UNIVERSITY OF CALIFORNIA,
IRVINE

Pruning Large Search Spaces using Context Networks

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Arjun Satish

Dissertation Committee:
Professor Ramesh Jain, Chair
Professor Nalini Venkatasubramanian
Professor Deva Ramanan
Professor Bill Tomlinson
Dr. Amarnath Gupta

2013

DEDICATION

(Optional dedication page)
To ...

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

CURRICULUM VITAE

Arjun Satish

EDUCATION

Doctor of Philosophy in Computer Science **2012**
University name *City, State*

Bachelor of Science in Computational Sciences **2007**
Another university name *City, State*

RESEARCH EXPERIENCE

Graduate Research Assistant **2007–2012**
University of California, Irvine *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant **2009–2010**
University name *City, State*

REFEREED JOURNAL PUBLICATIONS

Ground-breaking article **2012**
Journal name

REFEREED CONFERENCE PUBLICATIONS

Awesome paper **Jun 2011**
Conference name

Another awesome paper **Aug 2012**
Conference name

SOFTWARE

CueNet `https://github.com/wicknicks/cuenet/`
Polyglot implementation of the CueNet ecosystem to tag faces in photos.

ABSTRACT OF THE DISSERTATION

Pruning Large Search Spaces using Context Networks

By

Arjun Satish

Doctor of Philosophy in Computer Science

University of California, Irvine, 2013

Professor Ramesh Jain, Chair

The search spaces of real world AI problems are extremely large. The strategy adopted to prune large search spaces is to accurately model the environment, and reason which parts of the search space can be pruned without hurting the performance of the decision making algorithm. Although relatively easier in virtual environments, modeling dynamic real world environments is a very challenging problem.

Consider the example of tagging faces in a person's photo album. The search space contains a few billion potential candidates. Any algorithm which attempts to directly tag one of billion people in a given photo will perform poorly. But, given a complete model of the world, the search space needs to be limited only to the entities who were present close to the camera's field of view at the time of photo capture. Now, the algorithm needs to decide over few tens of candidates as opposed to billions.

In this dissertation, we present *Context Networks*, a representation of real world environments used to prune search spaces for real world AI problems, and a novel *Progressive Discovery* algorithm to construct such networks from heterogenous data sources. We facilitate our following discussions through an example system to tag faces in personal photos. We discuss the architecture of a complete system to model data sources, construct context networks for a given AI problem (which in the case of face tagging would be a personal photo and the

exhaustive search space) and provides a pruned version of the search space most relevant to the problem. We also present experiments to quantitatively demonstrate the efficacy of our algorithm.

Chapter 1

Introduction

Artificial Intelligence is defined as the study and design of intelligent systems. An intelligent system is one which perceives its environment and takes actions that maximizes its chances of success. The nature of this environment varies from system to system. For example, the environment for a chess playing system can be enumerated with a set of rules; the environment for an autonomous car is the size, position of cars, their relative speeds on the street and the position of important objects like traffic lights, stop signs. The search space of an AI problem is the total number of candidate objects over which a decision needs to made. In both the above examples, the search space contains an enormous of candidates. But for the second case, the problem of enumerating the search space is non-trivial. Firstly, It requires a large amount of knowledge about the environment. Also, the interactions between elements in the real world are very hard to predict. They need to be *sensed* in a real time fashion for decisions to be made accordingly. With the growing amount of sensors in the real world, its becoming possible to record events and activities of increasingly finer granularity. Sensory inputs in the real world could range from news articles on the web to stock market tickers to thermometer readings from wildlife parks. Such large amounts of heterogeneous information cannot be easily accessed and analyzed to prune search spaces.

Figure 1.1: Such streets would pose a significant challenge in designing automomous cars.

This dissertation addresses the problem of constructing computational representations of real world environments from various heterogeneous data sources, to reason which parts of the search space can be pruned without hurting the overall performance of the intelligent system. We refer to such a representation as the **Context Network** of the environment. The network describes real-world events occurring in the environment, the entities participating in them, and their semantic inter-relationships.

1.1 Approach

Before embarking on a mission to model the entire world, we ask ourselves the question: How much of the real world information is actually relevant to the intelligent system? Constructing the entire world model is extremely challenging and often unnecessary. For example, there might not be much value in representing sports events in New York in a model being used to help cars navigate in Japan.

Figure 1.2: Face tagging problems could be challenged by very large search spaces.

This primary contribution of this dissertation is a ***progressive discovery*** algorithm to ingest information from various real world data sources to construct context networks containing the most relevant information for pruning the search space for the system. Examples of data sources include social media web services to provide information about events and entities like Facebook, Twitter; services which can be queried to find information about places like Yelp; Sensors on personal mobile phones, for example GPS which inform applications of the location of a person is present at any given point in time.

What is progressive discovery? Progressive discovery is an incremental process where knowledge of real world events and entities can be added to a given context network. Given a context network and multiple data sources describing events and entities, a progressive discovery algorithm will obtain new information from the sources and relate it to context network. By iteratively executing this algorithm, we can grow a context network until the data sources can provide no further information or the information in the network prunes the search space well enough for the AI problem to be fully solved.

The discussion in the following chapters on context networks and their discovery from various data sources will be presented in conjunction with an application to **tag faces in personal photos**. The face tagging algorithm, whose search space contains a few million entities is a very hard real world AI problem. But if a real world model of the world existed, the search space which is relevant to this photo contains just the entities who are present within the field of view of the camera at the time the photo was captured.

1.2 Why Prune Search Spaces?

Search space pruning has been identified as an important problem since the earliest discussions on Computer Science topic. So what has changed in the last few years, that we should focus on it again? The following reasons describe, in a nutshell, the differences between this problem and previously tackled problems and why it is relevant today.

First, large amounts of personal and local information are now available. Personal information paves the way to reason about problems from an individual's point of view, which provides unique opportunities in comparison with using a generic model of the user. Local information, such as social, spatial and environmental data is also becoming increasingly available. Along with ever-increasing public databases, new ways of modeling real world data [22] and knowledge [33] they provide new opportunities to combine information, thereby creating new research challenges.

Second, The nature of applications today is different. Many real world applications can benefit with general principles to prune real world search spaces. For example, autonomous cars navigating streets as shown in 1.1, personal photo tagging applications tasked with photos similar to 1.2. Real world models can assist in modeling real time data from Twitter-like application streams to understand public reaction to events. Banks can utilize information about the user's activities to gain insights to how to detect fraudulent transactions, massively large models can be used to understand climate behavior and its relations with human activities.

1.3 Overview

This dissertation is organized into the following chapters. Chapter 2 provides an overview of context, how context has been used to address problems in various scientific disciplines and how we use context in our specific personal photo tagging application. Chapter 3 describes the related work in computer science, and how this work is informed by them. Chapter 4 describes our context discovery framework, how it models various data sources, and how our progressive discovery algorithm constructs models for real world problems. We facilitate this discussion with an example real world application to tag faces of people in personal photos. Chapter 5 analyzes the algorithmic complexity of different parts of the system, and provides experiments to verify the competence and performance of the system. We also present experiments to confirm the efficacy of our approach in the light of the real world application. Finally, chapter 6 attempts to describe the future possibilities of using context discovery in computer science.

1.4 Terminology

Before starting the discussion on Context Networks, it is necessary to include a short note on terminology to avoid any ambiguities. We use the word 'Object' to collectively refer to events and entities. An entity includes persons, places in the world, for example 'Starbucks, UC Irvine', 'The Eiffel Tower, Paris, France', or organizations, for example 'Google Inc', 'Royal Society of London'. The term 'object' has been used in literature to refer to things which have no temporal properties. But, in our discussion, an ‘object’ could imply an event which exhibits temporal properties.

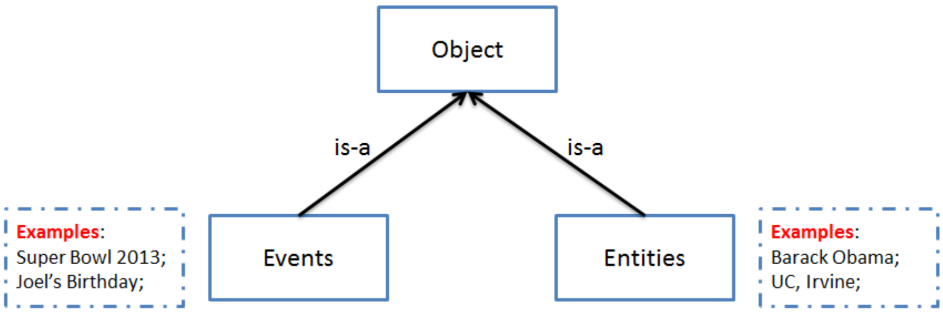

Figure 1.3: Objects, Events and Entities.

Chapter 2

What is Context?

This chapter describes what we mean by 'Context' and 'Context-Awareness'. We look at the previous definitions and usages of these terms, and propose why these are lacking for the purposes of this dissertation. As we go through each definition, we will pick out the most relevant parts to form the ground for the context framework introduced in this and developed in later chapters.

2.1 Previous Definitons

The earliest study on context was done by Bill Schilit et al. in [43]. The focus in this study was how to build software in dynamic environments. The dynamics of the environments were largely due to people requiring different computational services at the different times, the modality of request (through a mobile device or through a workstation), and the environment of the device (are there cameras and projectors nearby if the task requires video conferencing?). This software-centric view of context highlights the importance of two things. One, context is always described with respect to an object. In this case it is the software

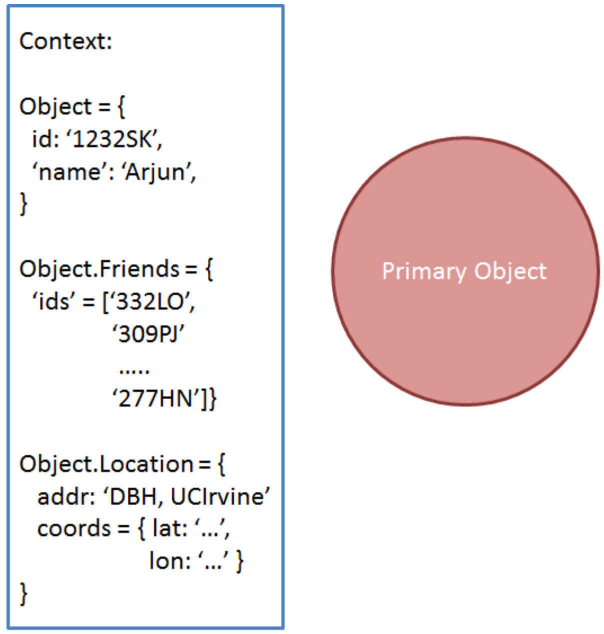

Figure 2.1: Information related to the situation of an Object.

which runs on processors distributed in a real world environment. Second, context is used to determine how this object interacts with events and entities near it? For example, Schilit uses the example that a workstation should automatically load his favorite text editor when he approaches it; and an rooster music sample must be played whenever fresh coffee is prepared. Both very different and precise interactions even though they might share common background (environment or participating entities). We would not expect a text editor to be shown when coffee is prepared, and the rooster music to be played when an employee walks to a workstation.

In his seminal paper, Anind Dey [13] describes context *as any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves*, as shown in figure 2.1. He proceeds to explain this definition with the example of an "indoor mobile tour", arguing that there are there are two additional pieces of information which can be used: *weather* and *presence of other people*. if the user is present with his friends, they might visist sites that are of interest to everybody. There the presence of other people is important context. Because the tour is indoor, weather does not affect the application. It is true that the weather has no direct affect on the application but what about the following scenarios:

- Could we use the weather information to serve different drinks in the cafeteria? On a cold day, placing the hot chocolate kiosk next to the entrance and the ice cream kiosk closer on a warmer day might boost some sales? And add to the overall experience of the tourists?

- If the tour is similar to Alcatraz, where a ferry ride takes people to the island, and back from it, a storm brewing in the ocean could lead to disrputed ferry services. Should the application warn its users who are liesurely touring at this time? Or should they continue the tour at the same pace, miss the last ferry and spend the night at Alcatraz? After all, accommodation is not a problem.

They then proceed to define Context-Aware computing as follows: *A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's tasks.* But, we need to ask ourselves why a system which uses this "additional information" should be considered a context-aware system. There are numerous system which simply would consider these "additional information" as regular inputs. What is the different between a system which takes in these inputs as processes them as regular data, and one which processes them as context?

Karen Henricksen et al. [26] makes the following interesting observation about context: Context information exhibits a range of temporal characteristics. Some context information can be static, for example the attributes of people using a system (for example, the sex of a person). But a large amount of information is dynamic. For example, relations change between people, location and events progress between moments, as shown in figure 2.2. There is no straightforward way to obtain this dynamic information other than through sensors.

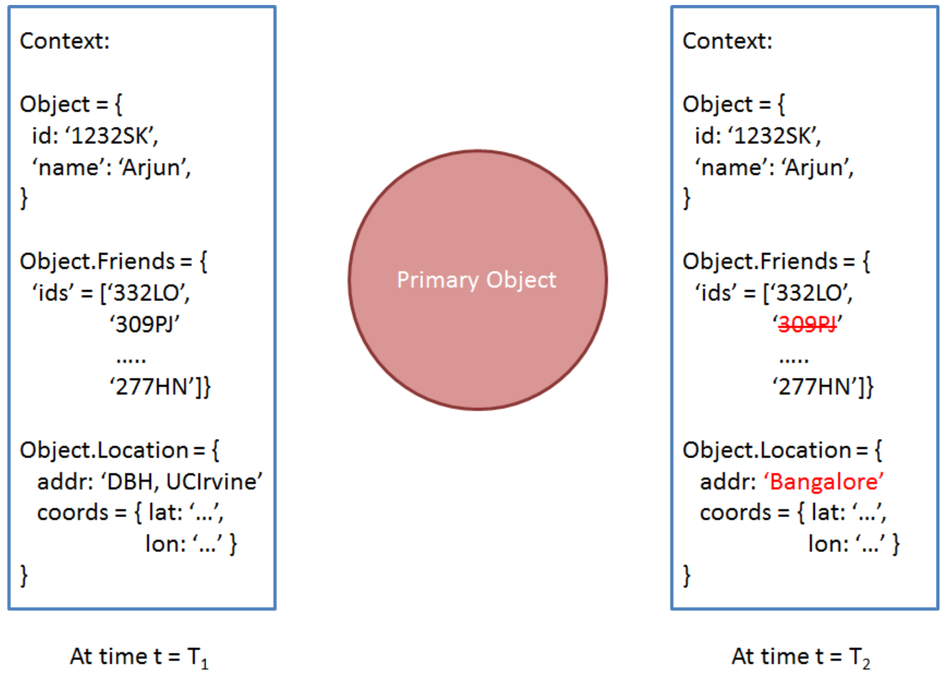

Figure 2.2: Henricksen's observation about temporality of Context.

But, such a approach tightly couples the application logic to the types of sensors used, and requires the system to convert the input data to usable representations. For example, the application might explicit modules to convert GPS coordinates to readable addresses. The problem with such an approach is that there are many ad-hoc modules built to tackle the sensors, and therefore causing the context-awareness to be tied to a specific application.

More recently, Vaninha Vieira et al. [48] uses a rule centric view of context to design their context sensitive system, Cemantika. Vaninha defines a contextual element as any piece of data or information which can be used to characterize an entity in an application domain, whereas the context of an interaction between an agent and an application is the set of instantiated contextual elements that are necessary to support the task at hand. Context awareness, for them, is to explicitly change the task the system is executing under different conditions, as shown in figure 2.3. For this they explictly model the *context sources* which includes heterogenous and external sources like sensors, user dialog interfaces and databases. This allows the various processes to operate independently of the type of sources. It should

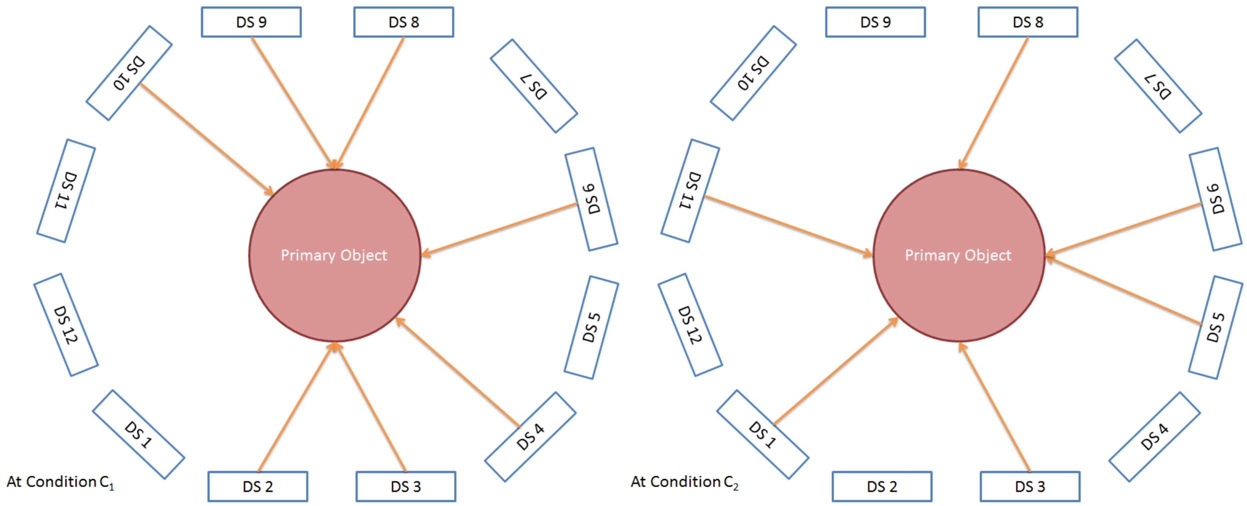

Figure 2.3: Modern context aware system obtain data from different sources upon different rule based conditions.

be noted that the use of ontologies is describing knowledge and context sources is becoming increasingly popular. A more listing of relevant work is provided in chapter 3.

The common ground behind these definitions is their object centric view of context. Context is largely a set of objects which sorround a primary object (whose context is in question). This is where Henricksen's observation of temporality raises an important question – how do we decide what goes into this set, and what doesn't? Does this set remain fixed forever? An object centric view cannot provide answers to these questions. Especially, with the current trend of ever increasing list of sensors and data sources in the world, there has to be a way of reasoning what is relevant and what is not relevant context. In order to provide some answers, we adopt a **relation-centric view** of context.

Let us extend the tour guide example to demonstrate the different properties of context. In this extension, we assume that this is the tour guide software to manage a visitor's experience at the Alcatraz Island, San Francisco. The visitor is allowed to walk through the exhibits as s/he pleases, with the tour guide headset providing explanations on the current exhibit, by sensing the current location. The explanations provided are such that they take into consideration what the visitor has already seen at the prison. For example, if the current

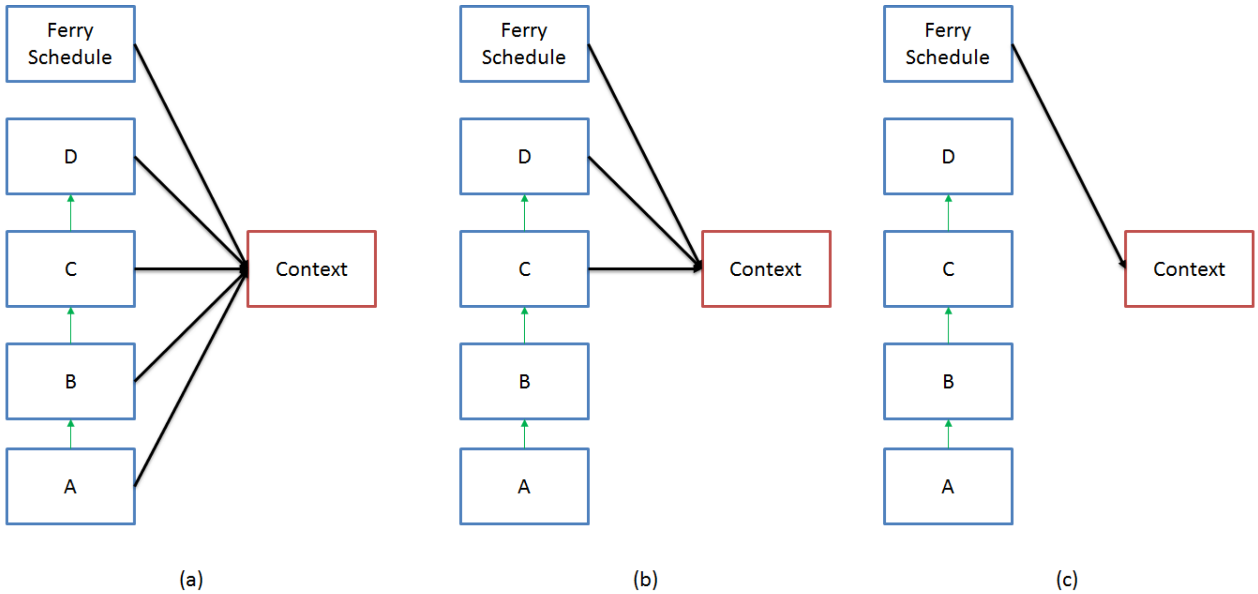

Figure 2.4: The set of items which contributes relevant context changes as the relations of the primary object with them change.

location requires knowledge of an event which happened at a different section of the prison, then it must not be told. The application must also ensure that all visitors make it to a ferry ride before the last one departs for the day. If there is any problem in that respect, it must present the appropriate data which informs the visitor on the reason for the problem. Since this application knows where every visitor is at a given time, it must ensure that some sections of the Island do not get more crowded that the others. It should either slow down or move the visitors faster if the number of people in a section are increased beyond a particular threshold.

Let's assume that the Island is divided into sections, each of which contains a person counter sensor. The counter sensor can be queried to determine how many people are present in the section. Each visitor carries a handheld device which is capable of sensing its location, and proximity to an exhibit. The ferry maintains a schedule, which is read by the system to make decisions. If there is a problem with the ferry, a maintenance schedule which can be queried to check for any damages to the ferry. We also assume we have access to a local weather channel to check for sudden changes in the weather which might affect the ferry.

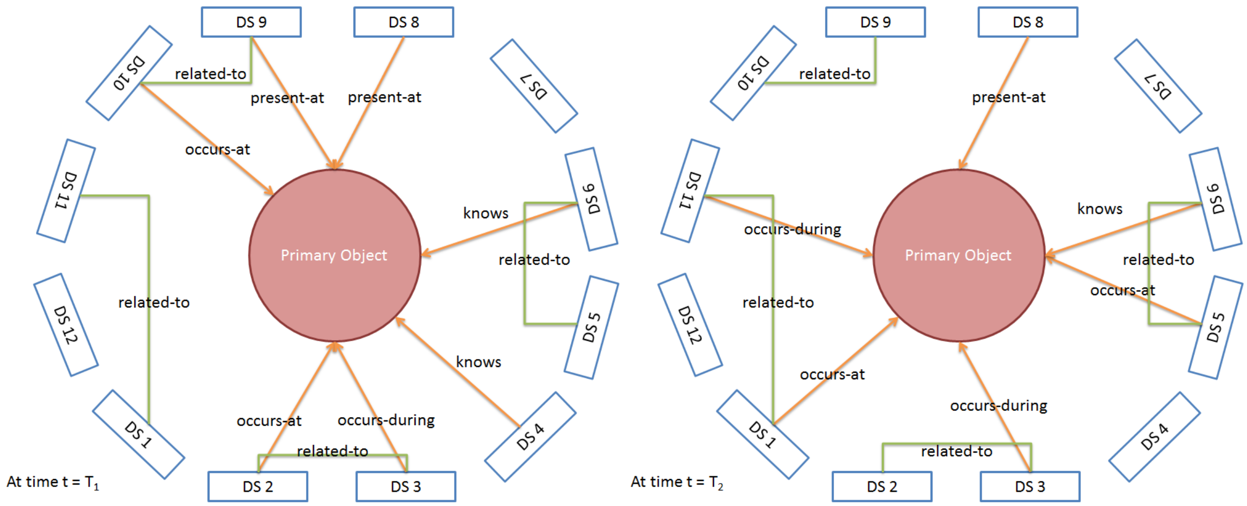

Figure 2.5: Utilizing relations to define context for the primary object.

When a visitors enters the museum, all sensors are contributing towards the context of the visitor. The person counters are being queried to decide which section the user should move to next, the previous locations are used to form coherent explanations at each new stop. The ferry schedule is checked to make sure there are no disruptions along the way. But what happens when the visitor has finished seeing all the exhibits? We can now ignore all the previous locations, and traffic situation at the different sections. Even though these sensors are in the proximity of the visitor, they are no longer needed. And therefore, contribute no useful context. We can also say **that the relationships between the visitor, and his environment has changed through time**. Thus, it holds to say that the exact configuration of relationships, at that instant of time, is pivotal in deciding what real world information contributes to context, and what is unecessary.

Relationships can be of different types. They can be simple labels like `friend-of` signifying a social relationship. Or they can more actionable like `located-at`, which relates a person to a particular location, and therefore causes a particular audio stream to play through the handheld device. This relation is not just a label, it imposes constraints on the properties of objects which it relates. Here, the spatial attributes of the the person and the exhibit must match if they are being related through a `located-at` attribute. In this dissertation, we will see that such relations, which impose such property constraints are critical in algorithmically determining which information is relevant context.

We define context of a given object at a particular time as **"the real world information which can be related to the object directly or indirectly through a known set of relationships"**. Context-awareness of a system is its **"ability to explore different types of information to identify context relevant to the objects in its ecosystem, and using this additional knowledge to reduce the complexity of a given task"**. In figure 2.5, the knowledge of relations between real world objects is marked in green, and those between the primary object and other objects in the environment are marked in orange.

2.2 Properties of Context

2.2.1 Object Specificity

Context is always specified with respect to a real world object. This primary object must be uniquely identifiable in the computational system, and must be an instance of one of the known classes. This object must have some real world attributes. For example, if the object is an event, then the interval during which it occurs, and location of occurence are two real world attributes. In the tour guide application, the visitor is one possible primary object whose context needs to determined.

2.2.2 Relation Centric View

We believe that any context is any event or entity which can be *related* to the variables in a problem. It must be noted that focus has now shifted from finding objects which are of a specific type, and that to objects (of any type), but related to the problem variables through specific relationship types.

We can design two relations for tour guide system: present-at, proceeds-to. The present-at relation relates an Island section to the primary context (the visitor), and the proceeds to relates the future sections to the visitor.

2.2.3 Temporal Relevance

The nature of relations between data in the environment and the primary object is temporal. If a source to be relevant throughout the entire interval of interaction between the primary object and the environment, we say that the relevance is high. And if the object provides context for a short duration of time, we say that the relevance is low. In the tour guide system, the relevance of the earlier sections is lower than the later ones.

2.3 Properties of Context-aware System

2.3.1 Real-World Knowledge

Modeling real world knowledge (what are the different objects in the world, and how do they precisely interact with each other) is critical in context based systems. This is in contrast with rule based systems, where a set of real world conditions are sensed to trigger a particular action. Examples of knowledge are: An academic conference has atleast one keynote talk; or Sodium reacts with atmospheric oxygen at room temperature; or water expands when it freezes. A network of knowledge constitutes the model for a context based system. This paves the way of it to expand its knowledge about the current situation of the primary object based on what is already known about the object, and what is needed. For example, if an Object is associated with a keynote talk, data about the co-occuring conference should be obtained. Alternatively, if the system had associated a conference with the object, data about the keynote must be sought out. This 'guiding light' trait of knowledge is a pre-requisite in a context-aware system.

2.3.2 Dynamic Linking

The relation centric view, and temporal relevance properties lead us to how the primary object is linked to different sources in the environment. If the primary object is linked to all sources in the environment, we say that it *statically links* to these sources. For example, if the tour guide application which brings in data from all sources all times. On the other hand, dynamic linking with only relevant sources has the ability to restrict input from many sources, and therefore be more capable in future ecosystems where thousands of sensors from the web or in the local network are availble. Also, such a design allows system developers to add more sources in the case of "Black Swan" events, which were not foreseen before, but are now posing a challenge to the performance of the system.

Also, context has been used in many non-real world problems. For instance, in natural language processing [30], ranking pages of the web [39], entity resolution [10], face clustering in images [50] and therefore it must not mean that contextual techniques must apply only to real world problems. For the purposes of this dissertation, we ignore the application of context in such problems.

2.4 Parallels

The real world is very big. Modeling it can be a very hard task. Do we really need such large and elaborate models if we are to call our application "context-aware"? In this section, we look at how *contextual thinking*[8] has helped understand, and in some cases solve, many of the long standing problems in different disciplines. We will start with an anecdotal example, and move to more elaborate examples which demonstrate how to reason in real world problems. The reason for this section is to justify the need for modeling many different types of real world information, which is a pre-requisite for employing context based techniques.

2.4.1 Black Swans

In his widely acclaimed book, The Black Swan[46], Nassim Nicholas Taleb presents many arguments against solely relying on prediction based techniques. Citing examples from stock markets to clinical psychology trials he presents the case that historical evidence is insufficient to position oneself in the future.

His example of a turkey brings to light an interesting point "Consider a turkey that is fed every day. Every single feeding will firm up the bird's belief that it is the general rule of life to be fed every day by friendly members of the human race *looking out for its best interest*. On the afternoon of the Wednesday before Thanksgiving, something unexpected will happen to the turkey. It will incur a revision of belief."

Any amount of prediction is futile in this case. Taleb plots the graph 2.6 showing the belief of the turkey in mankind. How could the turkey protect itself, yet reaping the benefits of the food given to it?

The answer lies not in developing complex techniques about reasoning what is already known. But to look at the problem from different vantage points. From the standpoint of the turkey, the Wednesday event is a Black Swan event. But from the standpoint of the butcher, it is not, since its occurrence is not unexpected. So, if the turkey explores the world around it, it will soon reach a point where it is able to reason the nature of human behavior, their customs, and finally arrive at the conclusion, that there exists one such custom which is

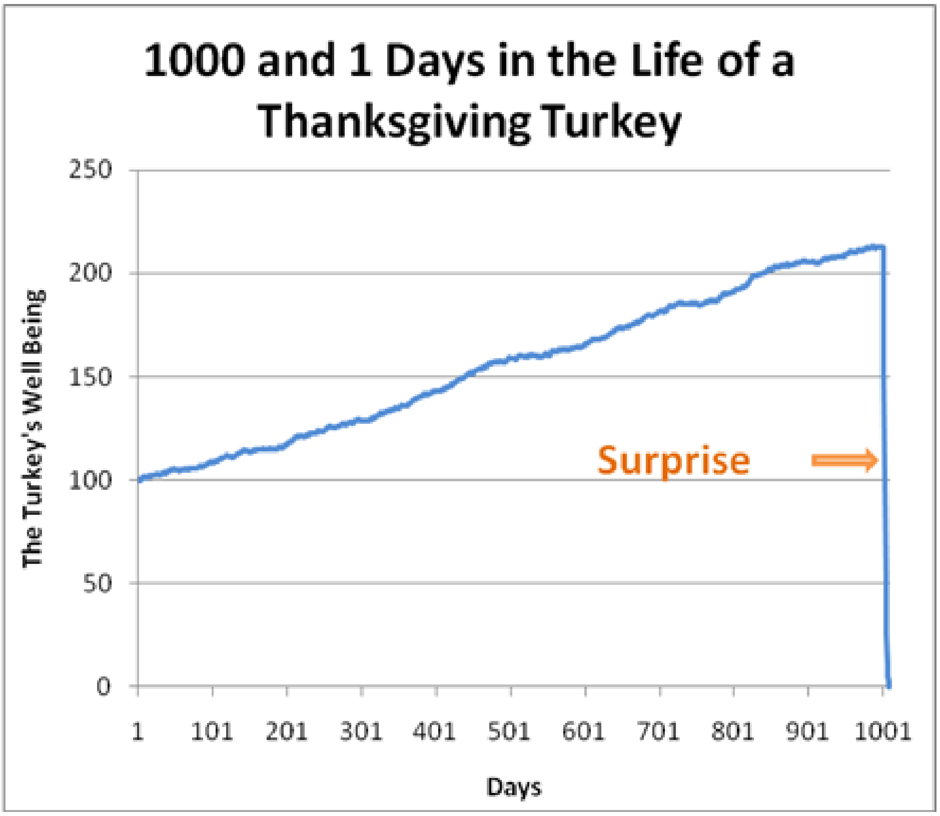

Figure 2.6: Expectations of a Turkey as Thanksgiving approaches.

absolutely disastrous for the turkey's well being.

The simple example shows how new objects bring with them different semantics. And highlights that interactions can be disrupted heavily with even the slight change in relations. The second thing it shows is the need for awareness in the turkey to "look-out" for potential causes of harm. This action of being context-awareness is criticial in any real world systems.

2.4.2 Context in History

Historians love context. Almost all their works heavily rely on finding interesting events in a particular timeframe which can explain the events that followed. In his book, Guns, Germs and Steel [15], Jared Diamond argues the need to understand specific environmental diversities, and use them to reason why history happened the way it did. We are familiar with the conquest story of the Inca emperor Atahuallpa by the Spanish conquistador Francisco Pizzaro at Cajamarca, Peru in 1532[1]. Historians attribute the success of Pizzaro to better Spanish ammunition and warfare techniques. But the more interesting question is *Why was Pizzaro at Cajamarca conquering the Incas, and not Atahuallpa crossing the Atlantic to conquer Spain?* Effectively, how did South Americans evolve so different from Europeans? The answer, it turns out, lies in different environmental conditions.

Diamond outlines the various steps that led to such cultural diversity in a flowchart similar to 2.7. Let us see how he arrived at it. Any conflict usually favors the party with a stronger political, military, economic and social structure. The South American society consisted of the high ranking chiefs who were treated as Gods, and were the only people permitted to read and write. Everyone else was engaged in the daily activities of food production. Thus, a majority of the man power was spent in solving the daily food problem. There was no scope for any organizations to develop which would construct and develop military and economic disciplines. On the other hand, these organizations not only existed in Europe, but also *co-existed* each other, thereby driving each other to be more mature and stronger. The stronger organizations led to creation of stronger ammunition (guns, steel, sword), transport services (ships which could travel across the atlantic), writing technology (since everyone in Pizzaro's army was capable of communicating messages through time and space).

Why did the Europeans form such a stratified society, and not the South Americans? The majority of the south americans were concerned about procuring food for the next day. Food production systems in Europe reached a critical peak, because of which they developed technology for storing food. Because day-to-day food was no longer a concern, a significant section of the population now was "free" to indulge in other activities. This led to development of art, policital, military units, economic systems, which in turn evolved each.

Why did food production reach an all time high in Europe and not in South America? The reason lies in three environmental factors: the soil, flora and fauna of the

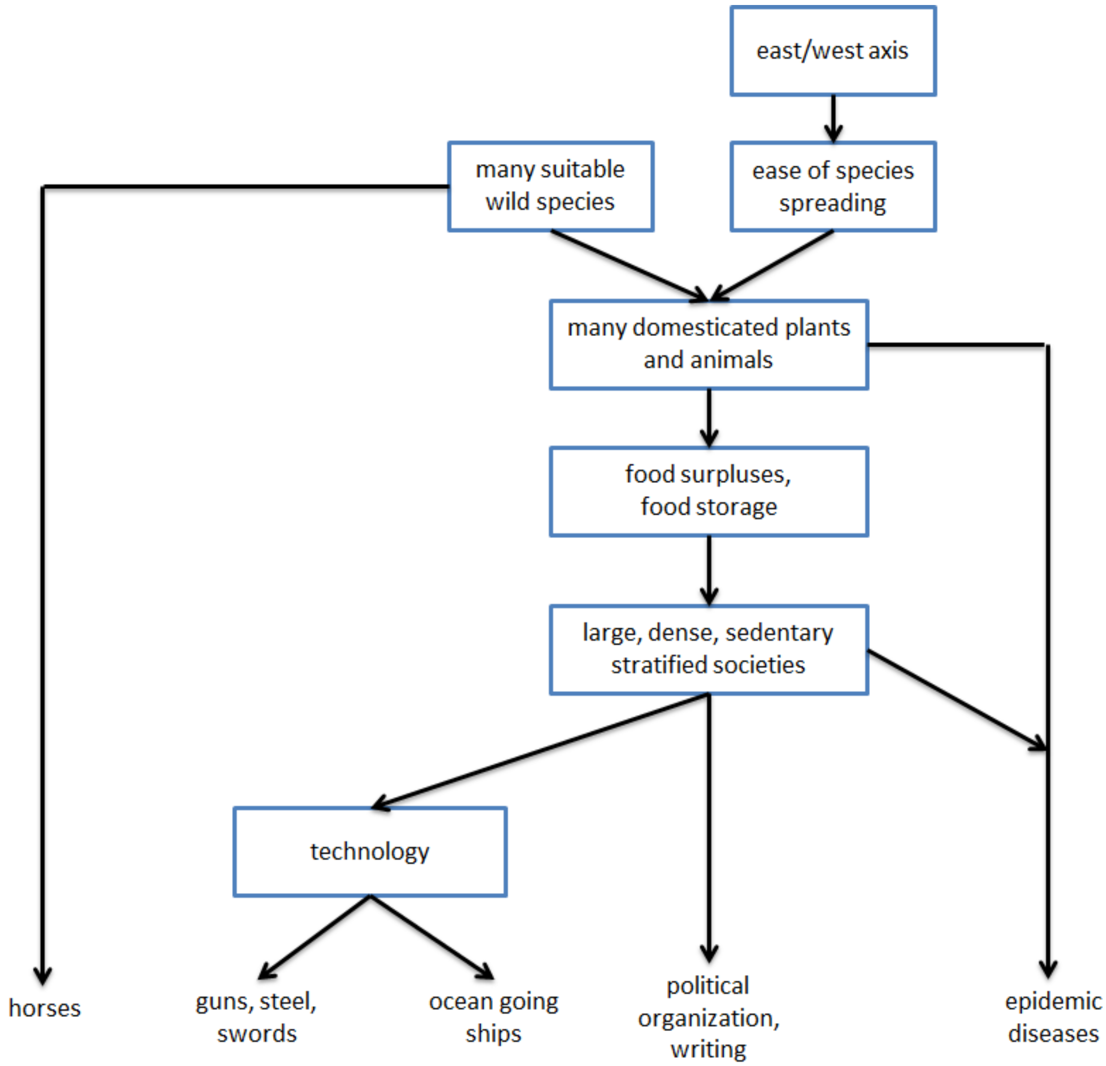

Figure 2.7: Reasons for diversity among cultures.

area and how they interacted over a larger time span. Europe and Asia has been home to a more diverse set of animals and plans. Since 6000B.C., farmers and animals of the area were able to choose from this bigger variety of plants, and evolve them over thousands of years to form better crops. The role of a larger set of animals in the area is interesting too. Over the years, Europeans have domesticated most of the animals than people from any part of the world. Big mammals like cows and bulls were found in plenty in Europe and Asia. This led to much higher yields than tilling the soil by hand. Regions like South America lacked such variety of domesticable animals. For example, there is no point in domesticating an elephant, as it takes 14-15 years to reach adulthood and be of use. Animals like Rhinos are very strong, and therefore, can be excellent farm animals, but are very hard to tame (in fact they are extremely dangerous). These reasons led the South Americans to rely only on man power to tackle the daily food problem.

Finally, why did biodiversity emerge in Europe/Asia, and not in the Americas or Africa? Thousands of years ago, the biggest deterrent in sustaining life was the climate. People lived nomadic lifestyles which came in contact with different weather, and acclimatizing to it required almost a reboot of their knowledge, environmental know-how and customs. Now, lets take a look at the map of the world, shown in 2.8 – what do we see? Europe and Asia span longitudinally, whereas Americas and Africa extend along the Earth's latitude. What is the biggest advantage this offers Eurasians? When they moved from place to place, the structure of their continent allowed them to move to places with **similar weather**. This allowed them to move to different regions and enjoy the same weather, grow similar crops and domesticate the same animals. But people in Africa or America had to move out of their comfort zone, and move to area containing different weather and biodiversity, and had to redo everything from scratch. Effectively, cultures which could move freely without rebooting their knowledge found it easier to develop than others.

There is also this idea that cultures which had to undergo such reboots frequently had to deal with more complexities of the world, and could rely less on objective knowledge, and therefore made people rely more on religious principles to lead their lives. This led to prevalence, and eventually domination by religious organizations.

Diamond's book is an exploration of the broadest pattern of history. What it tells us is that explicit modeling of real world objects with their native properties are extremely important in reasoning about the world.

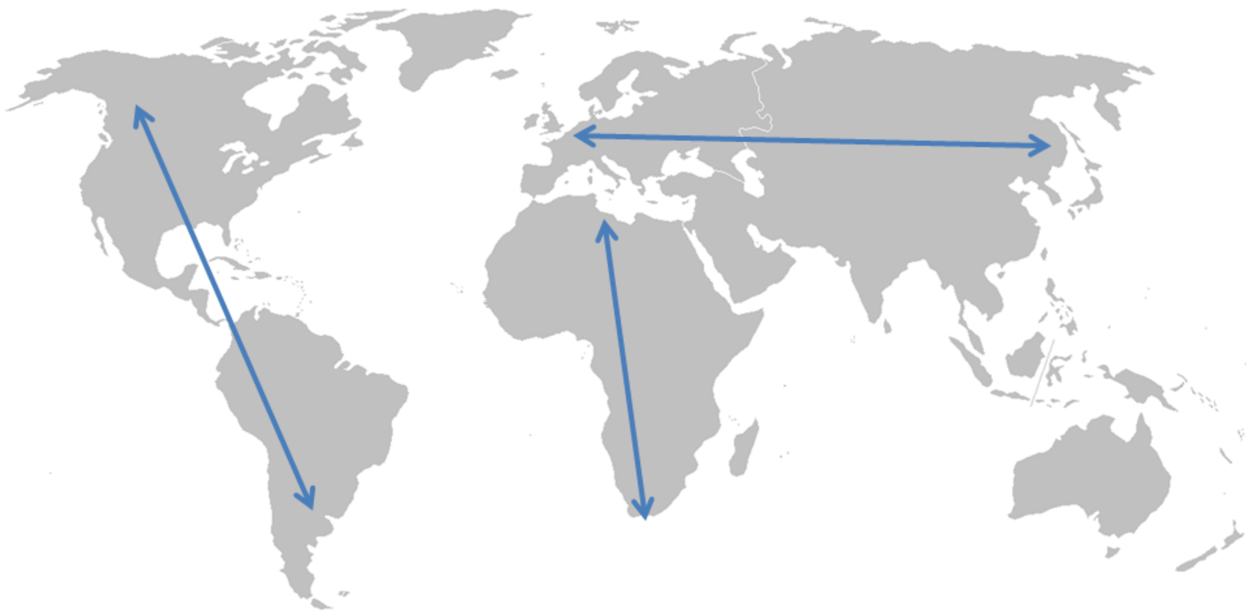

Figure 2.8: Horizontal/vertical extents of different continents.

2.4.3 The Gaia Hypothesis

Charles Darwin's work describes how living being have evolved over billions on years. But it does not answer the question – how does life sustain itself on Earth? It turns out that the few billion years of life we have had is too little to let life have evolved randomly, and just let natural selection determine who succeeds. There had to be other factors which let life sustain and grow in specific ways.

One of the theories put forward by James Lovelock and Lynn Margulis in the 1970s, more commonly known as the Gaia Theory, proposes the presence of natural feedback loops to explain the sustanence of life[32]. Their argument is that for life to sustain, the amount of carbon dioxide in the atmosphere must not exceed a certain threshold. But who keeps this in check? How has the amount of CO_2 in the atmosphere been almost constant for the last millions of years. Their reasoning is as follows:

The Earth's volcanoes have spewed out huge amounts of carbon dioxide for millions of years. Plant and animals recycle massive amounts of CO_2 in the process of photosynthesis, respiration and decay. According to the Gaia theory, the excess of CO_2 is recycled by a vast feedback loop, which involves rock weathering as a key ingredient. Rainwater washes atmospheric CO_2 onto rocks. In the process of rock weathering, rocks, rainwater and CO_2 form chemicals known as carbonates. The carbonates are then washed down where tiny algae invisble to the naked eye, absorb them and use them to make shells of chalk. So the CO_2 that was in the atmosphere has now ended up in the shells of those minute algae. In addition, ocean algae also absorb CO_2 from the surface of the ocean.

When the algae die their shells are then washed down into the ocean, where they form sediments of limestone. Limestone is very heavy, and gradually sinks into the mantle of the earth. Eventually, some of the CO_2 is released back to the atomsphere by volcanoes. The entire cycle – linking volcanoes to rock weathering, to soil bacteria, to oceanic algae, to limestone sediments, and back to volcanoes – acts as a feedback loop which regulates the Earth's temperature. It turns out that bacteria in the soil vastly increases the rate of rock weathering (acting as a catalyst). In the sun gets hotter, the bacteria in the soil get higher than average ambient temperature, which causes them to increase the rock weathering activity, causing more CO_2 to be removed from the atmosphere and sent to the ocean, thus cooling the planet.

This remarkable explanation involves rocks, bacteria and ocean dynamics and the even the sun to explain how earth's temperature is regulated. By studying one or more objects in isolation makes the analysis far from complete. More interestingly, the relations between the objects are very precise, involving specific biochemical reactions within some acceptable ambient conditions.

2.4.4 Significance

These examples are not to present here to report the latest advances in different disciplines, but to glance at how reasoning is done in the real world. Both the use cases require very precise knowledge of relations between different objects in the world (for example, how the sun can affect bacterial growth in the soil), and have access to a large amount of sensors and data sources. Once these two have been established, the computational challenge is how to form accurate models of the models of the world, and how to use those models to either reduce the complexity of the problem so a human can operate with ease, or apply additional algorithms to solve the problem.

In the next section, we see what we types of information we shall consider as context for the problem of tagging personal photos.

2.5 Context for Personal Photos

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 the following:

- **Event Objects**: includes description of events like conferences, parties, trips or weddings, and their structure (for example, what kind of sessions, talks and keynotes are occurring within a particular conference).
- **Entity Objects**: information about a user's social graph, people whom she corresponds with using email and other messaging services.
- **Geographical Objects**: various tools like Facebook Places, Google Latitude or Foursquare provide information about where people are at a given time.

The important relations which dealing with such data are:

- **Subevent Relation**: If two events occur such that, one spatio-temporally contains the other, we say that it is the subevent of the one with larger spatio-temporal span. For example, a talk event is a subevent of the conference during which it happens.
- **Participation Relation**: If an entity is participating in an event, s/he is said to be a participant-of that event. Note that this relation constraints the spatio-temporal boundary where the entity could be present during the interval of the event.
- **Social Relation (knows)**: Social relations relate people who are acquainted with each other. This is used to model social networking information obtained from sources like Facebook or Email.

- **Spatio-Temporal Relations**: Events occur in specific time intervals, and at some location. We use the relations occurs-at and occurs-during to model these properties. More details are provided in the next chapter.

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. We classify sources into the following types:

- **Personal Data Sources**: include all sources which provide details about the particular user whose photo is to be tagged. Some common examples of personal data sources include Google Calendar, Email and Facebook profile and social graph.
- **Social Data Sources**: include all sources which provide contextual information about a user's friends and colleagues. For example, LinkedIn, Facebook and DBLP are some of the commonly used websites with different types of social graphs.
- **Public Data Sources**: include all sources which provide information about public organizations (like restaurants, points of interest or football stadiums) or about public events (like fairs, concerts or sports games).

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.

At this point we should revisit 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, as now, sources are not hardwired to photo, but instead need to be discovered gradually.

In chapter 4, we will see how these different objects, relations and data sources are used by out context-aware framework to assist tagging faces in personal photos.

Chapter 3

Related Work

The role of context in computing has been studied in [9]. The use of context in image retrieval is emphasized in [28, 12]. Barthelmess et al. extract semantic tags from noisy datasets containing discussions, speeches about a set of photos in question[4]. Naaman et al. have exploited GPS attributes to extract place and person information [35, 36]. Rattenbury [40] devised techniques to find tags which describe events or places by analyzing their spatiotemporal usage patterns. Ames and Frohlich [3, 19] independently describe a survey conducted to study motivations for people to tag their photos. They noticed two broad motivations: Organization of photos and Communication with photos. Time alone is used for organizing photos in [21, 23]. Brave new world applications for photography have been described in [20, 16], where life logs were collected in the form of photos, emails, document scans and stored in SQL Server database, and photos were retrieved using SQL queries. The photo content was tagged by the user in this case. The Computer Vision community has contributed extensive work in the area of detecting scenes [49], humans [11] or geo localization [25]. Context information and image features are used in conjunction by [38, 6, 5, 7] identify tags. The semantic web community is using linked data technologies to annotate and query photographs [34, 37]. Collaborative games also have been evaluated as a possible way to tag photos[14]. Systems like Picasa, iPhoto and [21] organize photos based on time, GPS coordinates and sometimes faces in the photo. These attributes of the photo do not capture event semantics [42]. Events are a natural way of categorizing photo media. Events also allow large number of photos captured during a single event be organized hierarchically using subevents.

The core of their view comes from the earlier proposed definition from Brezillon [CITE], who considers context to be formed from three types of knowledge: Contextual knowledge (knowledge about entities, their environments), external knowledge (knowledge which is currently not relevant), and proceduralized context, which has to do with the reasoning applied to the current situation.

150 definitions of context by Brezillon.

3.1 Ontologies

Lack of facilities to express spatio-temporal constraints.

3.2 Data Integration

3.3 Computer Vision: Face Recognition/Verification

Chapter 4

Context Discovery Framework

In this chapter, we will see how to dynamically discover context from various data sources. And how this discovery can be used to assist the face tagging application.

4.1 Pruning Search Spaces with CueNet

Automatic media annotation algorithms essentially assign one or more labels from a search space to a given input image. Figure 4.1 shows the various approaches of constructing such a search space for such an algorithm. The traditional approach is shown in 4.1(a). These spaces were limited to a set of labels chosen by an expert, with no way of pruning the search space in case it got very large. The focus was instead on extracting the best features from images, to obtain high overall classification accuracy[47].

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 like conferences, weather events and sports games. With such context sources, the search

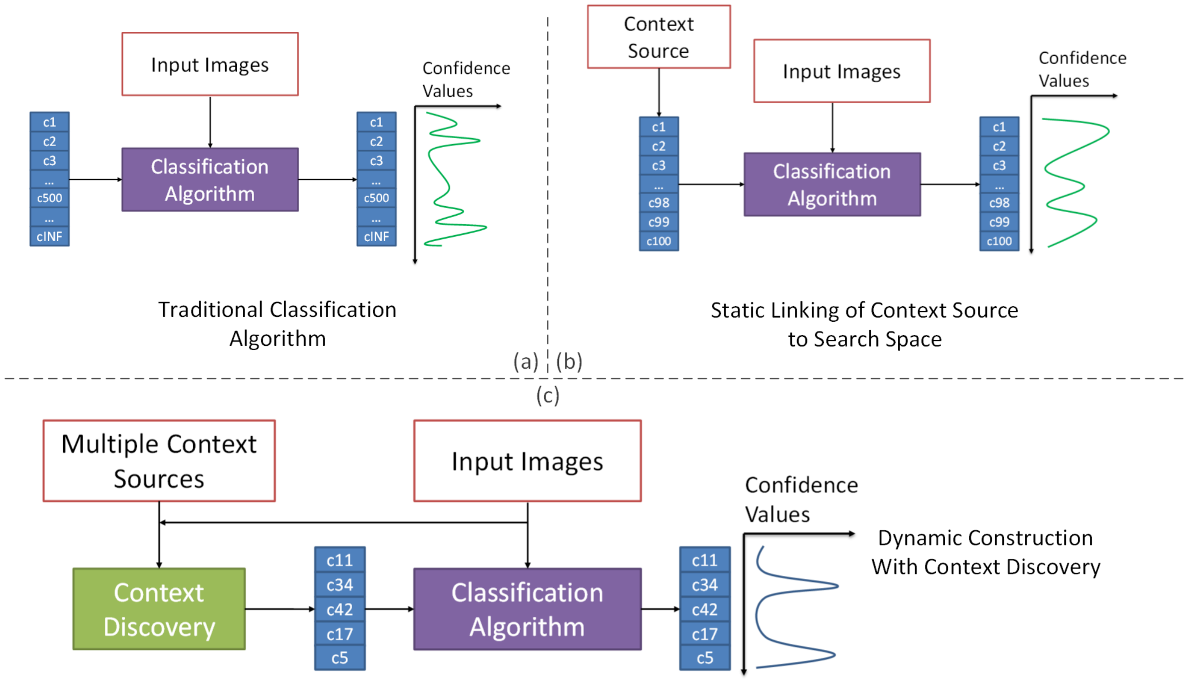

Figure 4.1: The different approaches in search space construction for a multimedia annotation problem. A traditional classifier setup is shown in (a) where the search space candidates are manually specified. Context is used to generate large static search spaces in (b). The desired framework is shown in (c), which aims to produce small search spaces with many correct annotations.

space construction is being delegated to one or a few sources [27, 31, 36, 38, 45] (figure 4.1(b)). These approaches rely on a single *type* of context. For example, time and location information or 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 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 4.1(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 a classification algorithm to exhibit

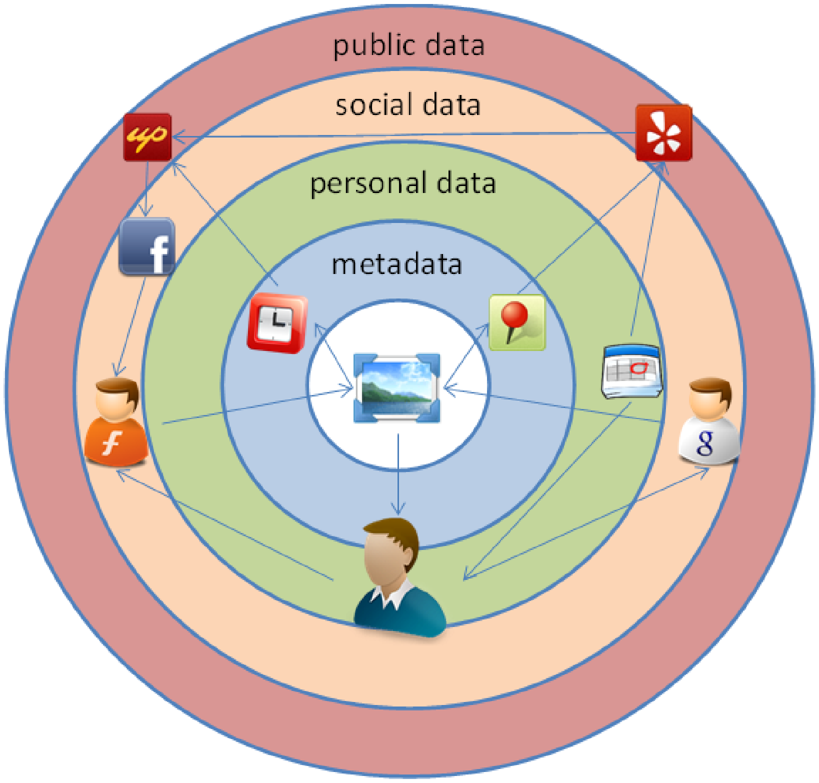

Figure 4.2: Navigation of a discovery algorithm between various data sources.

superior performance.

The CueNet framework, 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 ensure 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 by different users. These photos could contain friends, colleagues, speakers giving very interesting talks, or newly found acquaintances (who are not yet connected to the user through any social network). This makes the conference photos particularly interesting because no single source can provide all the necessary information. It emphasizes the need to utilize multiple sources in a meaningful way.

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 have 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 (as in [29]), which can perform very well as their search space is limited to two candidates.

Figure 4.3 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

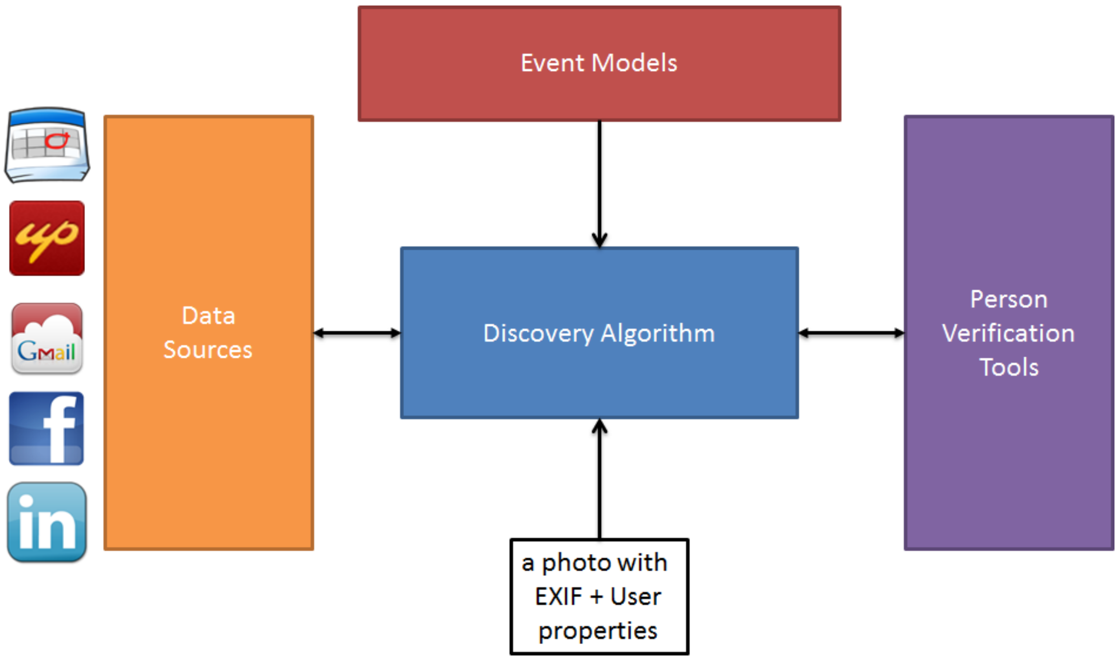

Figure 4.3: The Conceptual Architecture of CueNet.

with 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.

4.2 General Approach

Figure 4.3 shows a high level architecture of CueNet. The major functional blocks consist of a data integration system (left), which provides a uniform query interface to a multitude of autonomous data sources, which may reside within an enterprise or on the World-Wide Web [24]; a specification of model describing real-world knowledge in terms of objects and their relations, along with any axioms and constraints to be imposed on instance graphs (top); since we are assisting face tagging application, the final block (right) consists of a set of hooks to invoke appropriate face tagging algorithms by providing a set of candidate for the input photo. In this work, we shall assume verification semantics in such a tagging algorithm, where given an input photo and a candidate person, the algorithm returns true or false (with a confidence score). Face recognition models would have to be retrained when the candidate set changes. Also, as described in chapter 3, the state-of-the-art techniques for face verification perform much more reliably than their recognition counterparts.

4.3 Execution Trace

Use the above introduction to describe execution traces for two scenarios (probably from the demo?).

4.4 Event Model

Our ontologies extend the E* model[22] 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

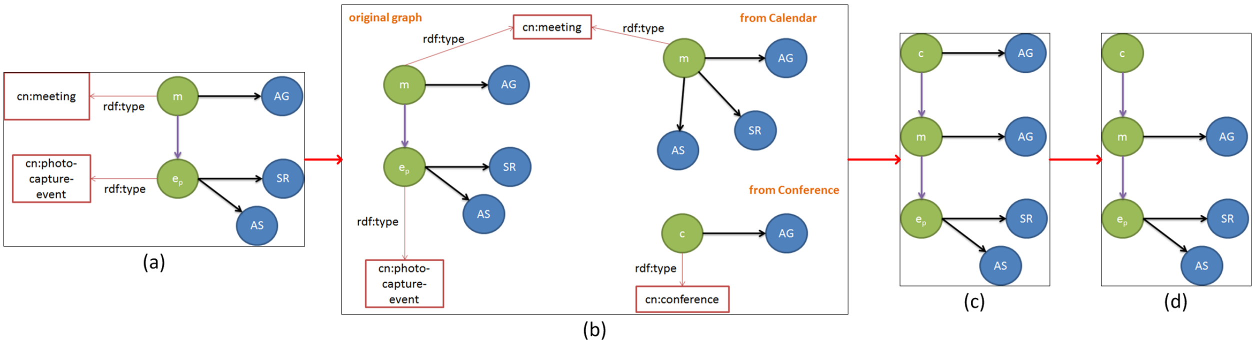

Figure 4.4: The various stages in an iteration of algorithm 1.

is a participant of E if the event's spatiotemporal span contained that of the participant.

$$\texttt{participant}(\mathrm{E}, \mathrm{P}) \iff (\mathcal{L}_P \sqsubset_L \mathcal{L}_E) \wedge (\mathcal{T}_P \sqsubset_T \mathcal{T}_E) \quad (4.1)$$

The symbols $\sqsubset_L$ and $\sqsubset_T$ indicate spatial and temporal containment respectively. Please refer to [22] for more details. 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.

4.5 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
   (:attrs url name time location title)
```

```
    (:rel conf type-of conference)
    (:rel time type-of time-interval)
    (:rel loc type-of location)
    (:rel attendee type-of person)
    (:rel attendee participant-in conf)
    (:rel conf occurs-at loc)
    (:rel conf occurs-during time)
    (:axioms
       (:map time time)
       (:map loc location)
       (:map conf.title ltitle)
       (:map conf.url url)
       (:map attendee.name name)))
```

A source declaration comprises of a single nested s-expression. We will refer to the first symbol in each expression as a keyword, and the following symbols as operands. This above declaration uses five keywords (`source`, `attrs`, `rel`, `axioms`, `map`). The `source` keyword is the root operator, and declares a unique name of the data source. The source mapper can be queried for finding accessors using this name. The `attrs` keyword is used to list the attributes of this source. Currently we assume a tuple based representation, and each operand in the attrs expression maps to an element in the tuple. The `rel` keyword allows construction of a relationship graph where the nodes are instances of ontology concepts. And edges are the relationships described by this particular source. In the above example, we construct individuals *conf*, *time*, *loc* and *attendee* who are instances of the *conference*, **time-interval**, **location** and **person** class respectively. We further say that attendee is a ***participant of*** the conference, which ***occurs-at*** location loc and ***occurs-during*** the interval time. Finally, the `mapping axioms` are used to map nodes in the relationship graph to attributes of the data source. For example, the first axiom (specified using the map keyword) maps the time node to the time attribute. The third map expression creates a literal called title, and associates it to the conference node, whose value comes from the ltitle attribute of the conference data source.

Formally, we represent the given ontology as O. The various classes and properties in O are represented by C^O and P^O respectively. Since our upper ontology consists of DOLCE and E*, we assume the inclusion of the classes `Endurant`, `Perdurant`, `Event` and `Person` in C^O. Each source S consists of three parts, a relation graph $G^S(V^S, E^S)$ where the nodes $V^S \in C^O$, specify the various "things" described by the source. The edges $E^S \in P^O$ specify the relations among the nodes. Any graph retrieved from such a source is an instance of the relation graph, G^S. Further, the tuple A_T^S consists of the attributes of the data source. Finally, the mapping $M^S : \{G^S \rightarrow A_T^S\}$ specifies how to map different nodes in the relation graph to the different attributes of the native data source.

4.6 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:

$$\begin{aligned} v_i^I = v_r^R \iff & (v_i^I.\textbf{type-of} = v_r^R.\textbf{type-of}) \wedge \\ & (v_i^I.\textbf{occurs-at} =_L v_r^R.\textbf{occurs-at}) \wedge \\ & (v_i^I.\textbf{occurs-during} =_T v_r^R.\textbf{occurs-during}) \end{aligned} \tag{4.2}$$

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`.

$$\begin{aligned} v_i^I &\sqsubset_L v_r^R, \\ v_i^I &\sqsubset_T v_r^R \end{aligned} \tag{4.3}$$

$$v_i^I.\textbf{Endurants} \cap v_r^R.\textbf{Endurants} \neq \{\phi\} \tag{4.4}$$

Here v_i^I.**Endurants** is defined as a set $\{w|w \in V_i^I \wedge w.\text{type-of=Endurant}\}$. If equation (4.4) 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 figure 4.4(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 events, equation (4.2) 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 (equations (4.3) and (4.4) 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 equation (4.3) 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.

4.6.1 Context Discovery Algorithm

Algorithm 1 below outlines the tail recursive discovery algorithm. The input to the algorithm is a photo (with EXIF tags) and an associated owner (the user). It must be noted that 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 4.4 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.

Figure 4.4 shows the various stages in the algorithm graphically. (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. A live visualization of these steps for different photos can be found at http://cuenet.site44.com.

4.7 Implementation

Algorithm 1: The Context Discovery Algorithm

Data: A photograph H, with a set of detected faces F. Voting threshold, T. The owner O of the photo.

Result: For each face f $\in$ F, a set of atmost k person tags.

```
begin

   function discover(): {
      while (DQ is not empty): {
         node = DQ.deque()
         results = query (node)
         E ← merge (E, results)
         if (termination_check()):
            return prepare_results();
      }
      reconstruct DQ ← E
      discover()
   }

   function merge(O, N): {
      remove_duplicates()
      M ← subevent_hierarchy_join(O, N)
      prune_up(M)
      if (less than T new candidates were discovered):
         push_down(M)
      else:
         vote_and_verify(M)
      return M;
   }

   E ← construct event graph with H and O
   construct discoverable nodes queue, DQ ← E
   return discover()
end
```

Chapter 5

Analysis and Experiments

5.1 Analysis

5.2 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 hypotheses. **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?

5.2.1 Setup

We use 368 photos taken at 7 different conferences in our face tagging experiment. The person database consists of 660 people. 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[18] 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 $1 \leq i \leq 8$ for each dataset). Table 5.1 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 660. This set forms the exhaustive search space, L from section **??**.

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

Table 5.1: Profile of datasets used in the experiments.

We divide the sources into different categories to facilitate a more general discussion. The

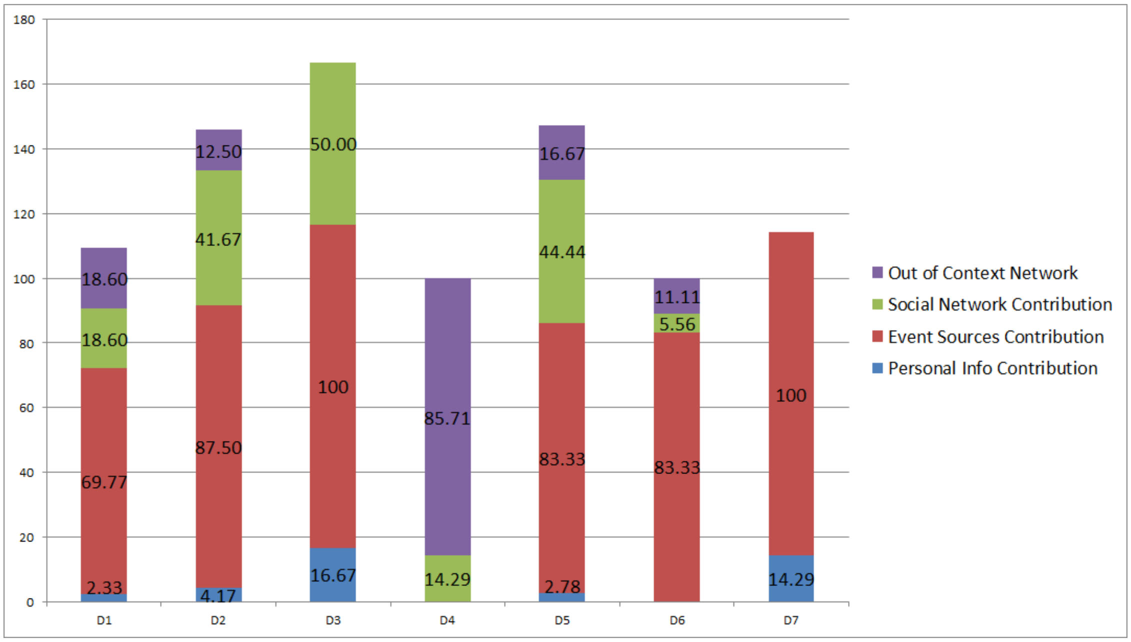

Figure 5.1: The distribution of annotations in the ground truth across various sources.

categories are "Personal Information" (same as Owner Information in section 4.6.1), "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 5.1 shows the distribution of the ground truth annotations across various sources, for each 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.

5.2.2 Context Discovery

Now, lets look at reduction obtained in state space with the discovery algorithm. The total number of people in our experiment 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 table 5.2. 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 table 5.2. For D4, the algorithm found no event sources (as seen in figure 5.1), 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.

Dataset	Reduced Search Space Size	Hit Rate
D1	42	83.72%
D2	338	87.5%
D3	231	100%
D4	1	0%
D5	254	83.33%
D6	20	88.89%
D7	7	100%

Table 5.2: Sizes of Search Space for each dataset.

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

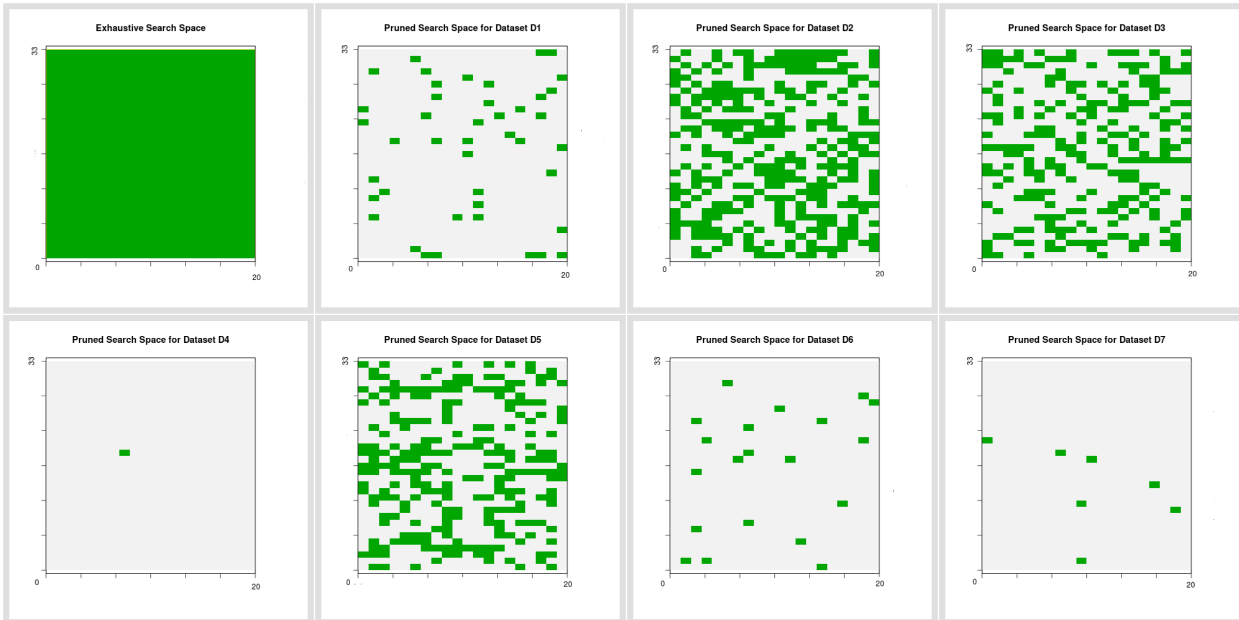

Figure 5.2: Grid plots showing the exhaustive search space and pruning in search space for different datasets.

said to be "contributed" from it. We profiled our algorithm to log all contributions which were true positives for the classification algorithm. Figure 5.4 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 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 networking sources would have led to a large number of false positives in the classifier performance. Even though the role 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.

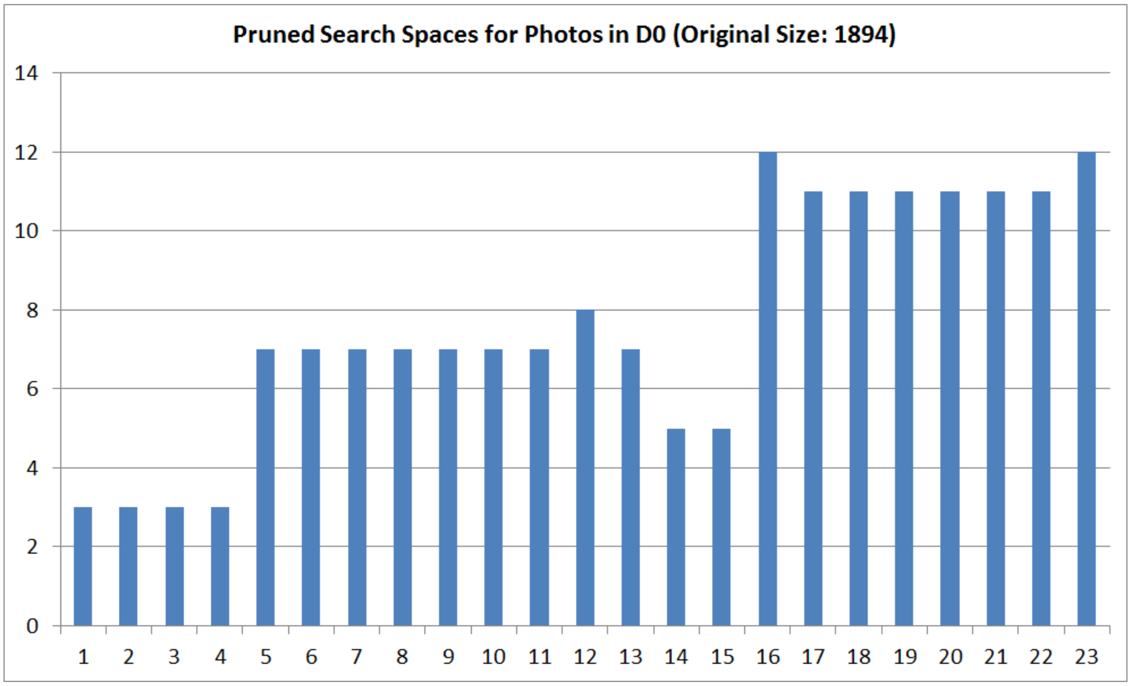

Figure 5.3: Pruned search space for photos in D0.

5.2.3 Individual List Sizes in a Dataset

Here we look at how CueNet reduces the number of possible candidates for all photos in a dataset. For this setup, the complete candidate set L, contained 1894 labels (total number of people present at the conference, user's emails and social graph). The figure 5.3 shows various statistics for each photo, which includes the maximum size of the list which was generated by the discovery algorithm, the actual number of people in the photos, the number of true positives and false positives. As it can be seen, the size of the discovered set S, never exceeded 12. This is 0.5% of the original candidate list. Because the total number of possible participants (list size) was low, our False Positive rate (FP) was very low too. Most of the false positives were due to profile orientation of faces or obstructions (this was because the face detector was smart enough to pick up profile faces, but verification worked better only on frontal faces).

5.2.4 Search Spaces

Finally, we compare the various search spaces constructed by discovery algorithm. We represent all people in our experiment universe in a color grid (with 33x20 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 5.2 shows the color grids describing search spaces for all datasets, and an exhaustive search space. The positioning of people along the grid is arbitrary, but consistent across grids. Our aim in this visualization is to see the diversity in search spaces created by the algorithm. The purpose of the exhaustive search space is to provide easy comparision to appreciate the reduction in search space.

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.

5.2.5 Conclusion

These experiments validate our three hypotheses. **First**, Event sources contain a large portion of true positives. From 70% in D1 to 100% in D7. There are events for which there is no documentation, and event sources are not able to contribute anything here, as in the case of D4. **Second**, the discovery algorithm is able to prune the search space using event, personal and social information. The reduction is atleast 50% for D2 (338 candidates out of 660) but can be very large in some cases (7 candidates for D7). **Third**, The reduced search space retains a high number of true positives. The hit rate is between 83% to 100% (with

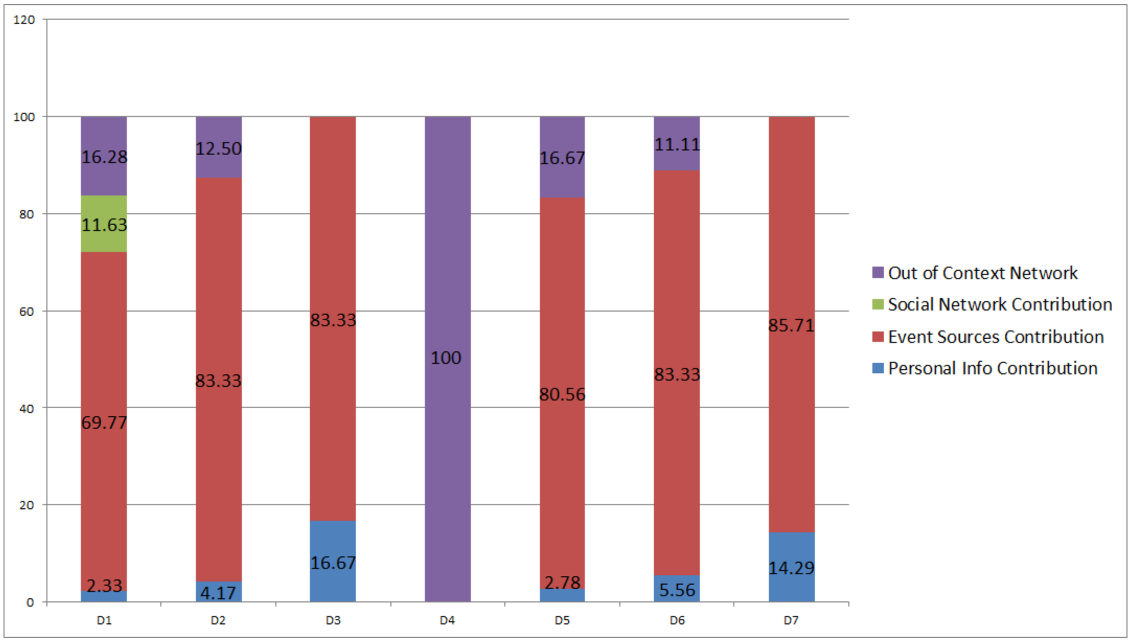

Figure 5.4: Graph showing the contribution of each source type in context discovery.

the exception of D4, where the search space provided no true positives). We also saw how unique the search spaces are, to each dataset, thereby demonstrating the dynamic nature of the algorithm.

5.3 Known Issues

In this section, we list some of the issues encountered in designing and building CueNet. Some of these are active areas of research, and whereas others are specific to our framework, and can be considered potential areas of research. Our experience with CueNet indicates that the following issues should be approached in a holistic manner, i.e., in conjunction with each other. Approaching these problems in the context of each other reduces the individual complexity of each sub-problem by possibly increasing the complexity of the entire framework, but making the problem more tractable.

Noise in Social Media: The problem of noise filtering in web data is a prominent one, and is being addressed by various communities in different ways. These range from entity matching and record linkage problems [17] to correcting missing data in information networks [41]. These problems get trickier because of the different variations in representing tiny details such as representation names of people, addresses of places, and time. In fact, there is a whole school of anthroponomastics [44] dedicated to studying variations in human names. Ideas in this field indicate that these differences arise due to cultural, historical and environmental issues[2]. Such issues cannot be trivially addressed.

Face tagging in social media sources like Facebook can also be very noisy. This strictly prohibits directly using this data to train verification/recognition models. Also, the quality of photos are poor, resulting in weaker features, which would have otherwise allowed better matching.

An exhaustive scientific characterization of noise in social media is beyond the scope of this paper, and is being investigated step by step in social media research communities.

CPU Efficiency: The query engine in CueNet is responsible for extracting data from different sources. If a very large number of photos are being tagged, our scheme of query generation and merging will prove inefficient. Processing many photos from different people provides a very rich opportunity to develop interesting heuristics using event semantics for the multi-query optimization problem. Also, partitioning the discovery algorithm such that the computations can occur in a distributed manner is a complex problem. Such steps will be required if the application workload is of the scales of Facebook or processing photos in real time at the scales of Instagram.

Face Verification: Even though face recognition has been studied in research for the last two decades, face verification, and its specific application to faces in the wild has been a relatively recent venture. Although the accuracy of these systems is commendable, the problems of occlusion, image quality, face alignment and differing lighting conditions exist.

These hard problems need to be solved before "perfect" or "near-perfect" verification can be established.

Execution Patterns: When is a good time to execute the algorithm? When a user takes a photo? Or before she uploads it to her favorite photo sharing site? For the current evaluation, contextual sources are assumed to be immutable. This is not true in the real world. Contextual sources are constantly being appended with new information, and old information is being updated. These updates may be vital in tagging a certain photo. So the question of when to execute the algorithm, or how and when to query the sources is an open question. If a large number of photos are to be tagged, and a busy source like Facebook is being used for context, the CueNet query engine must take into account various freshness metrics and crawling policies of the sources.

Open Datasets: The unavailability of a large public data set over which different techniques can be evaluated against each other is an open problem. As seen in our experiments, personal information is vital to contextual approaches, and this data is largely personal, and therefore cannot be shared openly. Optimal anonymization techniques need be to invented such that the privacy of the experiment participants are maintained, and at the same time the data is meaningful to be applied in contextual approaches to problems. This need to be solved so that new context discovery techniques can be evaluated independently and against each other, over a common platform.

Chapter 6

Conclusion and Future Work

Problems: 1. Absence of a language/tool which can be used to express real world models. Ontologies are logic driven. Usually deal with true/false. 2. Starting points of such a language – physical variables, graphics world modeling, entity identification.

Bibliography

[1] http://en.wikipedia.org/wiki/Spanish_conquest_of_the_Inca_Empire.

[2] A. W. Q. G. Al-Zumor. A socio-cultural and linguistic analysis of yemeni arabic personal names. *GEMA: Online Journal of Language Studies*, 9(2):15–27, 2009.

[3] M. Ames and M. Naaman. Why we tag: motivations for annotation in mobile and online media. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2007.

[4] P. Barthelmess, E. Kaiser, and D. McGee. Toward content-aware multimodal tagging of personal photo collections. In *Proceedings of the 9th international conference on Multimodal interfaces*. ACM, 2007.

[5] M. Boutell and J. Luo. Beyond pixels: Exploiting camera metadata for photo classification. *Pattern recognition*, 38(6), 2005.

[6] L. Cao, J. Luo, and T. Huang. Annotating photo collections by label propagation according to multiple similarity cues. In *Proceeding of the 16th ACM international conference on Multimedia*. ACM, 2008.

[7] L. Cao, J. Luo, H. Kautz, and T. Huang. Annotating collections of photos using hierarchical event and scene models. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008.

[8] F. Capra. *The web of life: A new scientific understanding of living systems*. Anchor, 1997.

[9] G. Chen and D. Kotz. A survey of context-aware mobile computing research. 2000.

[10] Z. Chen, D. V. Kalashnikov, and S. Mehrotra. Exploiting context analysis for combining multiple entity resolution systems. In *Proceedings of the 35th SIGMOD international conference on Management of data*, pages 207–218. ACM, 2009.

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, CVPR 2005. IEEE Computer Society Conference on*, volume 1. IEEE.

[12] R. Datta, D. Joshi, J. Li, and J. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (CSUR)*, 40(2), 2008.

[13] A. K. Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.

[14] N. Diakopoulos and P. Chiu. Photoplay: A collocated collaborative photo tagging game on a horizontal display. *UIST Adjunct Proceedings*, 2007.

[15] J. M. Diamond and D. Ordunio. *Guns, germs, and steel*. Norton New York, 1997.

[16] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. Robbins. Stuff i've seen: a system for personal information retrieval and re-use. In *Proceedings of the 26th ACM SIGIR conference on Research and development in Informaion Retrieval*. ACM, 2003.

[17] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *Knowledge and Data Engineering, IEEE Transactions on*, 19(1), 2007.

[18] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[19] D. Frohlich, A. Kuchinsky, C. Pering, A. Don, and S. Ariss. Requirements for photoware. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*. ACM, 2002.

[20] J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong. Mylifebits: fulfilling the memex vision. In *Proceedings of the tenth ACM international conference on Multimedia*. ACM, 2002.

[21] A. Graham, H. Garcia-Molina, A. Paepcke, and T. Winograd. Time as essence for photo browsing through personal digital libraries. In *Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*. ACM, 2002.

[22] A. Gupta and R. Jain. Managing event information: Modeling, retrieval, and applications. *Synthesis Lectures on Data Management*, 3(4), 2011.

[23] J. Hailpern, N. Jitkoff, A. Warr, K. Karahalios, R. Sesek, and N. Shkrob. Youpivot: improving recall with contextual search. In *Proceedings of the 2011 annual conference on Human factors in computing systems*. ACM, 2011.

[24] A. Y. Halevy. Answering queries using views: A survey. *The VLDB JournalThe International Journal on Very Large Data Bases*, 10(4):270–294, 2001.

[25] J. Hays and A. Efros. Im2gps: estimating geographic information from a single image. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. Ieee, 2008.

[26] K. Henricksen, J. Indulska, and A. Rakotonirainy. Modeling context information in pervasive computing systems. In *Pervasive Computing*, pages 167–180. Springer, 2002.

[27] D. Henter, D. Borth, and A. Ulges. Tag suggestion on youtube by personalizing content-based auto-annotation. In *Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia*, pages 41–46. ACM, 2012.

[28] R. Jain and P. Sinha. Content without context is meaningless. In *Proceedings of the international conference on Multimedia*. ACM, 2010.

[29] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Describable visual attributes for face verification and image search. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, October 2011.

[30] K.-F. Lee. Context-dependent phonetic hidden markov models for speaker-independent continuous speech recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 38(4):599–609, 1990.

[31] X. Li, C. G. Snoek, M. Worring, and A. W. Smeulders. Fusing concept detection and geo context for visual search. In *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, page 4. ACM, 2012.

[32] L. Margulis and J. E. Lovelock. Biological modulation of the earth's atmosphere. *Icarus*, 21(4):471–489, 1974.

[33] D. L. McGuinness, F. Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(2004-03):10, 2004.

[34] F. Monaghan and D. O'Sullivan. Automating photo annotation using services and ontologies. In *Mobile Data Management, 2006. MDM 2006. 7th International Conference on*. IEEE, 2006.

[35] M. Naaman. *Leveraging geo-referenced digital photographs*. PhD thesis, Citeseer, 2005.

[36] M. Naaman, R. Yeh, H. Garcia-Molina, and A. Paepcke. Leveraging context to resolve identity in photo albums. In *Digital Libraries, 2005. JCDL'05. Proceedings of the 5th ACM/IEEE-CS Joint Conference on*. IEEE, 2005.

[37] B. Nowack. Confoto: Browsing and annotating conference photos on the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(4), 2006.

[38] N. O'Hare and A. Smeaton. Context-aware person identification in personal photo collections. *Multimedia, IEEE Transactions on*, 11(2), 2009.

[39] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[40] T. Rattenbury and M. Naaman. Methods for extracting place semantics from flickr tags. *ACM Transactions on the Web (TWEB)*, 3(1), 2009.

[41] E. Sadikov, M. Medina, J. Leskovec, and H. Garcia-Molina. Correcting for missing data in information cascades. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 55–64. ACM, 2011.

[42] N. Sawant, J. Li, and J. Wang. Automatic image semantic interpretation using social action and tagging data. *Multimedia Tools and Applications*, 2011.

[43] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*, pages 85–90. IEEE, 1994.

[44] M. Schneider. What's in my name? toward a generative anthroponomastics. *Anthropoetics*, 15(1):1–9, 2009.

[45] Z. Stone, T. Zickler, and T. Darrell. Autotagging facebook: Social network context improves photo annotation. In *Computer Vision and Pattern Recognition Workshops, CVPRW'08*. IEEE, 2008.

[46] N. N. Taleb. *The black swan: The impact of the highly improbable*. Random House Trade Paperbacks, 2010.

[47] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1), 1991.

[48] V. Vieira, P. Tedesco, and A. C. Salgado. Designing context-sensitive systems: An integrated approach. *Expert Systems with Applications*, 38(2):1119–1138, 2011.

[49] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE, 2010.

[50] L. Zhang, D. V. Kalashnikov, and S. Mehrotra. A unified framework for context assisted face clustering. 2013.

Appendix A

Appendix

Supplementary material goes here.

A.1 Source Mapping

```
(:source google-calendar
   (:attrs event email time location title description attendee)
   (:rel ev type-of event)
   (:rel owner type-of person)
   (:rel ti type-of time-interval)
   (:rel ev occurs-during ti)
   (:rel participant type-of person)
   (:rel owner participant-of ev)
   (:rel participant participant-of ev)
   (:io disk (:db mongodb))
   (:type personal)
```

```
   (:axioms
      (:map ev event)
      (:map ti time)
      (:map owner.email email B)
      (:map ev.occurs-during time)
      (:map ev.occurs-at location)
      (:map participant attendee)
      (:map ev.title title U)
      (:map ev.description description U)))

(:source fb-user
   (:attrs id name birthday location work email)
   (:rel person type-of person)
   (:rel named-place type-of named-place)
   (:rel address type-of address)
   (:rel person works-at named-place)
   (:rel person lives-at address)
   (:io disk (:db mongodb))
   (:type personal)
   (:axioms
      (:map person.name name)
      (:map person.dob birthday)
      (:map address.street-address location.name)
      (:map named-place.name work.name)))


(:source email
```

```
   (:attrs from to cc)
   (:rel pf type-of person)
   (:rel pt type-of person)
   (:rel pc type-of person)
   (:io disk (:db mongodb))
   (:type personal)
   (:axioms
      (:map pf.email from)
      (:map pt.email to)
      (:map pc.email cc)))


(:source fb-relation
   (:attrs name1 name2)
   (:rel p1 type-of person)
   (:rel p2 type-of person)
   (:rel p1 knows p2)
   (:io disk (:db mongodb))
   (:type personal)
   (:axioms
      (:map p1.name name1 F)
      (:map p2.name name2 U)))

(:source academix
   (:attrs name1 name2)
   (:rel p1 type-of person)
   (:rel p2 type-of person)
```

```
   (:rel p1 knows p2)
   (:io disk (:db mongodb))
   (:type public)
   (:axioms
      (:map p1.name name1 F)
      (:map p2.name name2 U)))

(:source conferences
   (:attrs time location ltitle stitle url)
   (:rel conf type-of event)
   (:rel time type-of time-interval)
   (:rel loc type-of location)
   (:rel conf occurs-at location)
   (:rel conf occurs-during time)
   (:axioms
      (:map time time)
      (:map loc location)
      (:map conf.title ltitle)
      (:map conf.name stitle)
      (:map conf.url url)))


(:source confattendees
   (:attrs url name time location ltitle stitle)
   (:rel conf type-of conference)
   (:rel time type-of time-interval)
   (:rel loc type-of location)
```

```
   (:rel attendee type-of person)
   (:rel attendee participant-in conf)
   (:rel conf occurs-at location)
   (:rel conf occurs-during time)
   (:axioms
      (:map time time)
      (:map loc location)
      (:map conf.title ltitle)
      (:map conf.name stitle)
      (:map conf.url url)
      (:map attendee.name name)))

(:source keynotes
   (:attrs url time location title name)
   (:rel conf type-of conference)
   (:rel k type-of keynote)
   (:rel k subevent-of conf)
   (:rel attendee participant-in k)
   (:axioms
      (:map conf.url url)
      (:map attendee.name name)
      (:map k.location location)
      (:map k.time time)
      (:map k.title title)))

(:source sessions
```

```
   (:attrs url time location title name)
   (:rel conf type-of conference)
   (:rel k type-of session)
   (:rel k subevent-of conf)
   (:rel attendee participant-in k)
   (:axioms
      (:map conf.url url)
      (:map attendee.name name)
      (:map k.location location)
      (:map k.time time)
      (:map k.title title)))

(:source talks
   (:attrs url time location title name)
   (:rel conf type-of conference)
   (:rel k type-of talk)
   (:rel k subevent-of conf)
   (:rel attendee participant-in k)
   (:axioms
      (:map conf.url url)
      (:map attendee.name name)
      (:map k.location location)
      (:map k.time time)
      (:map k.title title)))

(:source conflunches
```

```
   (:attrs url time location title name)
   (:rel conf type-of conference)
   (:rel k type-of lunch)
   (:rel k subevent-of conf)
   (:rel attendee participant-in k)
   (:axioms
      (:map conf.url url)
      (:map attendee.name name)
      (:map k.location location)
      (:map k.time time)
      (:map k.title title)))


(:source tweets
   (:attrs url name)
   (:rel conf type-of conference)
   (:rel attendee type-of person)
   (:rel attendee participant-in conf)
   (:axioms
      (:map conf.url url)
      (:map attendee.name name)))

(:source fb-events
   (:attrs event name time)
   (:rel ev type-of event)
   (:rel p1 type-of person)
   (:rel ti type-of time-interval)
```

```
(:rel ev occurs-during ti)
(:rel p1 participant-of ev)
(:axioms
   (:map p1.name name)
   (:map ev event)
   (:map ev.occurs-during time)
   (:map ti time)))
```