations from spatio-temporal data, they are only able to annotate geographic space in general and not specific locations. Furthermore, these approaches do not explicitly model locations, in particular, they do not consider location hierarchies.

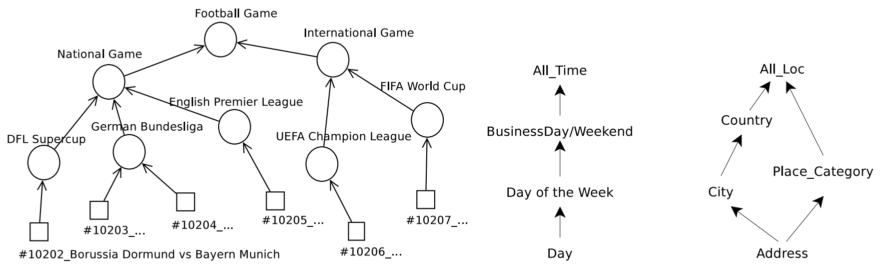
To the best of our knowledge, little work has been done yet to annotate locations with semantic tags. The most related work is [15], where a technique is proposed to annotate places with categorical tags such as '*restaurant*' or '*cinema*' by utilizing user check-in data. Similarly, the approach in [6] exploits check-in data to enrich places with semantic tags that are extracted from user interest-profiles on social networks. These approaches rely on characteristics of check-in data consisting of hidden user behaviors. Thus, they cannot be applied for event datasets for the following reasons. First, these approaches require a significant number of check-in records at a particular location and time to derive user behaviors. However, only few events occur at a particular location and time in an event dataset. Moreover, a check-in record as described in these approaches is a triple $\langle user\_id, time, location \rangle$ that does not contain semantic tags like an event description. Thus, in their approaches, a candidate set of tags for locations needs to be either predefined or obtained from an external source (user interest-profiles). Such a predefined set of tags is often small and only contains general, categorical tags. Rather than focusing on categorical tags like the above approaches, we aim at extracting more informative tags that can be used to discriminate one location from another. We also take concept hierarchies for time, locations and event topics into account, an important and useful piece of information not considered by the above approaches.

3 Basic Concepts and Notations

In the two following sections, we describe the concepts of events and event components to model semantic annotations for locations.

3.1 Events and Event Components

In this paper, an event is specified as a tuple $\langle e_{id}$, C, T, L$\rangle$, where the first component (e_{id}) is the event identifier and the last three are the event topic (context), time, and location, respectively, of that event. For example, a tuple $e =$ $\langle$*'#10202','Borussia Dortmund vs Bayern Munich', '2013-06-27', 'Signal Iduna Park'*$\rangle$ describes a football match. The topic, time, and location components of an event can be generalized to higher levels of abstraction and granularity, based on hierarchies, as detailed below.



(a) For event topics related to football (b) For time (c) For locations

Fig. 2. Example of hierarchies for concepts (event topics), time, and locations

In data sources managing information about public activities (e.g., festivals or sports), an event topic (ET) is typically provided as a textual description, e.g., '*Borussia Dortmund vs Bayern Munich*' for a football game. An ET can be generalized to higher levels of abstraction, based on a concept hierarchy. For example, Figure 2(a) shows a simple hierarchy related to football, where the ET '*Borussia Dortmund vs Bayern Munich*' can be generalized to '*DFL Supercup*', '*National Game*', and then '*Football Game*'. Such a hierarchy might be explicitly provided, or it can be built using a learning approach.

We employ the operator $\Uparrow$ to compute the set of all generalizations for an ET in a given hierarchy. For example, '*DFL Supercup*'$^{\Uparrow}$ is the set {'*National Game*', '*Football Game*'} based on the hierarchy shown in Figure 2(a). Event topics and their generalizations are key ingredients of semantic annotations for locations.

The time component of an event is typically specified as a time point. Since we focus on events such as festivals or sports, we assume that the time of an event is of granularity *Day*. Based on a predefined time hierarchy, an event time can be generalized to time concepts, e.g., a time point '*2013-06-27*' can be generalized to '*Friday*'→'*BusinessDay*'→'*All_Time*', based on the time hierarchy in Figure 2(b). We also use the operator $\Uparrow$ to compute the set of all generalizations of a time point, e.g., '*2013-06-27*'$^{\Uparrow}$={'*Friday*', '*BusinessDay*', '*All_Time*'}.

Finally, the location component of an event is specified at a location granularity. Since an event like a concert or a football game takes place at a particular location, we assume that locations of events are of granularity *Address*. They can be generalized to a coarser granularity like *City* or to a location concept (place category) like '*Stadium*', again based on a predefined hierarchy. Also here we use the $\Uparrow$ operator to specify the generalizations of a location. For example, '*Signal Iduna Park*'$^{\Uparrow}$ is the set {'*Dortmund*', '*Germany*', '*Stadium*', '*All_Loc*'}, based on the hierarchy in Figure 2(c).

Given an event topic f[1], a time concept T, and a location L, one might find some events whose respective components are related to f, T, and L from a given event dataset. The more such events are found, the more significant the association of f, T, and L is. In reality, some associations are more significant than others. For example, it is more likely to find events related to ice skating in

[1] In this paper, we often use 'e' to denote an event and 'f' to denote an event topic.

Winter than in Summer, or it is more likely to find a rock festival in some cities than in other cities. To model such associations, we introduce the concepts of *Location-Time-pair Instance* and *Location-Time-pair Class.*

3.2 Location-Time-Pair Instances and Classes

Let $\mathcal{D}$ be a dataset of events as $\langle e_{id}, \mathrm{C}, \mathrm{T}, \mathrm{L}\rangle$ tuples, where the time and location components of each event are of granularity *Day* and *Address*, respectively. To formulate the probability to find an event topic (ET) at a given location and time later on, we define a *Location-Time-pair Instance* (LTI) as a pair $[l, t]$, where l and t are the location and time of some event in $\mathcal{D}$. We use $\mathcal{D}[l, t]$ to denote a subset of $\mathcal{D}$ consisting of events whose location and time components are l and t, respectively. The set of LTIs with respect to a given dataset $\mathcal{D}$ of events is defined as

$$\mathcal{I}(\mathcal{D}) := \{[l, t] \mid \exists e \in \mathcal{D}, e.L = l \ \wedge \ e.T = t\}. \tag{1}$$

As mentioned before, a location l can be generalized to a concept L based on a given location hierarchy. Similarly, a time point t can be generalized to a time concept T, based on a time hierarchy. The pair $[L, T]$ is called a *Location-Time-pair Class* (LTC) and the LTI $[l, t]$ is called an *instance* of that LTC. For example, one can infer that [‘*Signal Iduna Park*’, ‘*2013-07-28*’] is an instance of [‘*Stadium*’, ‘*Weekend*’]. The LTC set of a given event dataset $\mathcal{D}$ is defined as

$$\mathcal{C}(\mathcal{D}) := \{[L, T] \mid \exists [l, t] \in \mathcal{I}(\mathcal{D}), L \in l^{\Uparrow} \wedge T \in t^{\Uparrow}\}. \tag{2}$$

Given an LTC, it is straightforward to retrieve the set of its instances (LTIs). For example, {[‘*Signal Iduna Park*’, ‘*2012-05-01*’], [‘*Signal Iduna Park*’, ‘*2013-07-28*’], [‘*Allianz Arena*’, ‘*2013-07-28*’],...} is the set of instances for [‘*Stadium’, ‘Weekend*’]. For each LTI, event topics can then be derived from events in that LTI. Therefore, it is reasonable to determine the correlation between a given LTC and an ET based on the occurrences of that ET in the LTC. Clearly, ETs that are strongly related to an LTC are important to represent the characteristics of that LTC. For example, topics such as ‘*football*’ or ‘*sport*’ are expected for [‘*Stadium’, ‘Weekend’*], whereas ‘*drink*’ or ‘*live music*’ are expected for [‘*Bar/Club*’,‘*Weekend*’].

To formulate correlations as the ones mentioned above, we employ the *Pointwise Mutual Information (pmi)* [14], commonly used in Computational Linguistics. The *pmi* value for an ET f with respect to an LTC Ω is computed based on the two following probabilities: (1) the probability to find f at any instance (LTI) of Ω, i.e., the conditional probability $\mathrm{P}(f|\Omega)$, and (2) the probability to find f at any LTI in the dataset, i.e., $\mathrm{P}(f)$. More precisely, the measure is computed as: $pmi(f; \Omega) = \log\left(\frac{P(f|\Omega)}{P(f)}\right) = \log\left(\frac{P(f,\Omega)}{P(f)P(\Omega)}\right)$.

The *pmi* value of an ET f with respect to an LTC Ω represents the logarithmic difference between the two probabilities $\mathrm{P}(f|\Omega)$ and $\mathrm{P}(f)$. Thus, the *pmi* can be zero, positive or negative. If it is zero, i.e., $\mathrm{P}(f|\Omega) = \mathrm{P}(f)$, f and Ω are independent. If the value is positive, i.e., $\mathrm{P}(f|\Omega) > \mathrm{P}(f)$, the events related to f occur more likely at Ω than at other LTCs. If the value is negative, i.e., $\mathrm{P}(f|\Omega)$

< P(f), the events related to f more rarely occur at Ω than at other LTCs. The *pmi* measure can be normalized to a value between [-1,+1], where -1 means negatively correlated, 0 for independence, and +1 for perfectly correlated [1].

Definition 1. ***(Normalized Pmi)*** *Given an event topic f and an LTC Ω, the* ***normalized pointwise mutual information (npmi)*** *of f and Ω is defined as:*

$$npmi(f;\Omega) := \frac{pmi(f;\Omega)}{-\log\left(P(f,\Omega)\right)} = \frac{\log\left(\frac{P(f,\Omega)}{P(f)P(\Omega)}\right)}{-\log\left(P(f,\Omega)\right)} \quad \in [-1,1]. \tag{3}$$

Since the *npmi* represents the difference between the probabilities P($f|\Omega$) and P(f), it typically gives an ET a high score with respect to a given LTC if the ET frequently occurs at that LTC but rarely at other LTCs. For example, with sport events crawled from the Website *eventful.com*, the topics '*borussia*', '*dormund*', or '*bundesliga*' get higher *npmi* scores than the topics '*football*' or '*soccer*' with respect to an LTC ['*Signal Iduna Park*', '*Weekend*'][2]. One can see that the first three topics are better to identify that LTC, and thus, they have priority over the last two topics to annotate the location '*Signal Iduna Park*'.

Another advantage of the *npmi* measure is as follows. Since a frequency-based measure like *tf-idf* always gives a non-negative value, it is not trivial for the user to pick a good threshold in order to filter out irrelevant ETs. On the other hand, a non-positive *npmi* value indicates an insignificant correlation between an ET and an LTC. Thus, one can use any positive threshold δ to filter out irrelevant ETs (whose *npmi* values are zero or negative). Based on a positive threshold δ, one can select only ETs that have significant correlations to a given LTC. A set of such event topics is called a *Location-Time-Profile.*

Definition 2. ***(Location-Time-Profile)*** *Let $\mathcal{D}$ be a dataset of events and Ω be an LTC in $\mathcal{C}(\mathcal{D})$. The* ***profile*** *of Ω with respect to a given threshold $\delta > 0$ is a set of ETs, defined as $Profile(\Omega) := \{f \in e.C^{\Uparrow} |\; e \in \mathcal{D} \wedge npmi(f;\Omega) \geq \delta\}$.*

For a particular purpose such as location clustering where feature selection can be viewed as a form of weighting, both *npmi* and *tf-idf* can be used. However, as shown in our experiments later on, the *npmi* measure performs better than *tf-idf* when considering semantic similarity between locations.

We now present our method to compute the *npmi* for a given ET f with respect to an LTC Ω, based on a given event dataset $\mathcal{D}$. For this, we count the LTIs that support f, where an LTI $[l,t]$ *supports* f iff there exists an event $e \in \mathcal{D}[l,t]$ such that e is an instance of f. Based on that, we estimate the probabilities P(f,Ω), P(f), and P(Ω) as follows.

Let N be the size of the LTI set (i.e., $\mathcal{I}(\mathcal{D})$), N_f the number of LTIs in $\mathcal{I}(\mathcal{D})$ that support f, N_Ω the number of LTIs in $\mathcal{I}(\mathcal{D})$ that are instances of Ω,

[2] Signal Iduna Park is the home stadium of the Borussia Dortmund football team playing in the German Bundesliga.

and $N_{f,\Omega}$ the number of instances of Ω that support f. The above probabilities are estimated as: $P(f,\Omega) = \frac{N_{f,\Omega}}{N}$, $P(f) = \frac{N_f}{N}$, and $P(\Omega) = \frac{N_\Omega}{N}$. Thus,

$$npmi(f;\Omega) = \frac{\log\left(\frac{P(f,\Omega)}{P(f)P(\Omega)}\right)}{-\log(P(f,\Omega))} = \frac{\log\left(\frac{N_{f,\Omega}N}{N_f N_\Omega}\right)}{-\log\left(\frac{N_{f,\Omega}}{N}\right)} = \frac{\log\left(\frac{N_{f,\Omega}N}{N_f N_\Omega}\right)}{\log\left(\frac{N}{N_{f,\Omega}}\right)}. \tag{4}$$

4 LT-Profiles and Applications

In this section, we first introduce a novel algorithm to generate LT-Profiles from an event dataset, and a scalable and efficient method to deal with periodic updates of the input data. We then show how to convert such profiles into location annotations. Finally, we describe how to exploit such information in semantic location search and clustering.

4.1 Generating Location-Time-Profiles

Given a dataset $\mathcal{D}$ of events, a set $\mathcal{H}$ of hierarchies for generating ETs from events and for generating location and time concepts, and a *npmi* threshold δ, this section describes a procedure called *LTProfile-Miner* to determine all profiles as defined in Definition 2.

Based on the Formulas (1) and (2), generating the set of LTIs ($\mathcal{I}(\mathcal{D})$) and the set of LTCs ($\mathcal{C}(\mathcal{D})$) is straightforward. For each LTC $\Omega \in \mathcal{C}(\mathcal{D})$, the set of ETs belonging to Ω is generated from all events that belong to any instance (LTI) of Ω. For each ET f in this set, the value of $npmi(f;\Omega)$ needs to be computed and compared with respect to the threshold δ. This can easily be done by counting LTIs in the set $\mathcal{I}(\mathcal{D})$ and then applying Formula (4). However, such a method is inefficient since the set $\mathcal{I}(\mathcal{D})$ will be scanned multiple times for all LTC-ET pairs. Thus, we propose a more efficient method as follows.

We utilize two data structures, called *Support_ET* and *Support_LTC*, where each one is a hash table mapping keys to LTI sets. Given an ET f, the set of LTIs that support f is retrieved by using the hash table *Support_ET*. This set is denoted *Support_ET*$[f]$. Let n_f, n_i, and n_{ei} be the number of ETs, the number of LTIs ($|\mathcal{I}(\mathcal{D})|$), and the average number of events of an LTI, respectively. The runtime complexity to build *Support_ET* is $O(n_f n_i n_{ei})$, since each element (with respect to an ET) is computed by scanning through all the LTIs and considering all events inside each LTI. Similarly, the hash table *Support_LTC* is used to retrieve the set of LTIs that are instances of a given LTC Ω, denoted *Support_LTC*$[\Omega]$. The complexity to build *Support_LTC* is $O(n_c n_i)$, where n_c is the number of LTCs ($|\mathcal{C}(\mathcal{D})|$) and n_i is the number of LTIs ($|\mathcal{I}(\mathcal{D})|$).

Utilizing hash tables allows the values N_f, N_Ω, and $N_{f,\Omega}$ in Equation (4) to be computed with several set operations: $N_f = |Support\_ET[f]|$, $N_\Omega = |Support\_LTC[\Omega]|$, and $N_{f,\Omega} = |Support\_ET[f] \bigcap Support\_LTC[\Omega]|$. Thus, the value of $npmi(f;\Omega)$ for each pair of an LTC Ω and ET f can easily be computed. Finally, the profile of each LTC Ω in $\mathcal{C}(\mathcal{D})$ is obtained based on Definition 2.

4.2 Updating Location-Time-Profiles

In the previous section, we presented an approach to extract all *LTProfiles* from a given dataset of events. Such a dataset consists of events in a certain time-interval (e.g., [2011,2012]). Thus, the extracted profiles are only valid in this interval. In reality, datasets are incrementally updated. For example, events in 2013 are added to a dataset of events in [2011,2012]. Running again that procedure for the merged dataset is a possible solution, which, however, is neither efficient nor scalable. To adapt to periodic updates of event data, we propose another procedure, called *LTProfile-Updater*.

Assume that after executing *LTProfile-Miner*, the following intermediate values are stored on a secondary storage: N, N_f (as an element of a list), N_Ω (as an element of a list), $N_{f,\Omega}$ (as an element of a matrix). Such data, called *support-data*, contain sufficient information to extract profiles without considering the original (previous) dataset $\mathcal{D}$.

Let $\mathcal{D}^*$ be the dataset of new events to update. It is reasonable to assume that each event in $\mathcal{D}^*$ occurred after all events in $\mathcal{D}$, i.e., events in $\mathcal{D}^*$ are newer than events in $\mathcal{D}$. Therefore, there is no overlap between the LTI sets of the two datasets. Thus, the values of N, N_f, N_Ω and $N_{f,\Omega}$ can be updated as: $N = N + |\mathcal{I}(\mathcal{D}^*)|$, $N_f = N_f + |Support\_ET^*[f]|$, $N_\Omega = N_\Omega + |Support\_LTC^*[\Omega]|$, and $N_{f,\Omega} = N_{f,\Omega} + |Support\_ET^*[f] \bigcap Support\_LTC^*[\Omega]|$. Note that $Support\_ET^*$ and $Support\_LTC^*$ are two hash tables computed from the update ($\mathcal{D}^*$) with the method described in the previous section.

Summing up, *LTProfile-Updater* first loads the *support-data* and then combines it with the update ($\mathcal{D}^*$) to update the current location profiles. Since this procedure utilizes the *support-data*, only the update $\mathcal{D}^*$ is scanned.

Based on *LTProfile-Updater*, an *anytime* approach to deal with very large datasets works as follows. First, the events in a (large) dataset $\mathcal{D}$ are sorted by the time attribute and distributed in increasing order into sub-datasets $\mathcal{D}_0$, $\mathcal{D}_1, \ldots$ such that each $\mathcal{D}_i$ fits into main memory. *LTProfile-Miner* is then called to compute the *support-data* from $\mathcal{D}_0$. Finally, *LTProfile-Updater* is iteratively called for each $\mathcal{D}_i$ ($i \geq 1$). If the mining process is interrupted after processing $\mathcal{D}_i$, the results are valid until the latest time in $\mathcal{D}_i$.

4.3 Location Annotations

Location-Time-Profiles, each consisting of significant ETs at an LTC, can be exploited to annotate locations. Here, we define a location annotation as a set, where each element is a pair of an event topic and a time concept. For example, annotation elements for a specific bar/club might be ['*jazz*', '*Tuesday*'] or ['*dancing*', '*Weekend*']. The formal definition is given as follows.

Definition 3. ***(Location Annotation)*** *Let $\mathcal{D}$ be an event dataset. The* ***annotation*** *of a location (or location concept) L is a set defined as:*
$Annotation(L) := \{[f, T] \mid \Omega = [L, T] \in \mathcal{C}(\mathcal{D}),\ f \in Profile(\Omega)\}.$

4.4 Similarity Measure for Location Search and Clustering

To determine how similar two locations are, we define a similarity measure for locations based on events. Basically, the more common event topics two locations have, the more similar they are. Given two locations L_1 and L_2, and their annotations AL_1 and AL_2, respectively, the similarity between the two locations is computed based on the Jaccard Index as: $sim(L_1, L_2) = \frac{|AL_1 \cap AL_2|}{|AL_1 \cup AL_2|} \in [0, 1]$.

This measure can be used to find locations that are similar to a given location or just to rank the results, for example, in the query '*Find all cities in the US like Munich (in Germany) in terms of beer festivals*'.

To apply clustering, the dissimilarity distance between two locations L_1, L_2 is computed as $dist(L_1, L_2)$=1-$sim(L_1, L_2)$. Based on that, locations can be clustered with one of the various clustering algorithms, e.g., hierarchical clustering.

5 Experimental Evaluation

We demonstrate the utility and efficiency of our approach using datasets crawled from the Website *eventful.com* for different topics from 2011 to 2012. Our framework is implemented in Java and runs with 24GB heap size. All experiments were run on an Intel Xeon 2.27GHz with 48GB RAM, running Ubuntu 64bit. Before presenting the results, we first describe the experimental setup.

5.1 Datasets and Experimental Setup

We crawled from the Website *eventful.com* for events in Germany and only festivals in Europe to easily validate the results later on. As raw data, each event consists of an event identifier, title, time, location, and a list of tags. Based on tags, one can select events for a particular topic, e.g., '*sports*' or '*music*'.

As mentioned in Section 4.1, the runtime complexity of our algorithm depends on not only the number of events but also the numbers of locations (more precisely, LTIs). Hence, for evaluation purpose, we select different datasets in various topics and sizes in terms of the number of events and locations. Table 1 shows five datasets used in our experiments, where the first two datasets (DE-Festival and DE-Sports) are smaller than the last three. All events took place in Germany ('DE-') or Europe ('EU-') in the years 2011 and 2012.

Table 1. Properties of datasets used in experiments

Dataset	**Topic**	**Area**	**Number of Events**			**Number of Locations**
			2011	**2012**	**Total**	
DE-Sports	sports	Germany	1,335	1,673	3,008	960
DE-Festival	festival	Germany	1,278	1,654	2,932	1,515
EU-Festival	festival	Europe	13,592	20,561	34,143	18,018
DE-Music	music	Germany	24,756	32,398	57,154	12,591
DE-All	all topics	Germany	72,672	85,995	158,667	20,141

First, the raw data of events are transformed into the form $\langle e_{id}, C, T, L \rangle$, where e_{id} is the event identifier, and the last three components are the following attributes: the event identifier, start-time, and venue identifier, respectively. Note that here the event identifiers are utilized for two purposes: to distinguish an event from others and to link event contexts to tags. We built a hierarchy for tags based on the method described in [9]. The event location is of granularity *Address* and can be generalized to *City* or *Place_Category*. The time component of an event in *Day* is generalized to *Day of the Week* (Mon, Tue, etc.), then *Businessday/Weekend* (BD/WE), and finally *All_Time*(AT).

With the above settings, we conducted a series of experiments to evaluate our framework. In the following section, we present the results obtained from extracting location annotations for the five datasets. We then demonstrate the utility of these annotations in location clustering in Section 5.3. Finally, we show the efficiency of *LTProfile-Miner* and *LTProfile-Updater* in Section 5.4.

5.2 Annotation Extraction

We run *LTProfile-Miner* to obtain LT-Profiles for the five datasets for the two years (2011-2012). Then, annotations for locations are obtained using the method described in Section 4.3. For each dataset, we tried different *npmi* thresholds (δ). Basically, the larger the threshold δ, the less locations are annotated, but the more confident the annotations are. For example, when $\delta = 0.1$, about 70-90% of the locations were annotated, whereas less than 30% of the locations were annotated when $\delta > 0.5$. Table 2 shows typical annotations we obtained. Note that the words describing topics are stemmed, and an item of a location annotation is followed by its *npmi* value, e.g., socc_WE:0.39.

Based on these annotations, one can easily find locations related to some given event topics. For example, *NürnbergMesse* (Germany) will be found when we search for places related to '*technology*' and '*exhibition*', as shown in Table 2. This can be explained by annual events related to computer software/hardware or electronic systems that are located there, such as '*embedded world*'.

From the extracted annotations, one can see that some annotations are obvious, for instance, the annotation of a cinema (e.g., *Kino Babylon Mitte*) contains event topics related to film and movie festivals; or an exhibition center (e.g., *Messe Essen*) contains event topics related to '*expo*', '*industry*', or '*tradeshow*'. We also found some interesting relationships, such as a relationship between the exhibition center *Messe Essen* and the topic '*fashion*'. This relationship is explained by a series of Modatex Fashion Fair events frequently occurring at that location. From the dataset EU-Festival, we also discovered some cities in Europe that are famous for their annual festivals . For instance, Torre del Lago, Peraso (Italy), and Montpelier (France) are famous for opera festivals.

5.3 Location Clustering

We exploit the extracted annotations to cluster locations. Such clusters will be utilized further to assign higher level semantic tags to locations or to build taxonomies of locations, as described in Section 4.4. For this purpose, we employ hierarchical clustering. The performance of location clustering is evaluated based on the F-score measure, commonly used in document clustering [8]. First, we describe how to obtain datasets with ground-truth for clustering evaluation.

Table 2. Example annotations extracted from the experimental datasets. Items in each annotation are sorted by their *npmi* values.

Location/Granularity	**Annotation**
DE-Sports	
Signal Iduna Park - Dortmund (Address)	{borussia_Sat:0.66, borussia_WE:0.61, borussia_AT:0.56, bundesliga_Sat:0.46, bundesliga_WE:0.42, football_WE:0.32, socc_WE:0.29,...}
Oschersleben Sachsen-Anhalt (City)	{circuitracing_AT:0.81, circuitracing_WE:0.77, motorsport_AT:0.71, autosport_AT:0.70, racing_AT:0.69, motorsport_WE:0.68,...}
DE-Festival	
Kino Babylon Mitte - Berlin (Address)	{filmfestival_BD:0.63, filmfestival_Thu:0.63, movi_BD:0.59, movi_Thu:0.59, film_BD:0.55, film_Thu:0.55, filmfestival_AT:0.54, movi_AT:0.50,...}
Messe Essen GmbH (Address)	{expo_Thu:0.66, fashion_Sat:0.66, convention_Thu:0.66, fashion_WE:0.64, homeexhibition_Fri:0.58, industry_AT:0.49, expo_BD:0.48,...}
DE-Music	
Bar/Night Club (Place category)	{elektronic_WE:0.55, hardstyl_WE:0.55, nightlif_WE:0.52, tranc_WE:0.45, rhythmnblu_BD:0.43, elektronic_AT:0.42, hardstyl_AT:0.42,...}
Concert Hall (Place category)	{philharmonieess_AT:0.67, doommetal_Tue:0.49, epic_Tue:0.49, jamsession_Mon:0.48, monstrosity_Wed:0.46, greatesthit_Mon:0.46,...}
DE-ALL	
Nürnberg Messe (Address)	{softwar_AT:0.62,expopromot_AT:0.53,school&alumni_AT:0.52, tool_AT:0.42, tradeshow_AT:0.41,scienc_AT:0.41, business_AT:0.41,...}
Philharmonie Berlin (Address)	{klassischekonzert_AT:0.64, cultur_AT:0.54, klassisch_AT:0.54, classical_AT:0.53, cultur_WE:0.49, symphony_WE:0.44, violin_Mon:0.42,...}
EU-Festival	
Torre del Lago - Tuscany - Italy (City)	{art&theatr_AT:0.87, art&theatr_BD:0.84, art&theatr_WE:0.73, opera_AT:0.27, opera_BD:0.26, opera_Fri:0.24, opera_WE:0.22,...}
LilianBaylisTheatre - London (Address)	{ballet_AT:0.80, ballet_BD:0.77, ballet_WE:0.69, clubbing_WE:0.47, nightlif_WE:0.47, danc_AT:0.40, theatr_Wed:0.32, art_Wed:0.28,...}

Since a location in our dataset can be generalized to a place category (e.g., '*Hotel*', '*Restaurant*'), we used such categories as ground-truth labels to evaluate location clustering, that is, locations of the same label are expected to be in the same cluster. We also removed locations with blank labels or non-categorical labels (e.g., '*postal code*' or '*named place*') because they produce meaningless results in that clustering evaluation method. Since the number of locations of different categories varies a lot (e.g., more than 100 for '*Concert Hall*', '*Bar/Club*', but less than 10 for '*Hospital*', '*Library*'), we finally selected only the top 7 categories (each category contains more than 10 locations) and generated datasets for clustering evaluation with a method as described below.

Let $L^k_{\{C_1,C_2,...,C_k\}}$ be a dataset consisting of locations of k categories C_1, C_2,..., C_k. Such a dataset is generated by choosing k categories from the top categories, e.g., $L^2_{\{Stadium,Theater\}}$. Let G_k be a group of generated datasets containing the same number of categories (i.e., k categories). For example, G_2 is a group of $\binom{7}{2} = 21$ datasets created by choosing 2 from the 7 categories, e.g., $L^2_{\{Stadium,Museum\}}$, $L^2_{\{Hotel,Museum\}}$. Instead of presenting the F-score for each individual dataset, we will show the mean F-score for each group G_k.

As mentioned in Section 3.2, an alternative to *npmi* is *tf-idf* that can be employed to weight event topics for location clustering. Here, we compare the performance of the *npmi* measure to the following versions of *tf-idf* that are widely used. Given an event topic f and an LTC Ω, two versions of *tf-idf*, called *tf-idf*$_1$ and *tf-idf*$_2$, are defined as *tf-idf*$_1(f,\Omega) = N_{f,\Omega} * \log(\frac{N}{N_f})$ and *tf-idf*$_2(f,\Omega) = (1+\log(N_{f,\Omega})) * \log(\frac{N}{N_f})$. The values N, N_f, and $N_{f,\Omega}$ are the number of LTCs, the number of LTCs that contain ET f, and the number of instances of the LTC

Ω that support f, respectively. Similar to the *npmi* measure, profiles of LTCs can be computed with Definition 2, where *npmi* is replaced by either *tf-idf*$_1$ or *tf-idf*$_2$. For a particular dataset and a particular measure, the threshold δ is selected so that the *F-score* is the largest. We use locations of the datasets DE-All and EU-Festival to assess the performance of location clustering since they covers all locations of the other datasets.

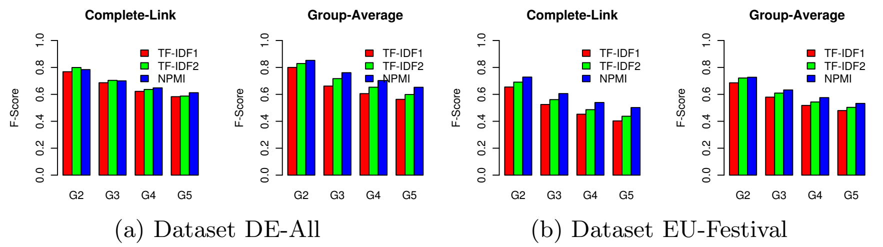


Fig. 3. Comparison of the measures *tf-idf*$_1$, *tf-idf*$_2$, and *npmi* in hierarchical clustering. The best result is achieved with the *npmi* measure.

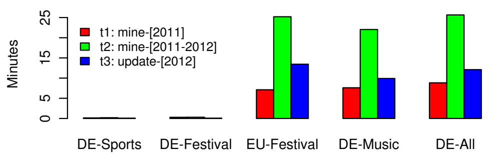


Fig. 4. Runtime of LTP-Miner and LTP-Updater

Figure 3 shows the comparison of the *npmi* measure with *tf-idf*$_1$ and *tf-idf*$_2$ in location clustering. Although different merging methods of hierarchical clustering were tried, due to space constraints, we present only results of two among the best methods: complete-link and group-average. In general, using the *npmi* measure gives the best result. A closer look at the generated profiles shows that a profile generated by the *npmi* measure contains more event topics presenting the characteristics of the corresponding location, as discussed in Section 3.2. In addition, one can see from Figure 3 that clustering on locations of the dataset DE-All gives better results than clustering on locations of the dataset EU-Festival. This is reasonable since it is more difficult to categorize locations of the latter dataset consisting of narrow topics, i.e., event topics related to '*festival*'.

We found many interesting clusters from the datasets. For example, we found clusters of bars/clubs regarding their music genres (e.g., jazz or r&b/soul); or a cluster of cities in Europe famous for opera festivals like Montpellier (France), Torre del Lago (Italy), or Pesaro (Italy).

5.4 Runtime and LTP-Updater Efficiency

We show the runtime of *LTProfile-Miner* for each dataset and also demonstrate the utility and efficiency of *LTProfile-Updater*. To do so, we split each dataset in Table 1 into two parts, each corresponding to one year. For example, from the dataset DE-Sports, we create two subdatasets DE-Sports$_{[2011]}$ and DE-Sports$_{[2012]}$, where the first one consists of events in 2011 and the latter one consists of events in 2012 of the dataset DE-Sports. From Table 1, one can see that the number of events in 2012 is larger than in 2011 (about 20 to 50%).

For each triple of datasets ($\mathcal{D}$, $\mathcal{D}_{[2011]}$ and $\mathcal{D}_{[2012]}$), we measure the runtime t_1 for *LTProfile-Miner* on $\mathcal{D}_{[2011]}$, the runtime t_2 for *LTProfile-Miner* on $\mathcal{D}$, and the runtime t_3 for *LTProfile-Updater* on $\mathcal{D}_{[2012]}$ (with support-data extracted from $\mathcal{D}_{[2011]}$). One can see the results in Figure 4. The first two datasets take only a few seconds to process. The cases of EU-Festival and DE-Music illustrate that the number of LTIs also affects the runtime. Although the number of events of the dataset EU-Festival is smaller than the dataset DE-Music, the number of locations of the dataset EU-Festival is larger, as shown in Table 1. In all cases, the runtime t_3 is larger than t_1, because the number of events in 2012 is larger than in 2011. However, in comparison to t_2, the runtime t_3 is much smaller for both datasets. This shows that using the *LTProfile-Updater* is an efficient and scalable approach to update the current location profiles with new data.

6 Conclusions and Ongoing Work

Event-based annotations of locations describe the event topics that are most related to a location, together with the time when the events of such topics most likely occur. We presented a comprehensive framework to extract such annotations from event datasets. Our approach is based on Location-Time-pair to associate a set of the most related event topics with a pair of a location and time. We also showed a scalable and efficient method to deal with periodic updates of event data. Our experimental results give a very good indication that the extracted annotations can be utilized well for semantic location search as well as clustering.

Using hierarchical clustering, taxonomies of locations can be built from annotated locations. We are currently developing a method to encode such taxonomies in RDF, and also to automatically link them to Linked Open Data sources. Another direction of current research focuses on exploiting negative *npmi* values in outlier detection. Such a method is important to detect errors or inaccurate information in event datasets.

References

1. Bouma, G.: Normalized (Pointwise) Mutual Information in Collocation Extraction. In: Proceedings of the Biennial GSCL Conference (2009)
2. Cao, X., Cong, G., Jensen, C.S.: Mining Significant Semantic Locations From GPS Data. Proceedings of the VLDB Endowment 3, 1009–1020 (2010)
3. Chakraborty, D., Spaccapietra, S., Parent, C.: SeMiTri: A Framework for Semantic Annotation of Heterogeneous Trajectories. In: EDBT, pp. 259–270 (2011)
4. Wang, C., Blei, D., Fei-Fei, L.: Simultaneous image classification and annotation. In: CVPR, pp. 1903–1910. IEEE (2009)

5. Derczynski, L.R.A., Yang, B., Jensen, C.S.: Towards context-aware search and analysis on social media data. In: EDBT, pp. 137–142. ACM Press (2013)
6. Hegde, V., Parreira, J.X., Hauswirth, M.: Semantic Tagging of Places Based on User Interest Profiles from Online Social Networks. In: Serdyukov, P., Braslavski, P., Kuznetsov, S.O., Kamps, J., Rüger, S., Agichtein, E., Segalovich, I., Yilmaz, E. (eds.) ECIR 2013. LNCS, vol. 7814, pp. 218–229. Springer, Heidelberg (2013)
7. Kulkarni, S., Singh, A., Ramakrishnan, G., Chakrabarti, S.: Collective annotation of Wikipedia entities in web text. In: KDD. ACM Press (2009)
8. Larsen, B., Aone, C.: Fast and effective text mining using linear-time document clustering. In: KDD, pp. 16–22. ACM Press (1999)
9. Le, A., Gertz, M.: Mining Spatio-temporal Patterns in the Presence of Concept Hierarchies. In: ICDM Workshops, pp. 765–772 (2012)
10. Pantel, P., Lin, D., Canada, A.T.H.: Discovering Word Senses from Text. In: KDD, pp. 613–619. ACM Press (2002)
11. Rattenbury, T., Good, N., Naaman, M.: Towards automatic extraction of event and place semantics from Flickr tags. In: SIGIR, pp. 103–110. ACM Press (2007)
12. Rattenbury, T., Naaman, M.: Methods for extracting place semantics from Flickr tags. ACM Transactions on the Web 3, 1–30 (2009)
13. Sengstock, C., Gertz, M.: Latent Geographic Feature Extraction from Social Media. In: SIGSPATIAL, pp. 149–158. ACM Press (2012)
14. Turney, P.D., Pantel, P.: From Frequency to Meaning: Vector Space Models of Semantics. Journal of Artificial Intelligence Research 37, 141–188 (2010)
15. Ye, M., Shou, D., Lee, W.-C., Yin, P., Janowicz, K.: On the semantic annotation of places in location-based social networks. In: KDD. ACM Press (2011)