Discovering Context to Prune Large and Complex Search Spaces

Arjun Satish[1](✉), Ramesh Jain[2], and Amarnath Gupta[3]

[1] Turn Inc., Redwood City, CA, USA
arjun.satish@turn.com
[2] University of California, Irvine, USA
[3] SDSC, University of California, San Diego, USA

Abstract. Specifying the search space is an important step in designing multimedia annotation systems. With the large amount of data available from sensors and web services, context-aware approaches for pruning search spaces are becoming increasingly common. In these approaches, the search space is limited by the contextual information obtained from a fixed set of sources. For example, a system for tagging faces in photos might rely on a static list of candidates obtained from the photo owner's Facebook profile. These contextual sources can get extremely large, which leads to lower accuracy in the annotation problem.

We present our novel **Context Discovery Algorithm**, a technique to progressively *discover* the most relevant search space from a dynamic set of context sources. This allows us to reap the benefits of context, while keeping the size of the search space within bounds.

As a concrete application for our approach, we present a simple photo management application, which tags faces of people in a user's personal photos. We empirically study the role of CueNet in the face tagging application to tag photos taken at real world events, such as conferences, weddings or social gatherings. Our results show that the availability of event context, and its dynamic discovery, can produce 80% smaller search spaces with nearly 100% correct tags.

1 Introduction

With the popularity of global social networks and proliferation of mobile phones, information about people, their social connections and day-to-day activities are becoming available at a very large scale. The web provides an open platform for documenting many real world events such as conferences, weather events and sports games. With such context sources, multimedia annotation algorithms [1–3] are being designed where the search space of tags is obtained from one or more sources (Fig. 1(b)). These approaches rely on a single *type* of context. For example, using social network information from Facebook to solve the face recognition problem. We refer to such a direct dependency between the search space and a data source as **static linking**. Although these systems are meritorious in their own right, they suffer from the following drawbacks: they do not employ

P. Brézillon et al. (Eds.): CONTEXT 2017, LNAI 10257, pp. 81–95, 2017.
DOI: 10.1007/978-3-319-57837-8_7

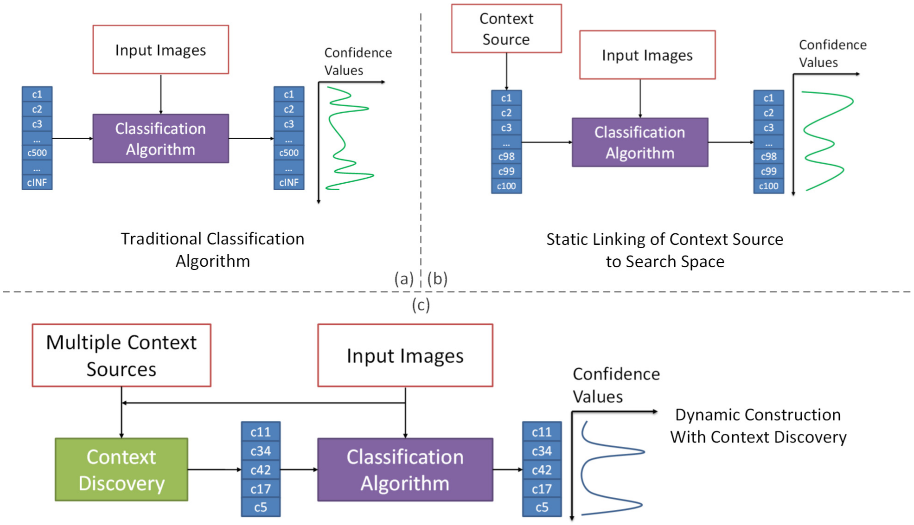

Fig. 1. The different approaches in search space construction for a multimedia annotation problem. A traditional setup (a), where the search space candidates are manually specified. Context is used (b) to generate large static search spaces. The CueNet framework (c), aims to produce small relevant search spaces.

multiple sources, and therefore the **relations** between them. By realizing that these sources are interconnected in their own way, we are able to treat the entire source topology as a network. Our intuition in this work is to navigate this network to progressively discover the search space for a given media annotation problem. Figure 1(c) shows how context discovery can provide substantially smaller search spaces for a set of images, which contain a large number of correct tags. A small search space with large number of true positives provides the ideal ground for an annotation algorithm to exhibit superior performance.

We present the CueNet framework, which provides access to multiple data sources containing event, social, and geographical information through a unified query interface to extract information from them. CueNet encapsulates our **Context Discovery Algorithm**, which utilizes the query interface to discover the most relevant search space for a media annotation problem. To facilitate a hands-on discussion, we show the use of context discovery in a real world application: face tagging in personal photos. As a case study, we will attempt to tag photos taken at conference events, weddings and social gatherings (birthday parties, for example) by different users. These photos could contain friends, colleagues, relatives or friends-of-friends or newly found acquaintances (who are not yet connected to the user through any social network). Real world event photos are particularly interesting because no single source can provide all the necessary information. It emphasizes the need to utilize multiple sources in a meaningful way (Fig. 2).

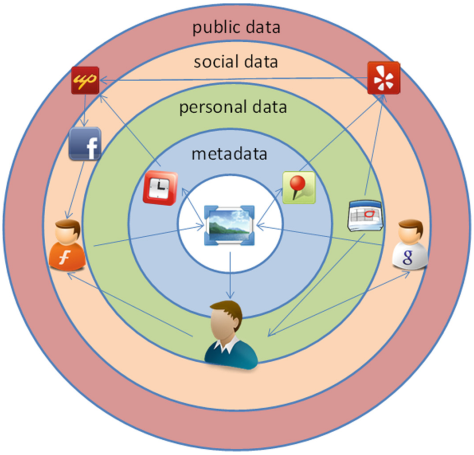

Fig. 2. Navigation of various data sources by the discovery algorithm.

Here is an **example** to illustrate CueNet's discovery process. Let's suppose that Joe takes a photo with a camera that records time and GPS in the photo's EXIF header. Additionally, Joe has two friends. One with whom he interacts on Google+, and the other using Facebook. The framework checks if either of them has any interesting event information pertaining to this time and location. We find that the friend on Google+ left a calendar entry describing an event (a title, time interval and name of the place). The entry also marks Joe as a participant. In order to determine the category of the place, the framework uses Yelp.com with the name and GPS location to find whether it is a restaurant, sports stadium or an apartment complex. If the location of the event was a sports stadium, it navigates to upcoming.com to check what event was occurring here at this time. If a football game or a music concert was taking place at the stadium, we look at Facebook to see if the friend "Likes" the sports team or music band. By traversing the different data sources in this fashion, the number of people, who could potentially appear in Joe's photograph, was incrementally built up, rather than simply reverting to everyone on his social network or people who could be in the area where the photograph was taken. We refer to such navigation between different data sources to identify relevant contextual information as **progressive discovery**. The salient feature of CueNet is to be able to progressively discover events, and their associated properties, from the different data sources and relate them to the photo capture event. We argue that given this structure and relations between the various events, CueNet can make assertions about the presence of a person in the photograph. Once candidates have been identified by CueNet, they are passed to the face tagging algorithm ([4], for example), which can perform very well as their search space is limited to two candidates.

Contributions: Real-world search spaces are large and complex (owing to their time varying relationships). Our contribution in this paper is a technique to discover the search spaces for multimedia annotation problems by using contextual information (events and their interrelations) from multiple data sources. We claim that this search space is significantly smaller than one obtained by static linking approaches, but retains a high number of true positives. We describe our findings when these ideas are applied to a personal photo annotation problem.

In the following sections, we develop our notion of context, and identify its properties which make discoveries like the above possible. We discuss the CueNet framework, its different components, and the conditions it creates which allow for progressive discovery. We present a context discovery algorithm to use these properties and tag faces in personal photos. Finally, we present an empirical evaluation to support our above claims.

1.1 Related Work

Our work has been deeply informed by the modeling strategy used by Karen Henricksen [10]. In the real world, relationships between objects change continuously. A search space where relationships change frequently is a **complex search space**. Our context discovery is built to handle such situations. In terms of tagging photos using context, Naaman et al. have exploited GPS attributes to extract place and person information [1]. Rattenbury and Naaman [5] devised techniques to find tags which describe events or places by analyzing their spatiotemporal usage patterns. Time alone is used for organizing photos by Graham et al. [6]. Context information and image features are used in conjunction by O'Hare and Smeaton in [2] and Cao et al. in [7] to identify event tags. The biggest difference in our work is the ability of our discovery algorithm to add personal, geographical or event tags to a given photo.

2 Context

Our justification for the use of context begins with the statement: *For a given user, the correctness of face tags for a photograph containing people she has never met is undefined.* This observation prepares us to understand what context is, and how contextual reasoning assists in tagging photos. The description of any problem domain requires a set of abstract data types, and a model of how these types are related to each other. We **define** contextual types as those which are semantically different from these data types, but can be directly or indirectly related to them via an extended model which encapsulates the original one. Contextual reasoning assists in the following two ways. **First**, contextual data restricts the number of people who might appear in the photographs. We can also argue that all the personal data of a user (her profile on Facebook, LinkedIn, email exchanges, phone call logs) provides a reasonable estimate of all these people who might appear in her photos. **Second**, by reasoning on abstractions in the contextual domain, we can infer conclusions on the original problem. We exploit this property to develop our algorithm in the later sections. Though CueNet can be applied to a variety of recognition problems, we focus on tagging people in personal photos for concreteness, where, the image and person tag form the abstractions in the problem domain. The types used in the contextual domain, but not limited to, are **Events:** includes description of events like conferences, and their structure (for example, what kind of sessions, talks and keynotes occur within the conference); **Social Relationships:** information about a user's social graph and **Geographical Proximity:** various tools like Facebook Places, Google Latitude or Foursquare provide information about where people are at a given time.

The above classes of contextual data can be obtained from a variety of data sources. Examples of data sources range from mobile phone call logs and email conversations to Facebook messages to a listing of public events at upcoming.com. Sources can be classified into **Personal Data Sources:** include all sources which provide details about the particular user whose photo is to be tagged (for example, email and Google Calendar); **Social Data Sources:** sources such as Facebook, DBLP or LinkedIn which provide contextual information about a user's friends and colleagues, or **Public Data Sources** which provide information about public events.

Social and public data sources are enormous in size, containing information about billions of events and entities. Trying to use them directly will lead to scalability problems faced by face recognition and verification techniques. But, by using personal data, we can discover which parts of social and public sources are more relevant. For example, if a photo was taken at San Francisco, CA (where the user lives) his family in China is less relevant. Thus, the role of personal information is twofold. **Firstly**, it provides contextual information regarding the photo. **Secondly**, it acts as a bridge to connect to social and public data sources to discover interesting people connected to the user who might be present in the event and therefore, the photo.

We must note the **temporal relevance** property of a data source. Given a stream of photos taken during a time interval, the source which contributed interesting context for a photo might not be equally useful for the one appearing next. This is because sources tend to focus on a specific set of event types or relationship types, and the two photos might be captured in different events or contains persons with whom the user maintains relations through different sources. For example, two photos taken at a conference might contain a user's friends in the first, but with advisers of these friends in the next. The friends might interact with the user through a social network, but their advisers might not. By using a source like DBLP, the relations between the adviser and friends can be discovered. We say that the temporal relevance of these context sources is ***low***. This requirement will play an important role in the design of our framework.

3 The CueNet Framework

Figure 3 shows the different components of the CueNet framework. The Ontological **Event Models** specify various event and entity classes, and the different relations between them. These declared types are used to define the **Data Sources** which provides access to different types of contextual data. The **Person Verification Tools** consist of a database of people, their profile information and photos containing these people. When this module is presented with

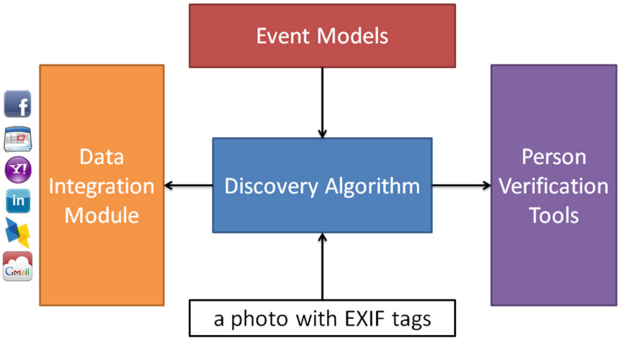

Fig. 3. The conceptual architecture of CueNet.

a candidate and the input photograph, it compares the features extracted from the candidate's photos and the input photo to find the confidence threshold. In this section, we describe each module, and how the context discovery algorithm utilizes them to accomplish its task.

3.1 Event Model

Our ontologies extend the E* model [8] to specify relationships between events and entities. Specifically, we utilize the relationships "**subevent-of**", which specifies event containment. An event $e1$ is a subevent-of of another event $e2$, if $e1$ occurs completely within the spatiotemporal bounds of $e2$. Additionally, we utilize the relations **occurs-during** and **occurs-at**, which specify the space and time properties of an event. Also, another important relation between entities and events is the "**participant**" property, which allows us to describe which entity is participating in which event. It must be noted that participants of a subevent are also participants of the parent event. A participation relationship between an event and person instance asserts the presence of the person within the spatiotemporal region of the event. We argue that the reverse is also true, i.e., if a participant P is present in $\mathcal{L}_P$ during the time $\mathcal{T}_P$ and an event E occurs within the spatiotemporal region ($\mathcal{L}_E$, $\mathcal{T}_E$), we say P is a participant of E if the event's spatiotemporal span contained that of the participant.

$$\texttt{participant}(E, P) \iff (\mathcal{L}_P \sqsubset_L \mathcal{L}_E) \wedge (\mathcal{T}_P \sqsubset_T \mathcal{T}_E) \qquad (1)$$

The symbols $\sqsubset_L$ and $\sqsubset_T$ indicate spatial and temporal containment respectively [8]. In later sections, we refer to the location and time of the event, $\mathcal{L}_E$ and $\mathcal{T}_E$ as E.**occurs-at** and E.**occurs-during** respectively.

3.2 Data Sources

The ontology makes available a vocabulary of classes and properties. Using this vocabulary, we can now declaratively specify the schema of each source. With these schema descriptions, CueNet can infer what data source can provide what type of data instances. For example, the framework can distinguish between a source which describes conferences and another which is a social network. We use a LISP like syntax to allow developers of the system to specify these declarations. The example below describes a source containing conference information.

```
(:source conferences
(:attributes name time title)
(:relation conf type-of conference)
(:relation t type-of time-interval)
(:relation attendee type-of person)
(:relation attendee participant-in conf)
(:relation conf occurs-during t)
(:mappings [ [t time] [conf.title title] [attendee.name name] ] )
```

The above source declaration consists of a s-expression, where the source keyword indicates a unique name for the source. The `attributes` keyword is used to list the attributes of this source. The `relation` keyword constructs the instances conf, time, loc, attendee which are of conference, time-interval, location and person class types respectively, and relates them with relations specified in the ontology. Finally, the `mapping`s are used to map nodes in the relationship graph constructed above to attributes of the data source. For example, the first mapping (specified using the map keyword) maps the conference's time-interval object (t) to the (time) attribute of the source.

3.3 Conditions for Discovery

CueNet is entirely based on reasoning in the event and entity (i.e., person) domain, and the relationships between them. These relationships include participation (event-entity relation), social relations (entity-entity relation) and subevent relation (event-event). For the sake of simplicity, we restrict our discussions to events whose spatiotemporal spans either completely overlap or do not intersect at all. We do not consider events which partially overlap. In order to develop the necessary conditions for context discovery, we consider the following two axioms:

Entity Existence Axiom: Entities can be present in one place at a time only. The entity cannot exist outside a spatiotemporal boundary containing it.

Participation Semantics Axiom: If an entity is participating in two events at the same time, then one is the subevent of the other.

Given, the ontology O, we can construct event instance graph $G^I(V^I, E^I)$, whose nodes are instances of classes in C^O and edges are instances of the properties in P^O. The context discovery algorithm relies on the notion that given an instance graph, *queries* to the different sources can be automatically constructed. A query is a set of predicates, with one or more unknown variables. For the instance graph $G^I(V^I, E^I)$, we construct a query $Q(D, U)$ where D is a set of predicates, and U is a set of unknown variables.

Query Construction Condition: Given an instance graph $G^I(V^I, E^I)$ and ontology $O(C^O, P^O)$, a query $Q(D, U)$ can be constructed, such that D is a set of predicates which represent a subset of relationships specified in G^I. In other words, D is a subgraph induced by G^I. U is a class, which has a relationship $r \in P^O$, with a node $n \in D$. Essentially, the ontology must prescribe a relation between some node n through the relationship r. In our case, the relation r will be either a **participant** or **subevent** relation. If the relationship with the instances does not violate any object property assertions specified in the ontology, we can create the query $Q(D, U)$.

Identity Condition: Given an instance graph $G^I(V^I, E^I)$, and a result graph $G^R(V^R, E^R)$ obtained from querying a source, we can merge two events only if they are identical. Two nodes $v_i^I \in V^I$ and $v_r^R \in V^R$ are identical if they meet the following two conditions. **(i)** Both v_i^I and v_r^R are of the same class type, and **(ii)** Both v_i^I and v_r^R have exactly overlapping spatiotemporal spans, indicated by the $=_L$ and $=_T$. Mathematically, we write:

$$\begin{aligned} v_i^I = v_r^R \iff (v_i^I.\textbf{type-of} = v_r^R.\textbf{type-of})\wedge \\ (v_i^I.\textbf{occurs-at} =_L v_r^R.\textbf{occurs-at})\wedge \\ (v_i^I.\textbf{occurs-during} =_T v_r^R.\textbf{occurs-during}) \end{aligned} \tag{2}$$

Subevent Condition: Given an instance graph $G^I(V^I, E^I)$, and a result graph $G^R(V^R, E^R)$ obtained from querying a source, we can construct a subevent edge between two nodes $v_i^I \in V^I$ and $v_r^R \in V^R$, if one is spatiotemporally contained within the other, and has at least one common `Endurant`.

$$\begin{gathered} v_i^I \sqsubset_L v_r^R, \\ v_i^I \sqsubset_T v_r^R \end{gathered} \tag{3}$$

$$v_i^I.\textbf{Endurants} \cap v_r^R.\textbf{Endurants} \neq \{\phi\} \tag{4}$$

Here v_i^I.**Endurants** is defined as a set $\{w | w \in V_i^I \wedge w.\text{type-of} = \text{Endurant}\}$. If Eq. (4) does not hold, we say that v_i^I and v_r^R co-occur.

Merging Event Graphs: Given the above conditions, we can now describe an important building block for the context discovery algorithm: the steps needed to merge two event graphs. An example for this is shown in Fig. 4(b–d). Given the event graph consisting of the photo capture event on the left of (b) and a meeting event m and conference event c, containing their respective participants. In this example, the meeting event graph, m is semantically equivalent to the original graph. But the conference event, c is telling that the person AG is also participating in a conference at the time the photo was taken. The result of merging is shown in (d). An event graph merge consists of two steps. The first is a `subevent hierarchy join`, and the second is a `prune-up` step.

Given an original graph, O_m, and a new graph N_m, the join function works as follows: All nodes in N_m are checked against all nodes in O_m to find identical counterparts. For entities, the identity is verified through an identifier, and for

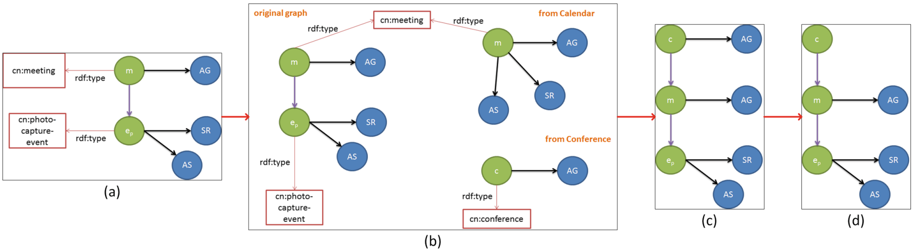

Fig. 4. The various stages in an iteration of algorithm in Sect. 3.4. (a) shows an example event graph describing a photo taken at a meeting. The meeting consists of three participants AG, SR and AS. The photo contains SR and AS. (b) shows two events returned from the data sources. One is a meeting event which is semantically identical to the input. The other is a conference event with AG. (c) shows the result of merging these graphs. (d) The `prune-up` function removes the duplicate reference to AG.

events, Eq. (2) is used. Because of the entity existence and participation semantics axioms, all events which contain a common participant are connected to their respective super event using the subevent relation (Eqs. (3) and (4) must be satisfied by the events). Also, if two events have no common participant, then they can be still be related with the subevent edge, if the event model says it is possible. For example, if in a conference event model, keynotes, lunches and banquets are declared as known subevents of an event. Then every keynote event, or banquet event to be merged into an event graph is made a subevent of the conference event, if the Eq. (3) holds between the respective events. It must be noted that node AG occurs twice in graph (c). In order to correct this, we use the participation semantics axiom. We traverse the final event graph from the leaves to the root events, and remove every person node if it appears in a subevent. This is the `prune-up` step. Using these formalisms, we now look at the working of the context discovery algorithm.

3.4 Context Discovery Algorithm

The input to the algorithm is a photo (with EXIF tags) and an associated owner (the user). By seeding the graph with owner information, we bias the discovery towards his/her personal information. An event instance graph is created where each photo is modeled as a photo capture event. Each event and entity is a node in the instance graph. Each event is associated with time and space attributes. All relationships are edges in this graph. All EXIF tags are literals, related to the photo with data property edges. Figure 4 graphically shows the main stages in a single iteration of the algorithm.

The event graph is traversed to produce a queue of entity and event nodes, which we shall refer to as DQ (discovery queue). The algorithm consists of two primary functions: **discover** and **merge**. The discover function is tail recursive, invoking itself until a termination condition is reached (when at most k tags are obtained for all faces or no new data is obtained from all data sources for all generated queries). The behavior of the query function depends on the type of the node. If the node is an event instance, the function consults the ontology to find any known sub-events, and queries data sources to find all these subevents, its properties and participants of the input event node. On the other hand, if it is an entity instance, the function issues a query to find all the events it is participating in. Results from data source wrappers are returned in the form of event graphs. These event graphs are merged into the original event graph by taking the following steps. First, it identifies **duplicate** events using the conditions mentioned above. Second, it identifies subevent hierarchies using the graph merge conditions described above, and performs a **subevent hierarchy join**. Third, the function **prune-up** removes entities from an event when its subevent also lists it as a participant node. Fourth, **push-down** is the face verification step if the number of entities in the parents of the photo-capture events is small (less than T). Push down will try to verify if any of the newly discovered entities are present in the photo and if they are (if the tagging confidence is higher than the given threshold), the entities are removed from the super event, and linked to the photo capture event as its participant. On the other hand, if this number is larger than T, the algorithm initiates the **vote-and-verify** method, which ranks all the candidates based on social relationships with people already identified in the photo. For example, if a candidate is related to two persons present in the photo through some social networks, then its score is 2. Ranking is done by simply sorting the candidate list by descending order of score. The face verification runs only on the top ranked T candidates. If there are still untagged faces after the termination of the algorithm, we vote over all the remaining people, and return the ranked list for each untagged face.

4 Experiments

In this section, we analyze how CueNet drives a real world face tagging application. The application contains a set of photos, and a database of people, and its goal is to associate the right persons for each photo, with high accuracy. The goal of CueNet, and the focus of our analysis, is to provide small search spaces so that the application can exhibit high accuracy in all datasets. In the following evaluation, we investigate three questions. **First**, what sources provide the most interesting context? **Second**, how small are the candidates lists constructed by the discovery algorithm, which are provided to the classification algorithm as a “pruned” version of the search space? And **third**, what percentage of true positives does this pruned search space contain?

4.1 Setup

We collected 2000 photos taken at 17 different real-world events by 6 different people in our face tagging experiment. The total number of candidates obtained from all the social, personal and public sources in our experiment is 7736. Each photo contains one or more persons from this database. The owner of the photos was asked to provide access to their professional Google Calendar to access personal events. Information from social networks was gathered. Specifically, events, social graph, photos of user and their friends from Facebook. In order to obtain information of the conference event, we used the Stanford NER [9] to extract names of people from the conference web pages. Descriptions of the keynote, session and banquet events were manually entered into the database. Our sources also included personal emails, access to public events website upcoming.com (Yahoo! Upcoming) and used Yahoo! PlaceFinder for geocoding addresses. The ground truth was annotated by the user with our annotation interface. For each photo, this essentially consisted of the ID of the persons in it. We will denote each dataset as 'Di' (where $1 \leq i \leq 17$ for each dataset). Table 1 describes each dataset in terms of number of photos, unique annotations in ground truth and the year they were captured. The total number of unique people who could have appeared in any photo in our experiments is 7736.

Table 1. Profile of datasets used in the experiments.

Dataset	Unique people	No. of photos	Year
D1	43	78	2012
D2	24	108	2012
D3	6	16	2010
D4	7	10	2010
D5	36	80	2009
D6	18	65	2013
D7	7	11	2013

We divide the sources into different categories to facilitate a more general discussion. The categories are "Personal Information" (same as Owner Information in Sect. 3.4), "Event sources", and "Social Networks". Event sources include Facebook events, Yahoo Upcoming web service, our conference events database among other sources. Social networks include Facebook's social graph. Personal information contained information about the user, and a link to their personal calendars. An annotation is considered "Out of Context Network" if it is not in any of these sources.

Figure 5 shows the distribution of the ground truth annotations across various sources, for each conference dataset. For example, the bar corresponding to D2 says that 87.5% of ground truth annotations were found in event sources, 41.67% in social networks, 4.17% in personal information and 12.5% were not found in any source, and therefore marked as "Out of Context Network". From this graph it is clear that event sources contain a large portion of ground truth annotations. Besides D4, a minimum of 70% of our annotations are found in event sources for all datasets, and for some datasets (D3, D7) all annotations are found in event sources. The sum total of contributions will add up to values more than 100% because they share some annotations among each other. For example, a friend on Facebook might show up at a conference to give the keynote talk.

Context Discovery: Now, lets look at the reduction obtained in state space with the discovery algorithm. The total number of people in our experiment

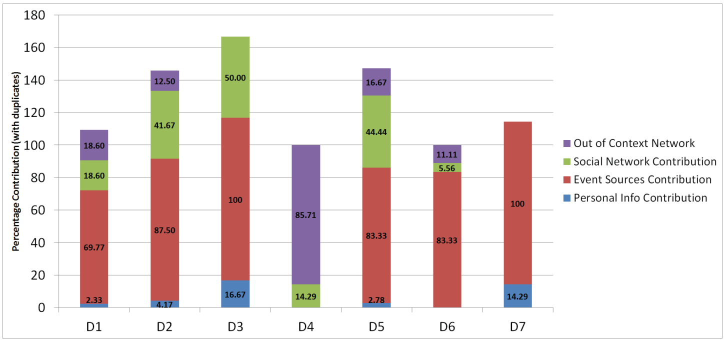

Fig. 5. The distribution of annotations in the ground truth for the conference data sets across various sources.

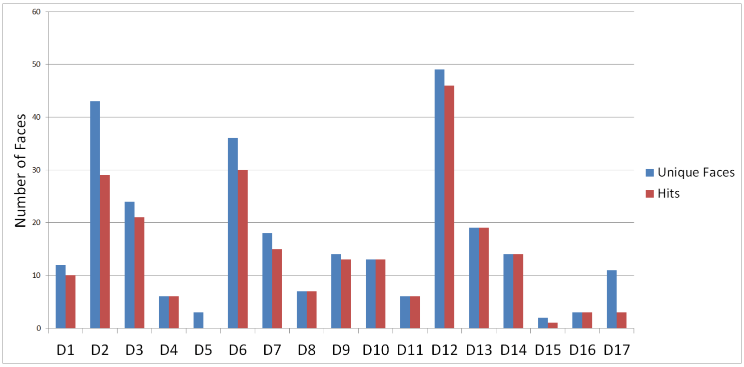

Fig. 6. Hit counts for all datasets using context discovery algorithm.

universe is 660. By statically linking the sources, we would expect the search space to contain 660 candidates for tagging any of the datasets. However, the context discovery algorithm reduced the size of the search space as shown in Fig. 6. The search space varies from 7 people in D7 (1%) to 338 people in D2 (51%). We denote the term hit rate as the percentage of true positives in the search space. Even if our search space is small, it might contain no annotations from the ground truth, leading to poor classifier performance. The hit rates are also summarized in Fig. 6. For D4, the algorithm found no event sources (as seen in Fig. 5), and therefore constructed a search space which was too small, thereby containing none of the ground truth. With the exception for D4, the hit rate is always above 83%. **We observe an overall reduction in the search space size, with a high hit rate for majority of the datasets**.

We now investigate the role of different context sources in the discovery algorithm. If an entity in the search space was merged into the event graph by an event source, they are said to be "contributed" from it. Figure 8 shows the contribution from various sources for all datasets. For example, D1 obtained 69.77% of true positives in its search space from event sources, 2.33% from

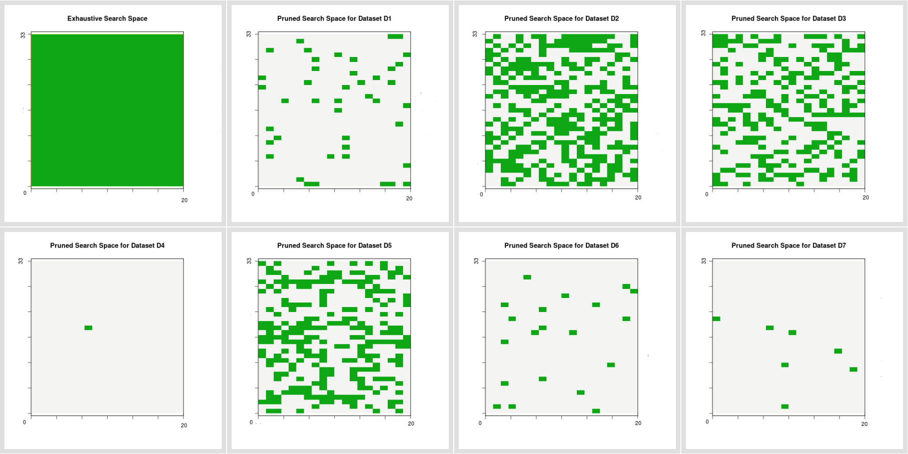

Fig. 7. Grid plots showing the exhaustive search space and pruning in search space for different datasets. (Color figure online)

personal information and 11.63% from social networks. 16.28% of true positives for D1 were obtained from no source, and were therefore marked as “Out of Context Network”. This graph brings to light our argument that most of the true positives, for all datasets, were obtained as a result of navigating the event sources. It will also be noted that the role of social networks is minimal. It was found useful for only one dataset. Relying on social sources alone would have led to a large number of false positives in the classifier performance. Even though the direct impact of personal information is negligible, it is critical in linking in photos to the owner, and from there to different events. Without the availability of personal information, the algorithm would not have reached the context rich event sources.

Finally, we compare the various search spaces constructed by discovery algorithm. We represent all people in our experiment universe in a color grid (with 33 × 20 cells for 660 people). Each cell represents the presence or absence of a person in the search space. If a person was present in the candidate list provided to the tagging algorithm, we color the corresponding cell green, otherwise it is colored white. Figure 7 shows the color grids describing search spaces for all datasets, and an exhaustive search space (top-right grid). The positioning of people along the grid is arbitrary, but consistent across grids. Our aim here is to see the diversity in search spaces created by the algorithm. It can be seen that CueNet prunes the search space very differently for different datasets. As we move from dataset to dataset, the data sources present different items of information, and therefore CueNet constructs very search spaces. Dataset D2, D4 and D5 are very large conferences hosting hundreds of people in the same field. This explains why a large portion of the grid is covered. Also, this was the same conference held in three different years, and therefore, had a lot of common attendees resulting in overlap.

4.2 Conclusion

These experiments show that we need to dynamically link sources to extract context out of them. A source which was relevant for one set of photographs was not very effective for another. We also see that event sources are very effective in pruning very large number of candidates yet retaining a very large number of true positives.

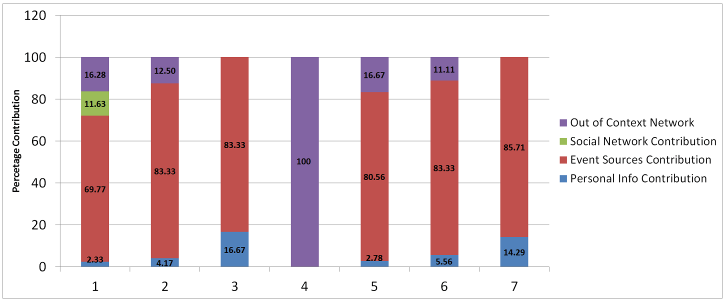

Fig. 8. Graph showing the contribution of each source type in context discovery.

5 Summary

We presented a innovative context based technique to prune complex search spaces for the entity identification problem in multimedia objects, specifically personal photos. We model context as a time varying relationship of entities and events to architect a novel context discovery framework CueNet, and designed its discovery algorithms to progressively query various heterogeneous data sources and merge context relevant to a given photo. We empirically analyzed the ability of context discovery to remove a large number of irrelevant candidates.

References

1. Naaman, M., Yeh, R.B., Paepcke, A., Garcia-Molina, H.: Leveraging context to resolve identity in photo albums. In: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (2005)
2. O'Hare, N., Smeaton, A.F.: Context-aware person identification in personal photo collections. IEEE Trans. Multimed. **11**(2), 220–228 (2009)
3. Stone, Z., Zickler, T., Darrell, T.: Autotagging facebook: social network context improves photo annotation. In: Computer Vision and Pattern Recognition (2008)
4. Kumar, N., Berg, A., Belhumeur, P.N., Nayar, S.: Describable visual attributes for face verification and image search. IEEE Trans. Pattern Anal. Mach. Intell. **33**, 1962–1977 (2011)
5. Rattenbury, T., Naaman, M.: Methods for extracting place semantics from Flickr tags. ACM Trans. Web (TWEB) **3**, 1 (2009)

6. Graham, A., Garcia-Molina, H., Paepcke, A., Winograd, T.: Time as essence for photo browsing through personal digital libraries. In: Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries (2002)
7. Cao, L., Luo, J., Kautz, H., Huang, T.S.: Annotating collections of photos using hierarchical event and scene models. In: Computer Vision and Pattern Recognition (2008)
8. Gupta, A., Jain, R.M.: Managing Event Information: Modeling, Retrieval, and Applications. Morgan & Claypool Publishers, San Rafael (2011)
9. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by Gibbs sampling. In: The 43rd Annual Meeting on Association for Computational Linguistics (2005)
10. Henricksen, K., Indulska, J., Rakotonirainy, A.: Modeling context information in pervasive computing systems. In: Mattern, F., Naghshineh, M. (eds.) Pervasive 2002. LNCS, vol. 2414, pp. 167–180. Springer, Heidelberg (2002). doi:10.1007/3-540-45866-2_14