Structured Knowledge Extraction from Events on Twitter

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science (by Research)
in
Computer Science and Engineering

by

Sandeep Panem
201207659
sandeep.panem@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2015

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Structured Knowledge Extraction from Events on Twitter" by Sandeep Panem, has been carried out under my supervision and is not submitted elsewhere for a degree.

19^{th} June 2015

Date

Adviser: Prof. Vasudeva Varma

To my Friends

Acknowledgments

First and foremost, I wish to thank Prof. Vasudeva Varma for being my advisor and guide and Dr. Manish Gupta for being as my co-guide to my research work. Their presence gave me support, advice gave me direction and their belief gave me motivation to pursue research with utmost dedication.

I thank Kushal Dave for advising me at my initial stages of research work. I thank Ajay Dubey, Kushal Dave, Jayanth Gupta, Aditya Mogadala, Dharmesh Kakadia, Priya Radhakrishnan and Nikhil Priyatam for giving me valuable feedback and guidance during the most critical times of my research. I thank Manish Gupta who helped me to develop my writing skills. I thank Romil Bansal with whom I worked and had a great learning experience during my research.

I thank my batchmates for having spent some great times with them during the course of my stay at IIIT. I thank all the members of SIEL lab especially, Ajay Dubey and Romil Bansal for making the period of research joyous and satisfactory.

Finally I thank my parents for being there in my support and having faith in me during my research. They gave me freedom and stood by my decisions. In the end their patience with me helped to give justice and quality to my research work.

Abstract

In the recent past, the Web has not only become the predominant source of information, but also a crucial player in the evolution of events. People reporting information about real time events has made it a significant indicator to detect and estimate the pulse of various communities. Currently the most amount of real-time information about events is generated by social networks like Blogs, Twitter and Facebook. In view of this, it is very important to have a system that can condense the data and can extract meaningful information without human intervention. A new area of research in information retrieval (IR) has developed over the past few years called Topic Detection and Tracking (TDT). Topic detection involves detecting the occurrence of a new event such as earthquake, election results, a new movie released, or a political scam in a stream of news stories from multiple sources. Topic tracking is the process of monitoring a stream of news stories to find those that report or discuss the same event as one specified by a user. Previous approaches for Topic detection mainly focused on sources like news streams, web pages, documents, reviews and other structured and semi-structured information sources.

In this thesis, we focus on detecting the underlying semantic aspects (Sub-topics) of an event and extract structured information from the sub-topics found. Topics or events on Twitter relate to major events in the real world; sub-topics on the other hand are fine-grained aspects of such events. The velocity, volume and variety with which Twitter generates text is increasing exponentially. Mining sub-topics from entities helps in trend analysis, social monitoring, topic tracking and reputation mining. In the recent past, Twitter has been widely used for spreading the social pulse about real world entities. Fresh latent sub-topics identified from Twitter feeds at any given point of time could be extremely useful in providing better topic-wise search results relevant to users' informational needs. Also, extraction of Structured information from tweets related to a sub-topic could provide a nice summary for the sub-topic. The two main challenges in mining sub-topics from tweets in real-time are (1) understanding the semantic and the conceptual representation of the tweets, and (2) the ability to determine when a new sub-topic (or cluster) appears in the tweet stream. We address these challenges by proposing two unsupervised clustering approaches. In the first approach (SSR), for each tweet, we generate a semantic space representation by keyword expansion and keyphrase identication. In the second approach (CSR), we transform each tweet into a conceptual space that represents the latent concepts of the tweet. For our evaluation, we use RepLab (CLEF) 2013 dataset. The dataset contains tweets for 61 entities. Each entity has about 700 tweets for training and 1500 tweets for testing. The SSR approach gave an F1 measure of 0.259 and CSR an F1 measure of 0.339. We show a significant increase in the F1 measure

value by 16.9% as compared to the Baseline and $\sim$1% as compared to the best system in RepLab 2013, which indicates that the proposed CSR approach performs better than the state-of-the-art methods.

In extension to the previous approaches, we propose a more sophisticated approach which also considers semantic similarity between concepts and a statistical way to determine a threshold for creating a new sub-topic. We initially try to detect Sub-topics from targetted entities and later try to extract structured information from them. We generate a rich semantic representation of tweets using external knowledge bases, and combine it with relevant similarity metrics to train a sub-topic detection classifier. Further, we train a Conditional Random Field (CRF) model to identify subject-predicate-object fact triplets from tweets. Comparisons with various state-of-the-art mechanisms for sub-topic detection on a dataset of $\sim$134K tweets show that the proposed approach outperforms others by a significant margin providing a best F1 score of 0.417. The proposed CRF-based fact triplets extraction approach also beats the baselines significantly providing an overall F1 of 0.43.

We also try to map the extracted information to manually generated event schemas. For this task we specifically focus on the natural disaster events on twitter. As soon as natural disaster events happen, users are eager to know more about them. However, search engines currently provide a ten blue links interface for queries related to such events. Relevance of results for such queries can be significantly improved if users are shown a structured summary of the fresh events related to such queries. This would not just reduce the number of user clicks to get the relevant information but would also help users get updated with more fine grained attribute-level information. Twitter is a great source that can be exploited for obtaining such fine-grained structured information for fresh natural disaster events. Such events are often reported on Twitter much earlier than on other news media. However, extracting such structured information from tweets is challenging because: 1. tweets are noisy and ambiguous; 2. there is no well defined schema for various types of natural disaster events; 3. it is not trivial to extract attribute-value pairs and facts from unstructured text; and 4. it is difficult to find good mappings between extracted attributes and attributes in the event schema. We propose algorithms to extract attribute-value pairs, and also devise novel mechanisms to map such pairs to manually generated schemas for natural disaster events. Besides the tweet text, we also leverage text from URL links in the tweets to fill such schemas. Our schemas are temporal in nature and the values are updated whenever fresh information flows in from human sensors on Twitter. We selected five natural disaster event types: earthquakes, hurricanes (or typhoons), floods, wildfires and landslides. Evaluation on $\sim$58000 tweets for 20 events shows that our system can fill such event schemas with an F1 of $\sim$0.6.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the advent of online media and the social web, monitoring online reputation has become essential for any organization. Companies are keen on information spreading through word of mouth in social networks. Tracking their reputation in social media is an important task for assessing overall sentiment across people about their products. Proactive alerts can be sent, based on severity of the topics that are critical to company's reputation. Twitter uses short text of 140 characters and special vocabulary with incomplete sentences, heavy use of abbreviations and mis-spelled words, which makes traditional text analysis methods less effective. Online Reputation Management consists of monitoring and handling the opinion of Internet users (also referred to as electronic word of mouth, eWOM) on celebrities, companies and products, and is already a fundamental tool in corporate communication. The vast use of social media to share facts and opinions about entities, such as companies, brands and public figures has generated the opportunity and the necessity of managing the online reputation of those entities. Recently Topic Modelling has gained attention for discovering topics in a large collection of documents. Latent Dirichlet Allocation [10] is such an example which discovers the latent topics from text documents. Similarly Twitter-LDA is another version of LDA to handle tweets, and discovers topics from tweets. Topic modelling approaches like LDA, however takes only the co-occurrence information into account available from the data, but the semantic dependencies and the sub-topic structure are not handled by LDA.

Recently there has been a lot of work on event detection from Twitter [30]. A summary of detected events is useful for a large number of applications like forest-fires tracking, sporting events detection, finding local festivals, detecting drug related adverse events, traffic events, epidemics, earthquakes, emerging controversial events, etc. Further, mining sub-topics from such Twitter events can help in trend analysis, social monitoring, topic tracking and reputation mining. "Topics" or events on Twitter relate to major events in the real world; "sub-topics" on the other hand are fine-grained aspects of such events. A large collection of research work has focused on extracting event summary beyond merely discovering a set of entities representing the event. This includes works on finding the best phrase to summarize an event [48, 66], finding event types [61], finding event timespans [48], finding representative tweets for an event [46], finding event location [16, 40], computing event credibility [31], and extracting structured

infoboxes for events [56]. Another way of summarizing events is by extracting subject-predicate-object fact triplets. In this thesis, we focus on two tasks: extracting high quality sub-topics from Twitter and then finding Structured information from tweets belonging to these sub-topics.

1.1 Sub-topic detection on Targeted Entities

1.1.1 Problem Description

The velocity, volume and variety with which Twitter generates text is increasing exponentially. In the recent past, Twitter has been widely used for spreading the social pulse about real world entities. Twitter has become a popular social media platform where individuals share their views on various entities like celebrities, organisation, social events, sports, economic, cultural, geographical, historical etc. Users also report news, expressing emotions in reaction to a particular event. Even before the news media has the knowledge of what could be the emerging topics, twitter can deliver the newly developing topics across the world and report to the web community in real time. It is critical to determine latent sub-topics from such tweet data at any given point of time for providing better topic-wise search results relevant to users' informational needs. Sub-topic detection on Targeted entities is to cluster related tweets about the entity by topic with the objective of grouping together tweets referring to the same subject/event/conversation. "Topics" on Twitter relate to major events in the real world; "sub-topics" on the other hand are fine-grained aspects of such events. For example, consider the tweet, "Recently listed on MLS: 2003 Volvo VHD64B200 #mixer from Transport Truck Sales in Kansas City, KS". Here the sub-topic is "buying or selling of trucks" and the topic is "Volvo". Sub-topic structuring helps in categorizing the tweets related to an entity. The two tasks involved are 1. Filtering of tweets related to particular entity. 2. Identifying sub-topics per entity.

1.1.2 Filtering

Filtering tweets relevant to a given entity is an important task for online reputation management systems. A major problem in monitoring the online reputation of companies, brands, and other entities is that entity names are often ambiguous (apple may refer to the company, the fruit, the singer, etc.). The problem is particularly hard in microblogging services such as Twitter, where texts are very short and there is little context to disambiguate. We address the filtering task of determining, out of a set of tweets that contain an entity name, which ones do refer to the entity. A major problem concerning the above task (retrieval of potential mentions) is that entity names are often ambiguous. For instance, the query Ford retrieves information about the motor company , but also might retrieve results about Ford Models (the modeling agency), Tom Ford (the film director), etc. There are two challenges that needs to be addressed:

- Multiple senses of a term: “apple may relate to a company or a fruit. In the same way Stanford may refer to a place or a university.
- Multiple aliases related to entity: For instance “Blue Screen of Death is referred to Microsoft windows.

1.1.3 Challenges

Sub-topic detection is challenging because (1) the number of sub-topics is unknown beforehand; and (2) many tweets containing different words could mean the same and so should belong to the same sub-topic. But how to build a semantic representation of the tweets? How to identify if a new tweet is part of an already observed sub-topic or marks the beginning of a new sub-topic? How to leverage various data-driven ways of establishing similarity for better sub-topic identification?

1.1.4 Overview of our approach:

In this thesis, we assume that we have tweets for an event related to a target entity. To identify sub-topics from such tweets, we use external semantic knowledge bases and annotation mechanisms to obtain a semantic and concept representation of the tweet. We first propose two unsupervised clustering approaches. In the first approach, we generate a semantic space representation for each tweet by keyword expansion and keyphrase identification. In the second approach, we transform each tweet into a conceptual space that represents the latent concepts of the tweet. Later we also take the similairty between two concepts into consideration. We use this semantic representation of the tweets along with a classifier to decide whether a tweet belongs to an existing sub-topic or should be assigned to a new sub-topic. Various similarity measures between different semantic representations of the tweet and the sub-topics are used as features. We also statistically determine a threshold of when to create a new sub-topic.

1.2 Extracting Structured Information from Events

1.2.1 Problem Description

As soon as real world events occur on twitter, users are eager to know more about them. Often times they look out for related facts. For example in case of natural disaster events, what is the severity of the storm or the magnitude of the earthquake etc. Searchers are also interested in knowing about the damage caused by these natural calamities, e.g., number of people dead or number of homes destroyed. In 2012, there were 905 natural catastrophes worldwide, 93% of which were weather-related disasters. Overall costs were US $170 billion and insured losses $70 billion. 45% were meteorological (storms), 36% were hydrological (floods), 12% were climatological (heat waves, cold waves, droughts, wildfires)

and 7% were geophysical events (earthquakes and volcanic eruptions). Between 1980 and 2011 geophysical events accounted for 14% of all natural catastrophes [43]. Search engines currently provide a ten blue links interface for queries related to such events. Relevance of results for such queries can be significantly improved if users are shown a structured summary of the fresh event related to the query. This would not just reduce the number of user clicks to get the relevant information but would also help users get updated with more fine grained attribute-level information. This is especially useful in cases of disaster events because it can help users obtain information quickly and thereby help in reducing anxiety levels. To show a structured summary of such events, one needs to obtain fresh information about such events. News media, blogs, Twitter are all good sources of information. However, for natural disasters, information is found fastest on Twitter [77].

1.2.2 Challenges

Twitter is a great source that can be exploited for obtaining such fine-grained structured information for fresh natural disaster events. Such events are often reported on Twitter much earlier than on other news media. First, one needs to identify all tweets related to the query disaster event. Next, these tweets need to be linguistically analyzed to extract useful structured information. However, extracting such structured information from tweets is challenging because: 1. tweets are noisy and ambiguous; 2. there is no well defined event schema for various types of natural disaster events; 3. it is not trivial to extract semantic attribute-value pairs and facts from unstructured text; and 4. it is difficult to find good mappings between extracted attributes and standard attributes in the event schema. Hence, extracting useful information from tweets is challenging.

1.2.3 Fact-Triplet Extraction

A fact triplet consists of three main parts: Subject, Predicate and the Object. For example consider the tweet Volvo Ocean Race set for raft of changes to boats, teams and route in bid to appease sailors and sponsors via @Telgraph http://soc.li/AenbU9M. The extracted fact triplet for this tweet is (Volvo Ocean Race : appease: sailors, sponsors). The task of extraction of fact triplets from tweets is challenging because (1) tweets are noisy and ambiguous; (2) tweets do not follow any formal grammatical structure; (3) even for a human it is quite difficult to identify the subject, predicate and object; (4) there is no publicly available benchmark dataset to evaluate the quality of fact triplet extraction. For extraction of fact triplets, we use multiple linguistic analysis signals to learn a Conditional Random Field [38] (CRF) model.

1.2.4 Overview of our approach

We deal with a few of the above challenges as follows. We designed novel algorithms for extraction of both numeric as well as textual attribute-value pairs from tweets. We also provide a novel algorithm

for extracting fact triplets from tweets. Next, we generate schemas for five different event types by leveraging Wikipedia Infoboxes along with some manual efforts. Finally, we present a novel algorithm to map extracted information to standard structured fields in the event schemas. Besides the tweet text, we also leverage text from URL links in the tweets to fill such schemas. Our schemas are temporal in nature and the values are updated whenever fresh information flows in from human sensors on Twitter. To the best of our knowledge, the proposed system is the first to focus on extraction of structured event Infoboxes from Twitter for natural calamity events.

1.3 Contributions of this Thesis

- We design interesting ways of semantically representing tweets, and further combine them with relevant similarity metrics to define features for the sub-topic detection classifier.
- We exploit various linguistic analysis techniques like Part-Of-Speech (POS) tagging, dependency analysis, etc. to learn a CRF model for extraction of fact triplets.
- We generate a labeled dataset for evaluation of the subject-predicate-object fact triplets extraction task. We make the dataset available publicly [1].
- We propose the problem of automatically generating structured Infoboxes for events from social media.
- Specifically, we consider natural disaster events and propose novel algorithms for extracting attribute-value pairs and facts.
- Through experiments on $\sim$58000 tweets related to natural disaster events of five different types, we show the effectiveness of the proposed system.
- We perform extensive experiments to show the effectiveness of our methods and comparisons with state-of-the-art mechanisms.

1.4 Thesis Organization

Chapter 2 presents literature survey on Topic Modelling, Sub-topic detection and extracting structured information from those Sub-topics. We present related systems for attribute-value extraction and fact-triplet extraction tasks. It also discusses about the evaluation measures in the context of sub-topic detection and extracting structured information from tweets.

Chapter 3 describes Entity tracking using Sub-Topic detection in latent semantic and concept spaces. It first explains about the Semantic space representation and Concept space representation approaches. It also compares our system to previous state-of-the-art systems and results show that our system is effective.

[1] `http://tinyurl.com/factTripletExtractionData`

Chapter 4 describes Structured Information extraction from Natural disaster events on twitter. It discusses about Numeric Attribute-value extraction, Text Attribute-value extraction and Fact Triplet extraction. In Section 4.2, we present an introduction to Stanford dependencies and then present our novel algorithms for extracting both numeric and textual attribute-value pairs. In Section 4.3, we present a novel algorithm for extracting fact triplets. In Section 4.4, we first discuss event schema generation and then provide a novel mechanism to map attribute-value pairs extracted in Section 4.2 to event schemas. We present results of our experiments on 20 events in Section 5.3 and a temporal analysis done on a case study.

Chapter 5 describes Extraction of Fact Triplets from Sub-topics on Twitter. It explores on the similarity between concepts and statistically determining threshold for creation of a new sub-topic. In Section 5.1, we describe the process of detecting sub-topics from tweets for an event related to a target entity. In Section 5.2, we describe the process of extracting facts from the detected sub-topics. Section 5.3 describes the datasets, results and comparisons with other systems.

Chapter 6 concludes the thesis explaining the work done and describing the results of the experiments. It also provides the details of future work with respect to the thesis.

Chapter 2

Related Work

2.1 Sub-Topic detection

The problem of identifying topic-based clusters has been modeled in various ways. Unlike normal documents, these text & web segments are usually noisier, less topic-focused, and much shorter, that is, they consist of from a dozen words to a few sentences. Because of the short length, they do not provide enough word cooccurrence or shared context for a good similarity measure. Therefore, normal machine learning methods usually fail to achieve desire accuracy due to the data sparseness. Existing topic detection methodologies are generally based on probabilistic language models, such as Probabilistic Latent Semantic Analysis (PLSA) [33] and Latent Dirichlet Allocation (LDA) [11]. LDA cannot find the fine grained sub-topics accurately, because it sees only the co-occurrence information within the corpus, and the tweet corpus is rather small. Also the number of clusters need to be pre-defined.

Qi He[32] used the busty vector space models (B-VSM) to cluster the news streams into topical groups. He also proposed an unsupervised greedy event detection algorithm to detect both periodic and aperiodic events. Mario Cataldi[12] formed a navigable topic graph which connects the emerging terms using aging theory to form the real-time clusters from the live stream. Shuangyong, Qiudan Li, [67] first detected query topic's burst periods, then discover its cobursting relationship with other topics. Then they rank topics' associate degree with a spatio-temporal similarity computing method. UNED[5] experimented with three approaches. (i) Twitter-LDA approach, which is a variant of LDA adapted to the characteristics of Twitter (ii) Agglomerative clustering based on term co-occurrences to identify the terminology of each topic, clustering the terms occurring in the input entity stream of tweets and assigning tweets to the identified clusters. (iii) The third approach was based on clustering wikified tweets. It relied on the hypothesis that tweets sharing concepts defined in a knowledge base such as Wikipedia are more likely to belong to the same cluster than tweets with none or less concepts in common. Few limitations of these approaches are (1) they often mix multiple incoherent sub-topics together, and (2) they cannot find novel topics in streaming scenarios as they need all the data at once. The topics that are identified by Twitter-LDA are not accurate and are more generic and assigns a single topic for entire tweet where it can relate to multiple topics.

REINA at CLEF 2013 [6] used similarity matrix and community detection techniques for topic detection. By exploiting tweet contents, LIA at CLEF 2013 [6] applied a large variety of machine learning methods for clustering of tweets. However, these methods require the number of topics as an input, and assume that a single document contains rich information, which is not applicable to microblogs.

2.1.1 Topic Detection on Microblogs

As tweets are short and provide less context, clustering them just based on lexical anaysis results in incorrect topics. Tan xu and Douglas[75] used wikipedia as a semantic resource to cluster Microblog posts. They leveraged Wikipedia's link structure to calculate semantic similarity between tweet terms. In addition to that, we also extract various lexical and semantic features like, the content of the web page pointed by the url in the tweet and extracting Key phrases from the tweet. Weili Xu, et al. (2012)[76] applied a frequent pattern mining approach to extract topics from microblog posts. They extract topical keywords and represent them as transaction items and apply Apriori algorithm to extract topics. Qiming Diao, et al. [20] find bursty topics from microblogs. They propose a topic modelling approach, which also considers temporal information, bursty patterns and seggregates "personal" posts from event-driven posts. Kriti, et al. (2010)[58] extracted topics using standard topic modelling on user's tweets. They considered a single user's tweets into a single document. They used these latent topics to find social links between. Matthew, et al. (2010)[49] first detect entities present in the tweet and used a knowledge base to disambiguate the entities. They later find categories or topics which cover most entities. Yanyan Du, et al.(2011)[22] propose a bursty topic detection technique which calculates term weight by using user weight, number of listeners, replies and collections into account. User weight is calculated by PageRank algorithm.

2.2 Structured Information Extraction

There has been a lot of work on extracting structured content from news articles, Wikipedia pages, queries and general web documents. Also there has been work on extracting structured content from tweets. While our work also leverages linguistic analysis similar in philosophy to other previous works, our work focuses on extracting attribute-value pairs from tweets and map them to standard event schemas which has not been done earlier.

2.2.1 Extracting Structured Content from the Web

Sarawagi [65] provides a great survey on automatic extraction of information from unstructured sources. We describe a few other works here. Nakashole et al. [52] developed a system 'PATTY' which is a large resource for textual patterns that denote binary relations between entities. Fader et al. [24] developed a system 'REVERB' which is based on syntactic and lexical constraints on binary relations

expressed by verbs from English sentences. Wu et al. [72] developed a system 'Kylin Ontology Generator' which extracts structured data from Wikipedia raw texts and builds an ontology by combining Wikipedia Infoboxes with Wordnet using statistical relational learning. Wikipedia text follows a good grammatical structure and is easy to extract structured info than compared with noisy and short text information contained in tweets.

Rusu et al. [62] extracted triplets from general English sentences using Tree-bank and link grammar parsers. They have proposed an algorithm to extract subject, object and predicate for grammatically correct English sentences, which fail to work on tweets. Bellare et al. [9] proposed a lightly supervised method to extract attributes from different entities from natural language corpus like Web. They trained on a fixed entities like company, country, etc. and extracted attributes pertaining to that entities itself. Reisinger et al. [60] proposed a 'Bootstrapped Web-Search Extraction' to extract class attributes simultaneously from Web documents and query logs. But they do not map the attributes to an existing event schema, neither do they detect the event type. Wong et al. [71] proposed a methodology for extracting attribute-value pairs from web pages. In the first phase they generate candidate attributes and in second phase they do candidate filtering. Even in this case the attribute-values were not mapped to any event schema.

Enrique et al. [3] proposed a hierarchical topic model for automatically identifying syntactical and lexical patterns to detect relations from Web text. They leverage distant supervision using relations from knowledge base Freebase. Enrique et al. [4] presented a method for increasing the quality of automatically extracted instance attributes by exploiting weakly supervised and unsupervised instance relatedness data. The method organizes text derived data into graph and propogates attributes among related instances through random walks over the graph. Lee et al. [39] extract attributes for concepts and entities by integrating concept and instance based patterns into probabilistic typicality scores that scale to broad concept space. All these methods have been shown to perform well on Web text. But tweets are more noisy and much shorter than sentences on the Web, and hence these methods do not usually perform well on tweets.

2.2.2 Extracting Structured Content from Tweets

There is a large body of work on event detection from Twitter [29, 35, 46, 61]. But our proposed system goes much beyond simple event detection. Our focus is to display as much structured event content as possible. The closest to our work is the work by Marcus et al. [44] which focuses on providing an SQL-like interface to Twitter API. While their focus is on querying the event stream database, our focus is on populating such a database with highly structured data. TwiCal [61] extracts a 4-tuple representation of events which includes a named entity, event phrase, calendar date, and event type. It represents the events in a calendar format. It does not leverage the global context of tweets and does not focus on extracting attribute-values and facts from tweets but rather focuses on extracting 4-tuple representation. Abel et al. [1] inferred facets and facet values by enriching the semantics of tweets. They tried to tag person, location and organization etc. in the tweets which help in faceted search.

Their system does not focus on extracting attribute-value pairs and facts triplets which is the one of the main contributions of this thesis. Twitter has also been used to detect and track natural disaster events: locating wildfires [69], hurricanes, floods [68], earthquakes [64, 36] and tornados. In this thesis, we use such detected events as input and extract a drilled down structured summary.

2.2.3 Fact Triplet Extraction:

Reverb [24] extracts relations from the Web and Wikipedia documents using a logistic regression classifier. It imposes two constraints (1) a syntactic constraint, i.e., it eliminates incoherent extractions, it reduces uninformative extractions by capturing relation phrases expressed by a verb-noun combination) and (2) a lexical constraint based on the intuition that a valid relation phrase should take many distinct arguments in a large corpus. The resulting extractions are assigned a confidence score. TextRunner [8] also extracts relations from Web documents. TextRunner used a Naïve Bayes model with un-lexicalised POS and NP-chunk features, trained using examples heuristically generated from Penn Treebank. WOE [73] does an extension to TextRunner by also incorporating dependency-parse features. It does heuristic matches between Wikipedia Infobox attribute values and corresponding sentences to construct training data. It uses POS tags and dependency paths in the sentence to detect the relations. KnowItAll [23] does not require an NER and learns to label its own training examples to generate candidate facts using a small set of domain-independent extraction patterns. KnowItAll automatically tests the plausibility of the candidate facts it extracts using pointwise mutual information statistics computed from search engine hit-counts. However, KnowItAll requires large number of search engine queries and web page downloads. Rusu et al.'s technique [63] extracts subject-predicate-object triplets from English sentences. They use syntactical parsers for generating parse trees and then design rules on such parse trees to extract subject, predicate and object. All of these relation extraction systems have been built for general English sentences, Web documents and Wikipedia. As we discussed in Section 5.3, such models fail to work on the noisy Twitter text. We model the fact triplet extraction problem as a sequence tagging problem and design novel features for the same.

2.3 Evaluation measures

We use Replab 2013 Task Baselines and evaluation measures as our experiments are based on the dataset provided by task for Sub-topic detection.

2.3.1 Baselines

The baseline approach consists of tagging tweets (in the test set) with the same tags of the closer tweet in the (entity) training set according to the Jaccard word distance. The baseline is, therefore, a simple version of Memory-Based learning. The selection of this approach for several reasons: (i) It is easy to understand; (ii) It can be applied to every subtask in RepLab 2013; (iii) It keeps the coherence

between tasks: if a tweet is annotated as non-related, it will not receive any priority or topic tag. (iv) it exploits the training data set per entity.

2.3.2 Evaluation Measures

We employ combination of Reliability and Sensitivity (R&S) as a strict and theory grounded measure. Replab 2013 proposal consists of two complementary measures, Reliability and Sensitivity, with a straightforward initial definition [7]. Let us consider a system output X and a goldstandard $\mathcal{G}$, which are both a set of document relationships $r(d, d')$. The Reliability (R) of relationships in the system output is the probability of finding them in the gold standard. The Sensitivity (S) of predicted relationships is the probability of finding them in the system output when they appear in the gold standard. In other words, R and S are precision and recall of the predicted set of relationships with respect to the true set of relationships. In order to avoid the quadratic effect of document pairwise, R&S is computed for each document relationships and averaged in a second step.

$$R(\chi) \equiv P_{\chi}(r(d, d') \in G) \tag{2.1}$$

$$S(\chi) \equiv P_{\mathcal{G}}(r(d, d') \in \chi) \tag{2.2}$$

We can express Reliability as a sum of probabilities pondered by the probability of each relationship in X :

$$R(\chi) \equiv \sum_{r(d,d') \in \chi} P(r(d, d') \in \mathcal{G}) P(r(d, d')|\chi) \tag{2.3}$$

The second component represents the weight of relationships according to their priority (rank position) in the system output. This allows to define a discounting function for IR results. Given that $P(d|r(d, d)) = 1$ we can express R as:

$$R(\chi) \equiv \sum_{r(d,d') \in \chi} P(r(d, d') \in \mathcal{G}) P(r(d, d')|d, \chi) P(d|\chi) \tag{2.4}$$

We assume then that the probability of a relationship $r(d, d')$ given d is proportional to the probability of d' within the set of documents related with d. Then:

$$R(\chi) \equiv \sum_{r(d,d') \in \chi} P(r(d, d') \in \mathcal{G}) P(d'|r(d, d'), d, \chi) P(d|\chi) \tag{2.5}$$

Then, we can express R and S in terms of weights of single documents. Being $w_X(d)$ the weight of d in X and being Wx,d the sum weight of documents related with d:

$$W_{\chi,d} = \sum_{r(d,d') \in \chi} w_x(d') \tag{2.6}$$

We can compute R and S as:

$$R(\chi) = \sum_{r(d,d')\in\chi} P(r(d,d') \in \mathcal{G}) \frac{w_x(d')}{W_{x,d}} w_x(d) \tag{2.7}$$

$$S(\chi) = \sum_{r(d,d')\in\chi} P(r(d,d') \in \chi) \frac{w_g(d')}{W_{g,d}} w_g(d) \tag{2.8}$$

If all documents have the same weight in the distributions, R and S simply turn into the average R(d) and S(d) associated to each document:

$$R(\chi) = Avg_d P(r(d,d') \in \mathcal{G} | r(d,d') \in \chi) \tag{2.9}$$

$$S(\chi) = Avg_d P(r(d,d') \in \chi | r(d,d') \in \mathcal{G}) \tag{2.10}$$

Chapter 3

Entity Tracking in Real-Time using Sub-Topic Detection in latent Semantic and Concept Spaces

3.1 Filtering

In this chapter we focus on two problems namely Filtering and Sub-Topic detection. To better identify sub-topics and eliminate noise , filtering step is necessary. Filtering tweets relevant to a given entity is an important task for online reputation management systems. A major problem in monitoring the online reputation of companies, brands, and other entities is that entity names are often ambiguous (apple may refer to the company, the fruit, the singer, etc.). The problem is particularly hard in microblogging services such as Twitter, where texts are very short and there is little context to disambiguate. A major problem concerning the above task (retrieval of potential mentions) is that brand names are often ambiguous. For instance, the query 'Ford' retrieves information about the motor company ,but also might retrieve results about Ford Models (the modeling agency), Tom Ford (the film director), etc. There are two challenges that needs to be addressed:

1. *Multiple senses of a term:* "apple" may relate to a company or a fruit. In the same way "Stanford" may refer to a place or a university. In such cases the URL content along with tweet content was used in disambiguating the topic.

2. *Multiple aliases related to entity:* For instance "Blue Screen of Death" is referred to "Microsoft windows". This distinction should also be made while filtering.

3.1.1 Approach

We address the filtering task of determining, out of a set of tweets that contain a company name, which ones do refer to the company. To address this task we studied a set of features that can be generated to describe the relationship between an entity and a tweet. We explored different types of features: text, keyword similarity scores between entities metadata and tweets and semantic similarities like GCD (Google concept data) score etc. The task we are tackling consists in building a relevance

classifier: given an entity ei and a tweet tj we want to classify tj as Related or Unrelated to ei. We use a supervised learning approach to address this problem. In this section, we describe our approach which comprises pre-processing of raw tweets and selecting the most appropriate feature representation of the relationship between entities and tweets. The architecture is shown in figure 3.1.

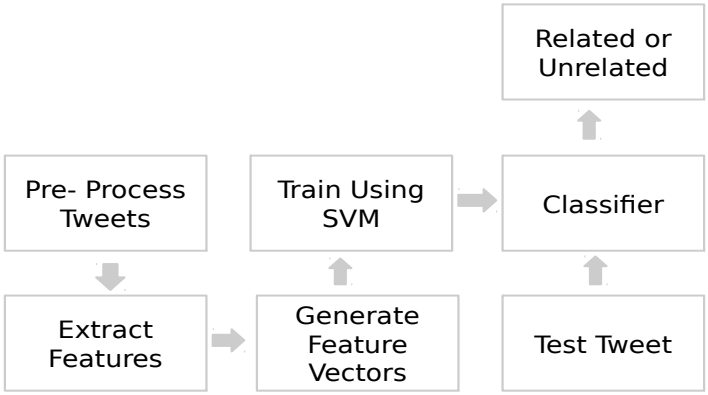

Figure 3.1 Filtering Architecture

3.1.1.1 Preprocessing

Contrary to other type of online texts (e.g. news or blog posts) tweets contain informal and non-standard language containing emoticons, spelling errors, wrong letter casing, unusual punctuation and abbreviations. Therefore, we apply some pre-processing techniques for text normalization.

1. We used CMU Pos tagger to extract keywords based on the following *(write here)* pos tags.
2. The hashtags are represented as normal words by replacing hash.
3. We extract URL links from tweet text and crawl the page content of the URL link.
4. We extract top 10 keywords from URL page content giving more weightages to the words that occur in title.
5. We remove user mentions and punctuation.

3.1.1.2 Features

1. **Tweet context similarity with entity using TF-IDF measure:** We train a classifier with features as TF-IDF scores of unigrams, bigrams and trigrams of tweet text in the training data. We use

probabilistic SVM to train the model which gives the probability of tweet text being related to the entity.

2. **GCD**[1] **similarity score:** The keywords from tweet 't' are taken and queried on GCD to get top 100 relevant wikipedia page titles. For each wikipedia page title obtained, we calculate the similarity between entity 'e' and page title using fuzzy string matching. If the similarity is greater than 0.6, we consider the wiki page being related to the entity. The GCD score is calculated as the ratio of number of wikipages found relevant to the entity and number of total wikipages retrieved.

$$score_{gcd}(e,t) = \frac{N_R}{N_T} \tag{3.1}$$

Here N_R is the number of wikipages found relevant to the entity and N_T is the total number of wikipages retrieved.

3. **similarity between Tweet text and Entity name:** We calculate Jaccard similarity between the tweet text (t_{text}) and entity name (e_n).

$$sim(t_{text}, e_n) = \frac{|t_w \cap e_w|}{|t_w \cup e_w|} \tag{3.2}$$

Here t_w refers to tweet words and e_w refers to entity words.

4. **Similarity between Tweet text and entitie's Home Page:** We extract keywords from entitie's home page (e_h) and from Tweet text (t_{text}). Then the similarity score is calculated as the ratio of number of matched keywords and total number of keywords in the entitie's home page.

$$sim(t_{text}, e_h) = \frac{|t_w \cap e_{hkw}|}{|e_{hkw}|} \tag{3.3}$$

Here e_{hkw} refers to entitie's home page keywords.

5. **similarity between Tweet text and entitie's Wiki page** : We extract keywords from entitie's Wikipedia page (e_{wp}) and from Tweet text (t_{text}). Then the similarity score is calculated as the ratio of number of matched keywords and total number of keywords in the entitie's Wiki page.

$$sim(t_{text}, e_{wp}) = \frac{|t_w \cap e_{wkw}|}{|e_{wkw}|} \tag{3.4}$$

Here e_{wkw} refers to entitie's wiki page keywords.

6. **similarity between tweet URL text and entitie's home page:** We extract keywords from entitie's home page (e_h) and from tweet URL text ($tURL_{text}$). Then the similarity score is calculated as the ratio of number of matched keywords and total number of keywords in the entitie's home page.

$$sim(tURL_{text}, e_h) = \frac{|tURL_w \cap e_{hkw}|}{|e_{hkw}|} \tag{3.5}$$

Here $tURL_{text}$ refers to tweet URL text keywords.

[1] **http://nlp.stanford.edu/pubs/crosswikis.pdf**

7. **similarity between tweet URL text and entitie's wiki page:** We extract keywords from entitie's wikipedia page (e_{wp}) and from tweet URL text ($tURL_{text}$). Then the similarity score is calculated as the ratio of number of matched keywords and total number of keywords in the entitie's wiki page.

$$sim(tURL_{text}, e_{wp}) = \frac{|tURL_w \cap e_{wkw}|}{|e_{wkw}|} \tag{3.6}$$

8. **semantic similarity of keywords between tweet text and entitie's wiki page and home page:** We calculate the wordnet similarity between tweet text keywords and Top 10 keywords of entitie's home page and wiki page.

3.1.2 Dataset

We use Replab(CLEF) 2013 dataset for evaluation. The dataset contains tweets about 61 entities. Each entity has about 700 as training data and 1500 tweets for testing. The training and test data sets are manually labelled by annotators who are trained and guided by experts in online reputation management. For each tweet in the corpus we have the target entity id, the language of the tweet, the timestamp and the tweet id. The content of each URL in the tweets is also provided. The Wikipedia page and home page of the entity are also included.

3.1.3 Evaluation and Results

We trained over each entity separately using libsvm[14]. Using the test data for each entity, we calculated the accuracy for entire dataset. The average accuracy on 61 entities is observed to be 80.27%.

3.2 Sub-Topic detection in Real Time

The two main challenges in mining sub-topics from tweets in real-time are (1) understanding the semantic and the conceptual representation of the tweets, and (2) the ability to determine when a new sub-topic (or cluster) appears in the tweet stream. We address these challenges by proposing two unsupervised clustering approaches. In the first approach, we generate a semantic space representation for each tweet by keyword expansion and keyphrase identification. In the second approach, we transform each tweet into a conceptual space that represents the latent concepts of the tweet. We empirically show that the proposed methods outperform the state-of-the-art methods. Our two phase clustering approach as described in Section 3.2.1 deals with the above mentioned issues by mining sub-topics from streaming text. The proposed clustering is based on a novel representation of tweets using various combinations of concepts, keyphrases and keywords with appropriate weights assigned to them. We improve the accuracy of detecting sub-topics by discarding less frequent concepts. Also, *batched* updates ensure that our system is efficient enough to be practical.

3.2.1 Approach

In this chapter, we propose the following two approaches to tackle the problem of dynamic clustering by exploiting the semantic structure of tweets.

1. Semantic Space Representation (*SSR*) based approach
2. Concept Space Representation (*CSR*) based approach

Both the approaches consist of two phases: an offline cluster generation phase and an online cluster maintenance phase. In the offline phase, a graph-based clustering algorithm is used to obtain the initial clusters from a few tweets. These initial clusters are later used in the online phase to cluster new tweets from the tweet stream and also to update the clusters themselves. Figures 3.2 and 3.3 illustrate the offline and online phases for the two approaches respectively.

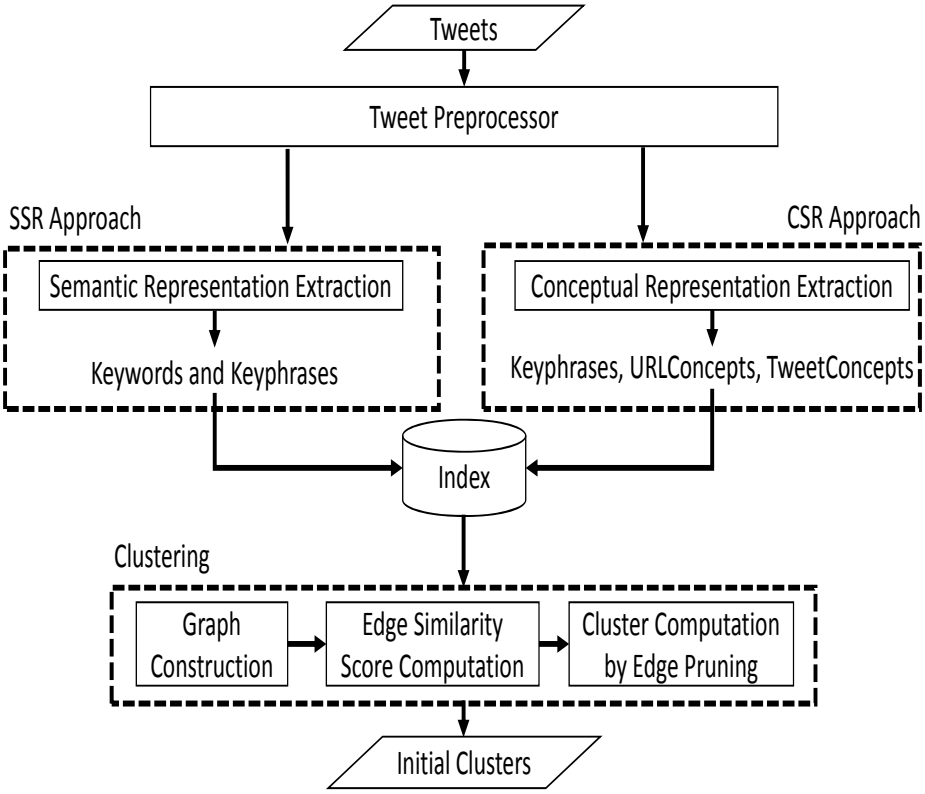

Figure 3.2 Offline Initial Cluster Generation Phase for Both Approaches

3.2.1.1 Semantic Space Representation (*SSR*) based Approach

In the offline phase, we first preprocess the tweets by removing stopwords and performing POS tagging and URL extraction. We consider only English tweets across multiple domains maintaining the

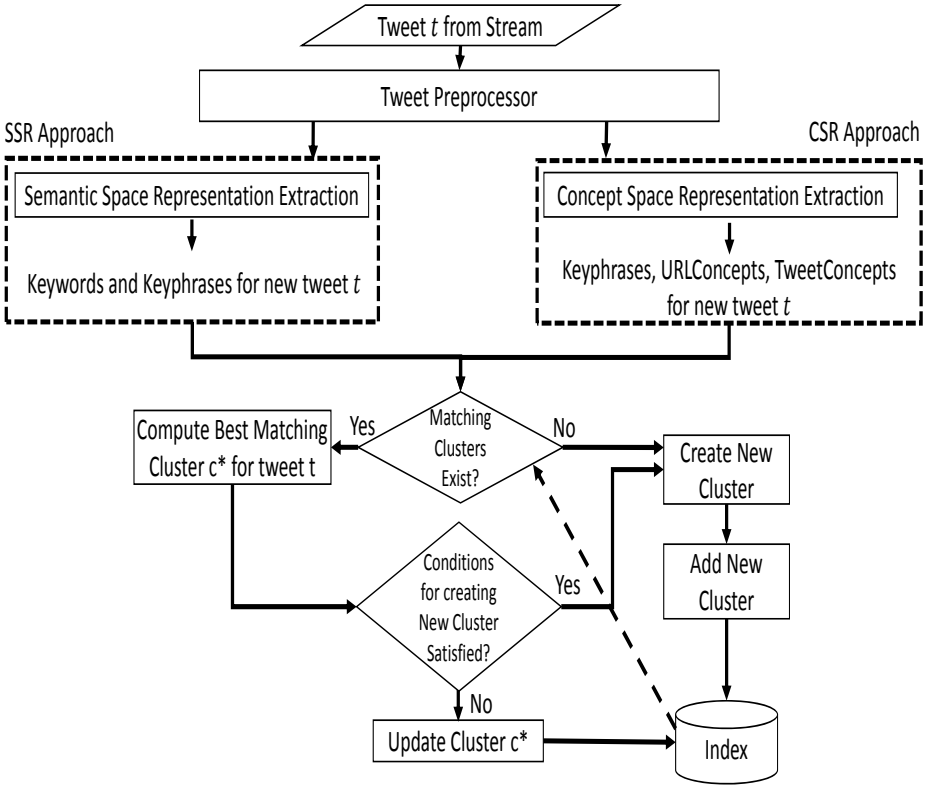

Figure 3.3 Online Cluster Maintenance Phase for Both Approaches

heterogeneity. We extract the longest sequence of nouns as well as proper nouns and keyphrases using POS chunking [17]. We consider nouns, hashtags and proper nouns as keywords and enhance them by finding the synonyms using WordNet and extracting top n words from the page content of the URL contained in the tweet.

3.2.1.1.1 Preprocessing and Keywords Extraction: Twitter being a noisy medium, multiple preprocessing steps have been performed. This includes POS tagging, stopword removal and URL extraction. Empty tweets and tweets of languages other than english were removed. CMU postagger was used to extract longest sequence of nouns as well as proper nouns. The tags "v, NN, NNP, NNS, NNPS, A, CD, #, S" were taken into consideration for extracting the nouns. If tweets does not contain any nouns other than the entity name, verbs and adverbs were considered. URL's were also extracted from the tweets and the URL content was used in determining which topic the corresponding tweet belongs to. Multiple occurrences of the letters in a word like "happppppppyyyyyy" are converted to atmost 2 occurrences i.e. "happyy". The tweets were indexed based on N Gram patterns present for each entity. For string matching, the words are stemmed and then matched as exact matching may give wrong results because of spelling mismatch. A general twitter specific stopword list has been created which were not indexed.

3.2.1.1.2 Finding related keywords: We use wordnet [50] for extracting synonyms related to a particular keyword. The keywords extracted were given input to wordnet and its noun synsets are taken and an enhanced keyword list is obtained. If there are no noun keywords in the tweet or there exists a single noun representing entity , for example, "*@WesRomaine EEEKK I want a Volvo soooo bad*" and "*@Ew4n: I think I need to buy a Volvo.*", where the only occuring keyword is the entity name '*Volvo*'. In this case it has to be in a separate cluster representing the '*Volvo customers*'. But because of sparse keywords we will not be able to distinguish. In this case the verbs in the tweet are taken into consideration along with the entity name. Referring to the previous example, '*want Volvo*' and '*need Volvo*' are grouped into same cluster '*Volvo customers*'. Similarly the word '*sale*' and '*purchase*' in the tweets "*Volvo #XC90 to Powerlease Car Sales*" and "*New SUV Added: 2006 Volvo XC90: We've just added a SUV to our inventory and it is available for purchase now! ...*" are grouped into same cluster as they are antonym of each other.

3.2.1.1.3 Keywords expansion using URL Content: The URL mentioned in the tweets has useful information and helps us to better classify the tweet by identifying keywords which would determine its similar cluster. The URL is parsed using HTML parser and keywords are extracted from title, metatags, meta-keywords and description. Using POS tagger only nouns and proper nouns are extracted.

3.2.1.1.4 Keyphrase Extraction through POS chunking: The basic N-Gram chunking could not determine the exact terms or tags related to the company because most of the topics are mixture of noun tags combined with adjectives and conjunctions. Consider the tweet "RT @9to5Google: Watch the Samsung Galaxy S4 unveil live stream here [Video] http://t.co/ntt78BPVoz". When N-Gram chunking is done on the following tweet it gives irrelevant product names or tags such as watch, Samsung Galaxy S4, unveil, Samsung Galaxy s4 unveil, Samsung galaxy, s4 unveil, stream, etc.

So totally 12 candidate keywords are formed with N-Gram chunking out of which only 3 are representable as keywords for topics. POS chunking[1] on the other hand finds only the potential keywords which are representable as topics. POS chunking was able to find the exact keyphrases in the tweet which is not possible with N-Gram chunking.
Following are some of the patterns followed by topics:

- (Noun)+ : For topics containing one or more noun tags. For example: Samsung Galaxy S Duos, Samsung Galaxy S4, etc.
- (Adjective)+ (Noun)+ : For topics starting with adjective followed by one or more nouns. For example: digital cameras, flat panel, etc.

3. [CD] (Noun)+ / (Noun)+ [CD]: For topics containing one or more nouns and a cardinal. For example: Htc Desire 600, 301 Dual Sim, etc.

POS chunking greatly reduces number of candidate keywords by 75% in extracting topics. We consider keyphrases of length up to only 4, as almost all the topics can be identified of that length.

The semantic representation of a tweet t consists of keywords ($KW(t)$) and keyphrases ($KP(t)$). Next, we build a graph with tweets as nodes and the similarity between two tweets representing the edge weight. The edge weights are calculated using algorithm 1 and the clusters are formed according to the algorithm 2. The similarity between two tweets, t_1 and t_2 is computed as shown in Eq. 3.7.

$$sim(t_1, t_2) = w \times sim(KP(t_1), KP(t_2)) + (1 - w) \times sim(KW(t_1), KW(t_2)) \tag{3.7}$$

where,

$$sim(KP(t_1), KP(t_2)) = |KP(t_1) \cap KP(t_2)| \tag{3.8}$$

and

$$sim(KW(t_1), KW(t_2)) = |KW(t_1) \cap KW(t_2)| \tag{3.9}$$

Here, w denotes the weight given to the keyphrases and $(1 - w)$ denotes the weight given to the keywords. For our experiments we set $w = 0.6$. We rank the edges based on the similarity among tweets and prune the ranked edges by removing low weight edges until all the vertices continue to be covered by the remaining edges. We then cluster the tweets based on nearest-neighbors similarity. Each cluster is stored along with its keywords and keyphrases in the index.

Algorithm 1 Initialization

function FINDTWEETSIMILARITY($tweet_a, tweet_b$)
    $similarity_{ab}$=w*$sim(keywords_a, keywords_b)$+ (1-w)*$sim(keyphrases_a, keyphrases_b)$
end function
function BUILDGRAPH($tweets$)
    for all $i \in tweets$ **do**
        for all $j \in tweets \mid j > i$ **do**
            $edge_{ij}$=$edge_{ji}$=$FindTweetSimilarity(i, j)$
end function

In the online phase, we process each new tweet t by first extracting its keywords and keyphrases. We then query the index for clusters containing the tweet's keywords and retrieve the cluster c^* with the highest score. Score of each cluster is based on the similarity between the tweet and the cluster in terms of their keywords and keyphrases. Tweet t is assigned to the cluster c^* or to a new cluster based on the following intuitive rules. If no clusters in the index match with the tweet keywords, tweet is assigned to a new cluster. Sometimes even though there is a match in the set of keywords, the tweet may belong to a very different sub-topic compared to the sub-topic of the most similar cluster c^*. To avoid

incorrect cluster assignment for the new tweet, we used "Wikipedia title matching" to distinguish subtopics related to a particular topic. We compare tweet's keyphrases with Wikipedia titles by performing a substring match. If at least one keyphrase occurs in one of the titles of a Wikipedia page and if none of the tweet's keyphrases match with the keyphrases of the matched cluster c^*, then we create a new cluster. Otherwise we assign the tweet to the cluster c^*. For example, consider the tweet "The Volvo ocean race has started this year on a high range in England". Based on the simple keyword match, this tweet would get assigned to the cluster containing "Volvo" and "England", but in reality it should form a new cluster namely "Volvo ocean race". Algorithm 3 shows dynamically assigning new tweets and creating clusters.

3.2.1.2 Concept Space Representation (*CSR*) based Approach

A lexical mismatch is caused by occurrence of different words in the tweet which are otherwise semantically related. Transformation of tweets into concept space can help reduce the lexical mismatch. This can enable matching even those tweets that are semantically relevant to each other but do not have any overlapping words. We obtain the semantic representation of a tweet by extracting concepts for a tweet using the TagMe [25] API. The API takes short text snippets as input, disambiguates the entities in the text, and maps these entities to Wikipedia pages. Using this API, we represent each tweet t conceptually as a combination of keyphrases ($KP(t)$), URLConcepts ($UC(t)$) and TweetConcepts ($TC(t)$). The offline phase for the *CSR* based approach is similar to the one for the *SSR* based approach.

In the online phase, we query the index for clusters containing the tweet's keywords and retrieve the k nearest clusters. Similarly, we retrieve the top k nearest clusters for URLConcepts and TweetConcepts. For our experiments, we set k as 10. We then assign a score $R(c)$ to cluster c as shown in Eq. 3.10.

$$R(c) = \frac{\alpha}{R_{KP}(c)} + \frac{\beta}{R_{UC}(c)} + \frac{\gamma}{R_{TC}(c)} \tag{3.10}$$

Here, α, β and γ are the weights given to keyphrases, URLConcepts and TweetConcepts respectively such that $\alpha + \beta + \gamma = 1$. Here $R_{KP}(c)$, $R_{UC}(c)$, $R_{TC}(c)$ are the ranks of the cluster retrieved when queried with keyphrases, URLConcepts and TweetConcepts respectively. A new tweet t is either assigned to the cluster c^* with the highest score or to a new cluster. If no clusters in the index match with the tweet keywords, the tweet is assigned to a new cluster. The tweet is assigned to the cluster c^* if at least one keyphrase matches between cluster c^* and tweet t, or at least one concept matches other than the entity name. Otherwise the tweet t is assigned to a new cluster.

3.2.1.3 Maintaining Cluster Purity and Cluster Labels

We preserve the purity of clusters, by removing the irrelevant concepts from the clusters at regular intervals of tweet arrivals into the clusters. After every m (we set m to four) consecutive tweets a cluster receives, we update the cluster by storing only the most frequently occurring concepts. The concepts of tweets in the cluster, and their frequency values are updated. In *SSR*, we label the clusters using the top occurring keyphrase and keyword. In CSR, we define cluster label using the top occurring keyphrase and concept. As the labels of the cluster change frequently with the incoming tweets, we update the labels of the clusters at regular intervals, δt. For our experiments, we set δt to fifty tweets. Figure 3.4 shows the subtopic graph for entity "Volvo".

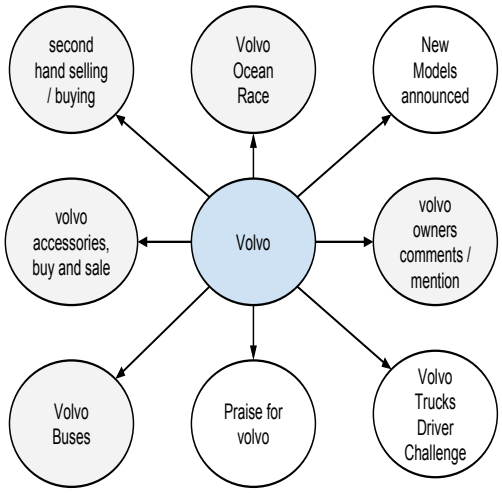

Figure 3.4 Sub-topic graph for the entity volvo

3.2.2 Dataset

Replab(CLEF) 2013 dataset was used. The dataset contains tweets about 61 entities. Each entity has about 700 as training data and 1500 tweets for testing. The training and test data sets are manually labelled by annotators who are trained and guided by experts in online reputation management.

Each tweet in the training and test sets is annotated as follows:

- *Related/Unrelated*: whether the tweet is about an entity.
- *Positive/Neutral/Negative*: the information contained in the tweet has positive/neutral/negative implications for the entity's reputation.

- *Topic*: identifier of the topic (cluster) the tweet belongs to.

- *Alert/Midly-important/Unimportant*: the priority of the topic (cluster) the tweet belongs to.

3.2.3 Experiments

We focus on the task of entity tracking using sub-topic detection. For the offline phase, tweets can be collected by querying for the entity name on Twitter. If there are no initial tweets related to an entity then the online phase starts with an empty cluster set. For our evaluation, we use RepLab (CLEF) 2013 [6] dataset. The dataset contains tweets for 61 entities. Each entity has about 700 tweets for training and 1500 tweets for testing. In the offline phase, we use the training tweets to obtain seed clusters, which are then used in the online phase to cluster test tweets. The data sets are manually labeled by expert annotators.

For evaluation we use two complementary measures, Reliability and Sensitivity as defined by Amigó et al. [7]. Let us consider a system output X and a gold standard $\mathcal{G}$, which are both a set of document relationships $r(d, d')$. The Reliability (R) of relationships in the system output is the probability of finding them in the gold standard. The Sensitivity (S) of predicted relationships is the probability of finding them in the system output when they appear in the gold standard.

Table 3.1 shows the system performance with various values of α, β, and γ. The setting $\alpha = 0.3, \beta = 0.2$ and $\gamma = 0.5$ gave the best results. We infer that both the keyphrases and TweetConcepts features are equally important. Because many tweets do not contain a URL, low weight is assigned to the URL-Concepts. Table 3.2 compares the performance of various methods. Here *Baseline* is the memory-based learning baseline supplied by RepLab. We compare our results with *UNED_ORM*, *REINA*, *LIA* (teams that participated in RepLab 2013 [6]) whose approaches are described in Section 2. The total number of sub-topics in the dataset were 9570. The proposed *SSR* and *CSR* based approaches detected 7545 and 8633 sub-topics respectively. We observe that for a few tweets, the concepts retrieved from TagMe were relatively inaccurate, and this resulted in lower reliability for *CSR*. *SSR* has higher reliability compared to *CSR* because most of the relationships predicted by the system are also found in gold standard. However, the sensitivity is lower for *SSR* because the number of discovered relationships in the system are less, as excessive keyword matching (probably because of WordNet usage) caused some sub-topics to merge into a single sub-topic. The higher sensitivity of *CSR* as compared to that of *SSR* is because *CSR* has higher coverage of relationships over the gold standard.

Table 3.1 Accuracy Results for Various Values of α, β and γ

α	β	γ	R	S	$F1$	#Sub-Topics
0.5	0.2	0.3	0.303	0.521	0.338	8586
0.3	**0.2**	**0.5**	**0.304**	**0.516**	**0.339**	**8633**
0.4	0.3	0.3	0.304	0.522	0.338	8794
0.6	0.2	0.2	0.303	0.512	0.335	8785
0.2	0.3	0.5	0.305	0.516	0.337	8760
0.2	0.4	0.4	0.303	0.529	0.338	8685

Table 3.2 Performance Comparison of Various Methods

Method	R	S	F
CSR	0.304	**0.516**	**0.339**
UNED_ORM	0.460	0.320	0.330
REINA	0.320	0.430	0.290
SSR	**0.496**	0.203	**0.259**
LIA	0.220	0.350	0.250
Baseline	0.150	0.220	0.170

CSR based approach maintains the consistency in both, identifying the number of sub-topics as well as their presence in the gold dataset for each entity. The $F1$ measure values are calculated for each topic individually and averaged over all the topics. As shown in Table 3.2, the proposed approach achieves the highest $F1$ measure value. The increase in the $F1$ measure value by **16.9%** as compared to the *Baseline* and $\sim$**1%** as compared to the best system in RepLab 2013 indicates that the proposed *CSR* approach performs better than the state-of-the-art methods.

3.2.4 Conclusions

In this chapter, we have explored a novel approach by exploiting the semantic and concept based representations of tweets for sub-topic clustering. In the *SSR* based approach, we used keywords (WordNet synonyms, URL keywords), keyphrases and Wikipedia title matching (as criterion for creating new cluster) as features. To handle the issue of overmatching of keywords due to WordNet usage, we propose the *CSR* based approach. In the *CSR* based approach, we used TweetConcepts, URLConcepts and keyphrases as features. We maintain the purity of clusters by periodically cleaning up the clusters. Experiments on the RepLab (at CLEF 2013) dataset showed that the proposed approach achieves significant performance gains over the baseline and other systems using metrics like Reliability, Sensitivity

and $F1$ measure. In the future, we would like to extend this study by incorporating the similarity between concepts in the *CSR* based approach. We would also like to run a second pass on the formed clusters to further improve the quality of clusters. Cluster splitting and merging can be done based on the inter-cluster and intra-cluster similarity to obtain fine-grained clusters. Efforts can also be put in the direction of classification of tweets into predefined categories like (1) Products/Services, (2) Innovation, (3) Workplace, (4) Citizenship, (5) Governance, (6) Leadership, and (7) Performance, (8) Others etc.

Algorithm 2 Creating Base Clusters

1: **function** CREATEBASECLUSTERS($weightedEdgeList$)
2: Sort($weightedEdgeList$)
3: **for all** $edge \in weightedEdgeList$ **do**
4: $source$=edge($source$)
5: $dest$=edge($destination$)
6: **if** not visited($source$) or not visited($dest$) **then**
7: $cluster_{source}$.add($source$)
8: $cluster_{source}$.add($dest$)
9: visited($source$) =true
10: visited($dest$) =true
11: **for all** $vertex \in$ neighbours($source$) and $vertex$!=$dest$ **do**
12: $SimVertex$=FindSimNeighbour($vertex$)
13: **if** $SimVertex$ == $source$ or $SimVertex$ == $dest$ **then**
14: $cluster_{source}$.add($vertex$)
15: visited($vertex$)=true
16: **if** all vertices visited **then**
17: break
18: **end function**
19: Similarly, repeat steps 11-17 for destination vertex.
20: **function** FINDSIMNEIGHBOUR($vertex$)
21: $MaxSim$=-1
22: $MaxSimVertex$=$\emptyset$
23: **for all** $node \in$ neighbours($vertex$) **do**
24: **if then**$MaxSim$¡$edge_{node,vertex}$
25: $MixSim$=$edge_{node,vertex}$
26: $MaxSimVertex$=$node$
27: **if** $MaxSim > 1.6$ and $keyphrases(vertex) \ni wikipedia_{titles}$ **then**
28: return $MaxSimVertex$
29: **else**
30: return $\emptyset$
31: **end function**

Algorithm 3 Dynamically Creating Clusters

Require: Base Clusters

function CREATECLUSTERS($tweet$)
    $keywords$=getKeywords($tweet$)
    $keyphrases$=getKeyphrases($tweet$)
    $entity$=GetTweetEntity($tweet$)
    keywords.add(wordnetSynonyms(keywords))
    keywords.add(URLkeywords($tweet$))
    $flag$=false
    $ClusterIds$=SearchIndex($keywords$,$entity$)
    for all $cluster \in ClusterIds$ **do**
        if $keyphrases \in cluster.getKeyphrases()$ **then**
            flag=true
            break
        if $keyphrases \in wikipedia_{titles}$ and $!flag$ **then**
            assign tweet to new cluster
        else
            $MaxScore = 0.0$
            $SimCluster = \emptyset$
            for all $cluster \in ClusterIds$ **do**
                $cluster_{keywords}$ = getKeywords($cluster$)
                $tempSim$=Similarity($keywords$,$cluster_{keywords}$)
                if $MaxScore < tempSim$ **then**
                    $MaxScore$=$tempSim$
                    $SimCluster = cluster$
end function

Chapter 4

Structured Information Extraction from Natural Disaster Events on Twitter

4.1 Motivation of Our Approach

As soon as natural disaster events happen, users are eager to know more about them. However, search engines currently provide a ten blue links interface for queries related to such events. Relevance of results for such queries can be significantly improved if users are shown a structured summary of the fresh events related to such queries. This would not just reduce the number of user clicks to get the relevant information but would also help users get updated with more fine grained attribute-level information.

Twitter is a great source that can be exploited for obtaining such fine-grained structured information for fresh natural disaster events. Such events are often reported on Twitter much earlier than on other news media. However, extracting such structured information from tweets is challenging because: 1. tweets are noisy and ambiguous; 2. there is no well defined schema for various types of natural disaster events; 3. it is not trivial to extract attribute-value pairs and facts from unstructured text; and 4. it is difficult to find good mappings between extracted attributes and attributes in the event schema.

We propose algorithms to extract attribute-value pairs, and also devise novel mechanisms to map such pairs to manually generated schemas for natural disaster events. Besides the tweet text, we also leverage text from URL links in the tweets to fill such schemas. Our schemas are temporal in nature and the values are updated whenever fresh information flows in from human sensors on Twitter. Evaluation on $\sim$58000 tweets for 20 events shows that our system can fill such event schemas with an F1 of $\sim$0.6.

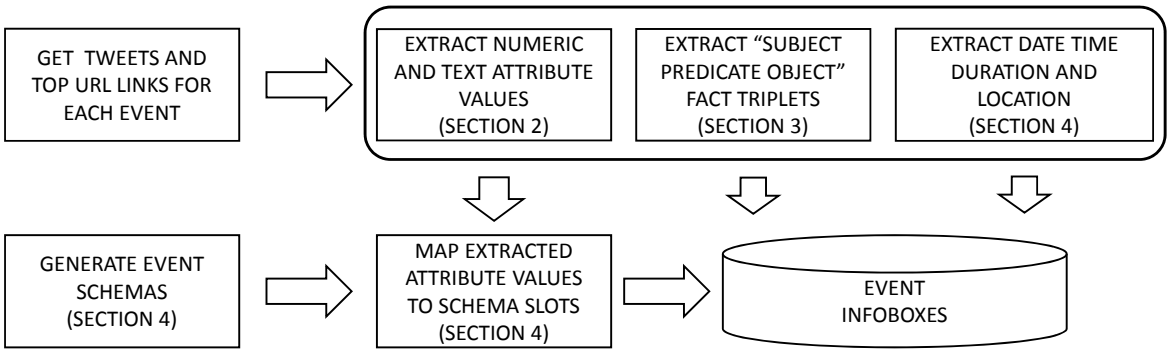

Figure 4.1 System Diagram

4.2 Extraction of Attribute-Value Pairs

We use linguistic tools like the Stanford Typed Dependencies [18] and the CMU Tweet POS Tagger [26] to understand the semantics of the tweets. We remove user mentions, URL links and retweet symbols from the tweet before performing linguistic analysis.

The aim of this section is to obtain attribute-value pairs from tweets. When the units of the values are also mentioned next to the value, we extract the units too. In this section, we first provide a basic introduction to Stanford typed dependencies. After that we propose two novel algorithms: one for extracting numeric attribute-value pairs from tweets and the other for extracting textual attribute-value pairs.

4.2.1 Basic Introduction to Stanford Dependencies

The Stanford typed dependencies representation [18] was designed to provide a simple description of the grammatical relationships in a sentence that can be used to extract textual relations. The representation contains approximately 50 grammatical relations. For example, consider the tweet "Arizona struggles to contain blaze: Conflagration engulfs 110,000 acres... http://bit.ly/RpMYv4". The dependencies obtained after parsing the cleaned tweet are as follows: "root(ROOT-0, struggles-2); nsubj(struggles-2, Arizona-1); aux(contain-4, to-3); xcomp(struggles-2, contain-4); dobj(contain-4, blaze-5); nsubj(engulfs-8, Conflagration-7); parataxis(struggles-2, engulfs-8); num(acres-10, 110,000-9); dobj(engulfs-8, acres-10); prep_of(acres-10, land-12)". Here, nsubj, amod, nn, etc. are all dependencies.

The dependencies are all binary relations: a grammatical relation holds between a governor (also known as a regent or a head) and a dependent. For example, for the dependency "nsubj(struggles-2, Arizona-1)", "struggles" is the governor, and "Arizona" is the dependent.

The following dependencies are important with respect to this paper and so we define them here.

- $root$: The root grammatical relation points to the root of the sentence.
- $nsubj$: A nominal subject is a noun phrase which is the syntactic subject of a clause.
- $dobj$: The direct object is the noun phrase which is the (accusative) object of the verb.
- $pobj$: The object of a preposition is the head of a noun phrase following the preposition, or the adverbs "here" and "there".
- nn: A noun compound modifier is any noun that serves to modify the head noun.
- $prep*$: A prepositional modifier of a verb, adjective, or noun is any prepositional phrase that serves to modify the meaning of the verb, adjective, noun, or even another preposition.
- num: A numeric modifier of a noun is any number phrase that serves to modify the meaning of the noun with a quantity.
- $number$: An element of compound number is a part of a number phrase or currency amount.
- $amod$: An adjectival modifier of a noun phrase is any adjectival phrase that serves to modify the meaning of the noun phrase.
- dep: A dependency is labeled as dep when the system is unable to determine a more precise dependency relation between two words.

For detailed understanding of these dependencies we redirect the reader to [18].

4.2.2 Numeric Attribute-Value Extraction

Identifying numeric attribute-value pairs from tweets is challenging. Naïve approaches like considering the neighboring words close to numeric literals as attribute names do not always work. For example consider the tweet: "Death toll rises to 123 in Mexico following Tropical Storm Ingrid". Here the attribute is "Death toll" and the value is "123". We can observe that "123" cannot be linked to

the previous word or the next word. It needs to be linked to the phrase which actually describes it by understanding the relation.

Our approach to numeric attribute-value extraction consists of the following sub-modules: splitting sentences into self-complete sub-units, handling of special cases of attribute-value mentions, extracting attribute-value pairs using dependencies, and extracting complete attribute names.

4.2.2.1 Splitting Sentences into Self-complete Sub-units

If the tweet has two or more sentences or multiple subjects then we split the tweet into separate sentences, each containing a single subject. Since each subject gives additional description about a particular attribute, so mapping a subject to its corresponding attribute is critical. For example, for the tweet "Arizona struggles to contain blaze: Conflagration engulfs 110,000 acres... http://bit.ly/RpMYv4". There are two subjects, but the appropriate one for the attribute "land" is "nsubj(engulfs-8, Conflagration-7)". The splitting is done based on a set of delimiters, or by considering subtrees of the parse tree.

4.2.2.2 Handling of Special Cases of Attribute-Value Mentions

Here we discuss two interesting cases as follows.

Case 1 - Numeric Values are mentioned side-by-side: Consider the tweet: "#USGS M 1.9 - 4km N of Hydesville, California: Time 2014-07-03 02:31:00 UTC 2014-07-02 19:31:00 -07:00 at ep..". Here, the attributes "M", "N" and values "1.9", "4km" are side by side. We handle this case using the following simple rule: Given any word marked as "numeric", if the next word is tagged as a noun (common noun, or proper noun) then we consider the next word as the attribute else we consider the previous word as the attribute name.

Case 2 - Attribute-Value pairs mentioned in a Sequence: In some tweets, attribute-value pairs are listed as a sequence with some delimitier. Repeated occurrences of ("numeric", "noun") pairs helps us detect such cases and identify the attribute and its value appropriately. For example, consider the tweet "Mag: 3 - Depth: 116 km - UTC 8:07 AM - Tarapaca, Chile - EMSC". Here attributes "Mag", "Depth", "UTC" are extracted along with the values "3", "116km", "8:07 AM" respectively.

4.2.2.3 Extracting Attribute-Value Pairs using Dependencies

After handling the special cases mentioned above, the remaining sentence (after removing the already extracted attribute-value pairs) is analyzed to extract the dependencies. We propose the following rules to exploit these dependencies to extract attribute-value pairs, the subject and the object.

- The subject and the object are extracted from the dependent parts of the $nsubj$ and $dobj$ dependencies respectively.
- The (governor, dependent) pair of every num dependency provides an attribute-value pair.
- The (governor, dependent) pair of every nn dependency provides an attribute-value pair if the dependent contains digits.

4.2.2.4 Extracting Complete Attribute Names

Since the dependencies provide relationships between pairs of words only, one needs to combine a few dependencies to extract complete attribute names. The detailed pseudo-code for the algorithm is presented in Algorithm 4. We propose the following rules for exploiting various dependencies to obtain complete phrases. Let AV be the set of extracted attribute-value pairs so far, where $AV[i].a$ and $AV[i].v$ denote the attribute and the value part of the i^{th} attribute-value pair.

- If the dependent of an $nsubj$ matches $AV[i].a$, expand $AV[i].a$ to include the governor too. (Lines 4 to 6)
- If the dependent or the governor of an nn matches $AV[i].a$, expand $AV[i].a$. (Lines 7 to 11)
- If the governor of a $prep\_*$ matches $AV[i].a$, expand $AV[i].a$ to include the preposition and the dependent. (Lines 12 to 15)
- If the governor of any $nsubj$, nn or $prep\_*$ matches the subject, expand the subject to include the dependent of the dependency too. (Lines 16 to 17)

4.2.3 Textual Attribute-Value Extraction

Textual attribute-value pairs are ones in which the value is non-numeric text. Compared to numeric attribute-value pairs, it is more challenging to mine textual attribute-value pairs due to the lack of any

Algorithm 4 Extraction of Complete Attribute Names

Input: (1) Dependency list DL (2) Set of extracted attribute-value pairs AV
Output: Set of attribute-value pairs AV with complete attribute names

```
1:  for all d ∈ DL do
2:      if d.rel ∈ {nn, prep_*, nsubj} then
3:          for all av ∈ AV do
4:              if d.rel == nsubj then
5:                  if av.a contains d.dep then
6:                      av.a ← d.gov + d.dep
7:              else if d.rel == nn then
8:                  if av.a contains d.dep then
9:                      av.a ← d.gov
10:                 else if av.a contains d.gov then
11:                     av.a ← d.dep + d.gov
12:             else if d.rel == prep_* then
13:                 Let p be the preposition.
14:                 if av.a contains d.gov then
15:                     av.a ← d.dep + p + av.a
16:         if Subject == d.gov then
17:             Subject ← d.dep + d.gov
```

numeric clues. For example, consider the tweet: "Hurricane Sandy cancels many flights at Orlando Airport". Here, "(Airport, Orlando)" and "(Hurricane, Sandy)" are the two attribute-value pairs.

Given a tweet, there are three ways to obtain attribute-value pairs: (1) a central attribute-value pair related to the subject of the tweet ($CentralAV$), (2) attribute-value pairs related to the root word of the tweet ($RootAV$), and (3) attribute-value pairs connected to preposition dependencies ($PrepAV$).

For example, in the tweet above, "(Hurricane, Sandy)" is the central attribute-value pair. Given the dependencies for the tweet, we can leverage them to identify the above three types of textual attribute-value pairs.

Algorithm 5 explains about the process of extraction of the three types of textual attribute-value pairs. Lines 2 to 9 relate to extraction of $RootAV$. First the root word is extracted using the root dependency. Next, other words dependent on the root word are extracted. Further, the $dobj$, $pobj$ and $amod$ dependencies are exploited to obtain $RootAV$ where the root word is the attribute.

Lines 10 to 17 relate to the extraction of $CentralAV$. First the subject and the verb related to the subject are extracted using the $nsubj$ dependency. Then the nn dependency is used to extract the $CentralAV$ pair where the subject forms the major part of the attribute name.

Lines 18 to 33 relate to the extraction of $PrepAV$. We refer to the (governor, dependent) pair of a $prep*$ dependency as a prepositional pair. Prepositional pairs could themselves be used as attribute-

value pairs. First we obtain the prepositional pairs. Next, we use nn dependencies to enhance the dependents and governors of the prepositional pairs. If the nn dependency does not contain a noun and its dependent matches the governor of a prepositional pair, it is used to obtain an attribute-value pair.

Finally, the sets $RootAV$ and $PrepAV$, and $CentralAV$ are merged to get the set of all textual attribute-value pairs.

Algorithm 5 Extraction of Textual Attribute-Value Pairs

Input: Dependency list DL
Output: Set of textual attribute-value pairs AV

Let $CentralAV$ be the central attribute-value pair, $RootAV$ be the set of attribute-value pairs related to the root word of the tweet, $PrepAV$ be the set of attribute-value pairs connected to preposition dependencies.
$rootWord \leftarrow d.dep$ where $d \in DL$ and $d == root$.
$rootWordValues \leftarrow \{d.dep | d.rel = dep$ and $d.gov = rootWord$ and $d \in DL\}$
for all $d \in DL$ **do**
    for all $v \in rootWordValues$ **do**
        if ($d.rel == dobj$ or $d.rel == pobj$) and ($d.gov == v$) **then**
            $RootAV.Add((rootWord, v))$
    if $d.rel == amod$ and $d.gov == rootWord$ **then**
        $RootAV.Add((rootWord, d.dep))$
$subject \leftarrow d.dep$ where $d \in DL$ and $d == nsubj$.
$subjectVerb \leftarrow d.gov$ where $d \in DL$ and $d == nsubj$.
for all $d \in DL$ **do**
    if $d.rel == nn$ and $d.gov == subject$ **then**
        if {root,dobj} $\notin DL$ **then**
            $CentralAV \leftarrow (d.dep + subject, subjectVerb)$
        else
            $CentralAV \leftarrow (subject, d.dep)$
Let PP be the set of all prepositional pairs.
for all $d \in DL$ **do**
    if $d.rel == prep*$ **then**
        $PP.Add(d.gov, d.dep)$
for all $d \in DL$ **do**
    for all $pp \in PP$ **do**
        if $d.rel == nn$ **then**
            if $d.gov == pp.gov$ **then**
                Update pp to $(d.dep + pp.gov, pp.dep)$.
            if $d.gov == pp.dep$ **then**
                if $d.gov$ or $d.dep$ is a noun **then**
                    Update pp to $(pp.gov, d.dep + pp.dep)$.
                else
                    $PrepAV.Add(d.dep, pp.dep)$
                    Remove pp from PP.
$PrepAV \leftarrow PrepAV \cup PP$
$AV \leftarrow RootAV \cup \{CentralAV\} \cup PrepAV$

4.3 Extraction of Fact Triplets

A fact triplet consists of three main parts: Subject, Predicate and the Object. For example consider the tweet "Volvo Ocean Race set for raft of changes to boats, teams and route in bid to appease sailors and sponsors via @Telgraph http://soc.li/AenbU9M". The extracted fact triplet for this tweet is "(Volvo Ocean Race : appease : sailors, sponsors)". In this section, we discuss a mechanism to extract such fact triplets using Stanford dependencies from any tweet. We extract fact triplets only from those tweets in which at least two entities occur.

Algorithm 6 presents our mechanism for extraction of fact triplets. First we obtain the subjects and objects in the tweet using various dependencies (Lines 1 and 2). Next, we obtain the root word and its index (Lines 3 and 4). Here index refers to the word position in the tweet. If there is no subject in the tweet, we use the root word to form a subject (Lines 6 to 7). Similarly, if there is no object in the tweet, we use the dep dependency to obtain an object (Lines 9 to 10). Further, we use various dependencies to expand the subjects and objects to get their complete forms (Lines 11 and 12). Subjects and objects are then matched using the verbs that appear with them in the dependencies (Lines 13 and 14). These verbs form the predicates, and are expanded using the prepositional modifiers (Line 15). Finally matching expanded (subject, predicate, object) are returned as fact triplets.

Algorithm 6 Extraction of Fact Triplets

Input: Dependency list DL
Output: Fact Triplets

1: $numSubjects \leftarrow$Number of $nsubj$ and $nsubjpass$.
2: $numObjects \leftarrow$Number of $dobj$ and $pobj$.
3: $rootWord \leftarrow d.dep$ where $d \in DL$ and $d == root$.
4: $rootIndex \leftarrow d.dep.index$ where $d \in DL$ and $d == root$.
5: Obtain list of subjects using $nsubj$ and $nsubjpass$.
6: **if** $numSubjects$ == 0 **then**
7: Add $d.dep + d.gov$ as the subject where $d.gov == rootWord$ and ($d.rel == amod$ or $d.dep.index < rootIndex$).
8: Obtain list of objects using $dobj$ and $pobj$.
9: **if** $numObjects$ == 0 **then**
10: Add $d.dep$ as the object where $d.rel == dep$.
11: Use dependents of nn and $amod$ with governor matching the object to expand the object.
12: Use dependents of dep and nn with governor matching the subject to expand the subject.
13: Match subjects with objects using matching verbs which are governor/dependent of dependencies containing the subject or object.
14: Set the matching verbs as predicates for (subject, object) pairs.
15: Expand predicates using prepositional modifiers.
16: Generate a fact triplet for every subject-predicate-object.

4.4 Filling of Event Schemas

In this section, we discuss the challenges in generation of event schemas and how we created schemas for natural disaster events. Next, we discuss our algorithm to map extracted attributes to schema slots (or attributes).

4.4.1 Generation of Event Schemas

One can generate schemas for natural disaster events automatically as follows. For any event type (e.g., earthquakes), gather Infoboxes for a few events of that type from Wikipedia. Consider the top most frequent fields as attributes for the schema for that event type. This technique does not work for events which do not have Infoboxes on Wikipedia (e.g., "Justin Bieber's birthday"). Also for general events, identifying the most relevant Infobox type is difficult. However, for natural disaster events, this is expected to work well.

But, there is a large mismatch between the Wikipedia Infobox attribute names and the attributes extracted from Twitter. Most of the attribute-value pairs extracted from Twitter events are quite new and are not present in Wikipedia Infoboxes. Hence, we had to resort to manual generation of event schemas with guidance from Wikipedia Infoboxes.

For each event type we use the tweets of one event in the training phase to manually learn the attributes for the event schema. Thus, we grow our schema beyond the one that could be obtained using Wikipedia Infoboxes.

Besides the attribute names, the event schemas contain more metadata information for every attribute described as follows. (1) We extract ranges for the Wikipedia Infobox attributes. For each event type, we determine the minimum and maximum value an attribute of that type can hold and define the range for each attribute. For attributes which are not present in Wikipedia, we manually assign attribute value ranges. Thus, each attribute of every event type is associated with a range of values it can take. (2) Next, we define the data type for each attribute of each event type. The event schema attributes can be of the following types: integer, float, string, date, time. (3) We also define the units for each event attribute. For example, for the attribute $wind\_speed$ the units will be 'mph','km/h' and for $funds\_donated$ would be '$' or 'euros' etc. (4) Finally, for each schema attribute for each event type, we identify a set of synonyms. For example, "total_cost, total_loss, money_loss" are synonyms for "total_economic_impact". Similarly, "body_wave_magnitude" is synonym for "mb" for the earthquake event schema.

4.4.2 Populating Values of Schema Attributes

As described in Section 4.2, for each attribute-value pair, we also extract the subject and object which are useful in mapping an extracted attribute-value pair to the event schema attribute. Often times, an attribute takes multiple values across various tweets for the event. We assign the most frequent value to each attribute. After that, we map the attribute-value pair to a schema attribute as described in the following. Algorithm 7 presents the pseudo-code for the mapping algorithm.

For each extracted attribute-value pair (a, v) and each schema attribute s, we compute a match score (Lines 3 to 28). The match score depends on the following: (1) does v lie within the range of attribute s, (2) similarity between units of s and v, (3) similarity between units of s and subject of a, (4) similarity between units of s and object of a, (5) similarity between s and subject of a, (6) similarity between s and object of a, (7) similarity between s and a.

When computing similarity values, the score is incremented dependent on whether the similarity is greater than a threshold T or not. The similarity values lie between 0 and 1. When computing the score, various factors are given appropriate weights based on their importance. Based on this match score, we find the best schema attribute for each extracted attribute (Line 29). This also gives us a list of candidate extracted attributes for every schema attribute. We sort this list by score and map the schema attribute to the extracted attribute with the highest score (Line 34). Ties are resolved using frequency of occurrence of the candidate extracted attributes.

Some extracted attributes do not get mapped to any schema attribute. If they are frequent enough, we also list such attribute-value pairs in the structured event summary. Also, we use the SUTIME library [13] for extracting date, time and duration values from tweets. Based on the frequency across all tweets for an event, the final date, time and duration for the event occurrence are determined. Finally, fact triplets extracted using Algorithm 6 are also included in the event summary.

4.5 Experiments

In this section, we present details of our dataset, our experiment design, results of experiments conducted, and analysis of results.

Algorithm 7 Mapping Attribute-Value Pairs to Event Schemas

Input: (1) Event Schema (2) Attribute-Value Pairs for an Event
Output: Mapping between Attribute-Value Pairs and Attributes in Event Schema, $Mapping$

Get the most frequent value for each extracted attribute.
$schemaAttributeToCandidates \leftarrow \phi$
for all $(a, v) \in AV$ **do**
    $subject \leftarrow$ Subject related to (a, v).
    $object \leftarrow$ Object related to (a, v).
    $schemaAttributeToScore \leftarrow \phi$
    for all attribute $s \in$ schema attributes **do**
        $score \leftarrow 0$
        $units \leftarrow$ units for attribute s
        $range \leftarrow$ range for attribute s
        $type \leftarrow$ type for attribute s
        if v has type $type$ **then**
            $score \leftarrow score + 1$
        if v lies within the range $range$ **then**
            $score \leftarrow score + 1$
        if $sim(units, v) \geq T$ **then**
            $score \leftarrow score + 2 \times sim(units, v)$
        if $sim(units, subject) \geq T$ **then**
            $score \leftarrow score + 2 \times sim(units, subject)$
        if $sim(units, object) \geq T$ **then**
            $score \leftarrow score + 2 \times sim(units, object)$
        if $sim(a, s) \geq 0.8$ **then**
            $score \leftarrow score + 3 \times sim(a, s)$
        if $sim(s, subject) \geq T$ **then**
            $score \leftarrow score + 2 \times sim(s, subject)$
        if $sim(s, object) \geq T$ **then**
            $score \leftarrow score + 2 \times sim(s, object)$
        $schemaAttributeToScore.Add(s, score)$
    Find the schema attribute K with max score S
    if types of a and K do not match **then**
        $S \leftarrow 0$
    $schemaAttributeToCandidates.Add(K, (a, S))$
for all attribute $s \in$ schema attributes **do**
    $f \leftarrow$ the candidate attribute with max score using $schemaAttributeToCandidates[s]$.
    $Mapping.Add(s, f)$

4.5.1 Dataset

We selected five natural disaster event types: earthquakes, hurricanes (or typhoons), floods, wildfires and landslides. For each event type, we crawled tweets of 3–5 recent events listed as follows.

- Earthquakes: Chile Earthquake, Visayas Earthquake, Mexico Earthquake, Solomon Earthquake, Vizag Earthquake
- Hurricanes: Hurricane Sandy, Hurricane Amanda, Hurricane Ingrid Manuel, Typhoon Haiyan, Typhoon Phailin
- Floods: Balkan Floods, Serbia Floods, US Colorado Floods, Uttarakhand Floods
- Wildfires: California Wildfire, Alaska Wildfire, Arizona Wildfire
- Landslides: Washington Landslide, Zambales Landslide, Bolivia Landslide

We obtained related tweets using the Twitter search API. On an average the dataset consists of $\sim$3000 tweets per event. We use one event for each type to learn the event schema and then use that schema for all the events of the same type.

4.5.2 Accuracy of Filling the Event Schemas

For schema filling, we use threshold $T = 0.8$. Table 4.1 shows the precision, recall and F1 for the task of filling schemas for different events, except for the events used for learning the schema itself. As shown in the table, the event schemas contain around 30 attributes per event on an average. We measured the precision with which the proposed algorithms could fill the event schemas, and also the recall, i.e., the number of event attributes that could be filled.

Tweets may not always contain all information. We extract URLs mentioned in tweets for an event. We expand the shortened URLs and filter out links which relate to images, videos and Twitter or Facebook posts. For each event, we extract top 20 URLs and crawl the text of the URL links. We run the algorithms proposed in Sections 4.2 and 4.3 to extract attribute-value pairs and facts respectively on URL text too. Extracted attribute-value pairs from URLs are again mapped to event schemas.

We summarize the results shown in Table 4.1 across all events in Table 4.2 for all the three settings: "Only Web-links", "Only Tweets", and "Tweets + Web-links". The precision for "Only Web-links" is more because web text is more structured compared to the tweets. This results in accurate mapping of a

			Only Tweets			Only Web-links			Tweets + Web-links		
Event-Name	#Tweets	#Attributes	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Earthquake_Mexico	5675	35	0.961	0.714	0.819	1.0	0.2857	0.444	0.962	0.742	0.837
Earthquake_Solomon	976	35	0.823	0.4	0.538	0.615	0.228	0.332	0.833	0.428	0.565
Earthquake_Visayas	177	35	0.923	0.371	0.529	0.833	0.285	0.424	0.866	0.4	0.547
Earthquake_Vizag	199	35	0.8	0.228	0.354	1.0	0.142	0.248	0.818	0.257	0.391
Hurricane_Amanda	4390	42	0.833	0.476	0.605	1.0	0.166	0.284	0.869	0.50	0.634
Hurricane_IngridManuel	2253	42	0.857	0.285	0.427	0.875	0.333	0.482	0.863	0.452	0.593
Typhoon_Haiyan	5376	42	0.809	0.404	0.538	1.0	0.214	0.352	0.695	0.380	0.491
Typhoon_Phailin	345	42	0.818	0.261	0.395	0.909	0.238	0.377	0.846	0.261	0.398
Floods_Serbia	5835	35	0.75	0.428	0.544	0.8	0.228	0.354	0.75	0.428	0.544
Floods_USColorado	370	35	0.88	0.228	0.362	0.928	0.371	0.530	0.933	0.4	0.559
Floods_Uttarakhand	5115	35	0.764	0.371	0.499	0.916	0.314	0.467	0.947	0.514	0.666
Wildfire_Alaska	3118	29	1.0	0.448	0.618	0.833	0.517	0.638	0.894	0.551	0.681
Wildfire_Arizona	5765	29	0.875	0.482	0.621	0.866	0.448	0.590	0.944	0.586	0.723
Landslide_bolivia	721	19	0.88	0.421	0.569	0.8	0.21	0.332	0.9	0.473	0.620
Landslide_Zambales	31	19	0.80	0.21	0.332	1.0	0.421	0.592	1.0	0.526	0.689

Table 4.1 Accuracy of Filling the Event Schemas with Structured Information from Tweets, Web-links and Tweets + Web-links

	Avg. P	Avg. R	Avg. F1
Only Tweets	0.851	0.385	0.516
Only Web-links	0.891	0.293	0.429
Tweets + Web-links	0.874	0.460	0.595

Table 4.2 Average Precision (P), Recall (R), and F1 for the three Variations

value to a particular attribute. However, the recall is less as much of the information was not expressed in the top few Web-links, so less number of attribute-value pairs were discovered. The recall increased when both tweets and web-links were used to extract attribute-value pairs as expected. This is because some attributes were expressed in tweets and some were expressed in web-links, so in combination it resulted in more attribute-value pairs. Overall, we obtained best F1 with "Tweets + Web-links".

4.5.3 A Case Study: "Chile Earthquake"

Table 4.3 shows the attribute-value pairs extracted and mapped to the earthquake schema for the event "Chile Earthquake".

Note the variety of attributes that can be extracted from tweets. We could obtain magnitude of the earthquake on various scales including (1) local magnitude (ML), commonly referred to as "Richter magnitude," (2) body-wave magnitude (Mb), and (3) moment magnitude (Mw). We could also ob-

Attribute	Value
areas_affected	chile iquique antofagasta
distance_miles	6.6
magnitude	5.0
mw (moment magnitude)	5.9
mb (body-wave magnitude)	4.7
ml (local magnitude)	4.0
death_toll	1,655
people_evacuated	300
missing_people	40k
date	2014-05-05
duration	1 minute (P1M)
time	05:00
tsunami_warning	3
direction@e	98km
direction@ne	47km
direction@n	73km
direction@se	34km
direction@sw	67km
direction@s	20.1km
direction@nw	19km
depth	10.0

Table 4.3 Event Schema filled with Attribute-Values for "Chile Earthquake"

tain drilled down numbers about people affected: people dead, people evaluated and people missing. We could also obtain severity of the tsunami warning and the impact distances in various directions. Showing such structured information for the query "Chile earthquake" would surely be better than what popular search engines show today.

4.5.4 Temporal Analysis of Attribute-Value Pairs

We also performed temporal analysis regarding how the event schemas get populated and how the attribute-value pairs evolve over time. Table 4.4 shows the variation in the values of attributes over time from five different events. Each line describes an attribute from one of the five events. The columns correspond to hours (H) or days (D) after the event. Each table cell represents the value of the attribute at that time point and the number of tweets from which that value was extracted. From the temporal analysis, we make the following observations.

- People talk more about attributes like number of people died, magnitude, direction, number of people affected compared to other attributes. Technical attributes like "mb, ml, etc." also appear on Twitter but they are usually put up by news agencies.

- Usually technical attributes like the magnitude, depth of the epicenter, etc. appear first on Twitter. After some time, when field analysis gets done, people start tweeting about the damage. This is when we observe attributes like people affected, schools affected, people injured getting populated.

- Attribute values that appear in the beginning are not very trustworthy. Initially people tweet various values for an attribute from the sources they have access to. Slowly over time the attribute values become stable. For example, as shown in Table 4.4, the value of the "magnitude" attribute fluctuates a lot in the initial hours and becomes stable only around the 11^{th} hour after the event.

- Some attributes are inherently temporal in nature. For example, as can be seen in Table 4.4, the value for the attribute '$people\_dead$' gradually increases. As more and more people are confirmed dead, this number keeps on increasing.

	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12	12-24 H	2-5 D	6-10 D
direction@w	84km 35	84km 2	84km 1	-	-	84km 1	-	-	-	-	-	-	84km 2		
direction@e	64km 2	-	-	-	-	-	-	-	-	-	-	-	-	-	60km 16
depth	-	-	-	-	102km 12	-	-	-	-	-	-	-	-	-	-
mb	8.0 175	8.0 114	-	8.0 266	8.0 107	7.9 63	8.0 25	7.9 66	7.9 46	-	-	-	-	-	-
magnitude	6.2 189	5.1 28	5.5 86	5.3 25	4.6 20	6.9 2	6.3 2	5.0 28	4.9 55	7.2 421	7.2 421	-	-	-	-
people_affected	-	150 42	-	150 10	150 5	150 1	-	150 2	-	-	-	-	-	-	-
direction@n	-	-	-	-	-	-	62km 2	-	-	-	-	-	-	-	-
people_dead	24 4	-	43 3	43 1	70 7	36 7	58 1	80 5	-	123 3	180 11	-	2,100 3	-	-
people_evacuated	-	-	2 1	-	-	-	-	-	-	-	-	-	-	115 1	250 6
people_injured	-	-	-	-	58 1	-	-	-	-	-	-	-	100 19	-	-
schools_affected	-	-	-	-	-	-	-	-	-	-	-	-	four 14	-	150 1

Table 4.4 Temporal Analysis for five different Events

Tweet	Fact Triplet
Nearly 69,000 Con Edison customers in NYC and Westchester County have already lost power	(Edison customers, lost, power)
Waves begin crashing over Chelsea Piers in Manhattan	(Waves, crashing, Chelsea Piers)
Hurricane Sandy Hits NJ	(Hurricane Sandy, Hits, NJ)
Hurricane Sandy halts international #flights, too - from @LAtimes : http://tinyurl.com/93k6c3o	(Hurricane Sandy, halts, internation
Florida Crews Sent North To Assist Hurricane Sandy Victims http://cbsloc.al/RjDMSA	(Florida Crews, Assist, Hurricane S

Table 4.5 Fact Triplets for the Event "Hurricane Sandy"

4.5.5 News vs. Tweets

We also performed an analysis of what attributes get filled using news URL text versus tweets. We observed that on Twitter news agencies often put out very technical information about events like body-wave magnitude, moment magnitude, etc. Indeed this information is not found on news webpages. We hypothesize that this is because although the news agencies have this information, reporters prefer to publish only non-technical information which appeals the mass. Hence, they refrain from publishing very technical information, but concentrate on location of the events, number of people dead or injured. These are the aspects of events that appeal the mass in general.

4.5.6 Fact Triplets

Along with the attribute-value pairs, we also display fact triplets as part of the event summary. Table 4.5 shows a set of few such triplets extracted for the event "Hurricane Sandy".

4.6 Conclusion

In this chapter, we studied the problem of extracting structured information for natural disaster events from Twitter. Specifically we focused on extracting attribute-value pairs and fact triplets. We solved this problem by first performing linguistic analysis like part of speech tagging and dependency parsing. After that we proposed three novel algorithms for numeric attribute-value extraction, textual attribute-value extraction, and fact triplet extraction. We also proposed an algorithm to map the extracted attributes to a schema for the corresponding event type. Experiments on 58000 tweets for 20 events show the effectiveness of the proposed approach. Such a structured event summary can significantly improve the relevance of the displayed results by providing key information about the event to the user without any extra clicks. In the future, we would like to generalize this approach to other events on Twitter.

Chapter 5

Extraction of Fact Triplets from Sub-topics on Twitter

5.1 Sub-Topic Detection

In this section, we discuss our methodology to identify sub-topics from tweets related to an event entity. For example, for the entity "Volvo", sub-topics could be "second hand selling/buying", "volvo accesories, buy and sale", "Volvo buses", "Volvo ocean race", "Praise for volvo", etc. For sub-topic detection first we represent tweets in a rich semantic space as discussed in Sub-section 5.1.1. Further, we design a classifier to decide whether a tweet belongs to a particular sub-topic or not. We discuss the feature design for this classifier in Sub-section 5.1.2. Finally in Sub-section 5.1.3, we present the details of our sub-topic detection methodology.

5.1.1 Semantic Representation for Tweets

We extract various forms of semantic representation for every tweet and index these in Lucene [1]. Semantic concepts (or phrases) are computed across the following dimensions.

- **Tweet Concepts**: We obtain tweet concepts by mapping the tweet text to the corresponding Wikipedia page titles using TagMe [25]. TagMe identifies meaningful substrings in an unstructured text and links them to a pertinent Wikipedia page.

- **URL Meta Concepts**: We extract the title and keywords (from the HTML "meta" tag) of the page pointed by the URL present in the tweet. Then we get the corresponding concepts which are mapped to Wikipedia page titles using TagMe.

[1] `http://lucene.apache.org/`

- **URL Text Concepts**: We extract the text of the page pointed by the URL present in the tweet. Then we get the first ten concepts which are mapped to Wikipedia page titles using TagMe on this text.

- **GCD[2] Concepts**: Google Cross-Wiki Dictionary (GCD) is a string to concept mapping on the vast link structure of the web, created using anchor text from various pages across the web. A concept is an individual Wikipedia article. The text strings constitute the anchor texts that refer to these concepts. Thus, each anchor text string represents a concept. We extract common nouns, nominal+possessive, proper nouns, proper noun+possessive tag words from the tweet text using CMU POS Tagger [27] and query them on GCD anchor texts to get top ten unique concepts.

- **Hashtags**: We extract hashtags from the tweet.

- **Action Words**: We extract common nouns, nominal+possessive nouns and verbs from the tweet text using CMU POS Tagger [27].

- **Named Entities and Event Phrases**: We use Twitter NLP tools [3] to extract named entities and event phrases. Event phrases provide context to the entities occuring in the tweet. According to Ritter et al. [61], event phrases can consist of many different parts of speech as illustrated in the following examples.

  - Verbs: Apple to *Announce* iPhone 5 on October 4th?! YES!
  - Nouns: iPhone 5 *announcement* coming Oct 4th
  - Adjectives: WOOOHOO *NEW* IPHONE TODAY! CAN'T WAIT!

All concepts across the above dimensions together form the semantic representation of a tweet.

5.1.2 Similarity Metrics and Feature Design

The training data consists of tweets categorized in various sub-topics for each event entity. We use this training data to compute a semantic representation for the sub-topics with respect to each of the dimensions above. Semantic representation for a particular dimension for a sub-topic is computed as the top m most frequent phrases across phrases for that dimension for all tweets in the sub-topic. We use $m = 10$ for our experiments.

[2] **http://nlp.stanford.edu/pubs/crosswikis.pdf**
[3] https://github.com/aritter/twitter_nlp

Next, we obtain similarity between the tweet and the sub-topic across various concept dimensions, using multiple similarity metrics. We pick relevant similarity metrics for appropriate concept dimensions as described below. These similarity values are then used as features for the classifier. Similarity between a tweet and a sub-topic along a dimension d using a similarity metric s is computed as the weighted average of similarity between pairs of concept phrases (one each from the tweet and the sub-topic) as shown in Eq. 5.1. Here n is the number of concept phrases for the tweet along dimension d, w_i is the i^{th} concept phrase along dimension d in the tweet, r_j is the j^{th} concept phrase along dimension d in the sub-topic, $sim_s(w_i, r_j)$ is the similarity between w_i and r_j using similarity metric s, and $f(r_j)$ is the frequency of the concept phrase r_j in the sub-topic.

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{m} sim_s(w_i, r_j) \times f(r_j)}{n \times \sum_{j=1}^{m} f(r_j)} \tag{5.1}$$

Here is a list of similarity metrics we use for computing similarity values.

- **WordNet Similarity**: The Wu and Palmer WordNet Similarity measure [74] calculates relatedness between two words by considering the depths of the two synsets in the WordNet taxonomies, along with the depth of the longest common subsequence. We use WordNet Similarity as the measure to compute similarity along the "Action Words" dimension.

- **Wikipedia Miner Similarity**: This is calculated from the inlinks and outlinks of each Wikipedia page. Links that are common to both pages are used as evidence that they are related, while links that are unique to one or the other indicate the opposite [51]. We use this similarity measure to compute similarity along the "Tweet Concepts", "URL Meta Concepts", "URL Text Concepts", and "GCD Concepts" dimensions.

- **GCD Similarity**: GCD similarity between two terms is the ratio of numbers of documents in which these two terms co-occur in the top k retrieved documents when queried on GCD. For our experiments, we set k to 100. We use this similarity measure to compute similarity along the "Action Words", "Event Phrases", "Hashtags", and "Named Entities" dimensions.

- **String Similarity**: This is computed as 1-(string edit distance). We use the String Similarity measure to compute similarity along the "Event Phrases", "Action Words", and "Named Entities" dimensions.

5.1.3 Sub-topic Detection Methodology

Pre-processing: The dataset consists of both English and Spanish tweets. We convert all Spanish tweets to English using Bing Translation API [4]. We remove stopwords, perform stemming and lower case the tweets.

Training: For each event entity, we train a separate binary SVM classifier using RBF kernels. Each instance in the training data for the classifier corresponds to a (tweet, sub-topic) pair. A class label of 1 indicates that the tweet belongs to the sub-topic while 0 indicates otherwise. The features for the classifier are the similarity values computed as described in Sub-section 5.1.2 which in turn uses the semantic representation of the tweets and the sub-topics as discussed in Sub-section 5.1.1. We have a total of 12 features. The training data for the task helps us get positive (tweet, sub-topic) pairs for every sub-topic. By choosing a random sub-topic other than the associated sub-topic for every tweet, we get equal number of negative labels. For each event entity, we tune the SVM classifier parameters (C and γ) separately using 5-fold cross validation. Intuitively, γ defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. The C parameter trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly.

Besides building the classifier, concept phrases for every tweet across all dimensions are indexed in Lucene [47]. As detailed in the following, we use the Lucene index and the classifier to find the most relevant sub-topic for a tweet in the semantic space, i.e., the space of concept phrases across various dimensions.

Sub-topic Detection Methodology: Given a tweet for an event entity, we need to identify the sub-topic to which it belongs. The tweet can belong either to an already existing sub-topic or to a new sub-topic.

Model 1: For a new test tweet, we extract the concept phrases across various dimensions (as described in Sub-section 5.1.1) and query them on the Lucene index. We retrieve top K tweets from the query results. Since we already have the tweet to sub-topic mapping for the indexed tweets, we can compute the most frequent sub-topic among these K tweets which we refer to as the target sub-topic. Ties are resolved using the Lucene hit-score (or the relevance score). If the target sub-topic appears more than σK times in the query results, the tweet is assigned to the target sub-topic. We use $\sigma = \frac{2}{3}$ in our experiments. If the target sub-topic is less frequent in the query results, we compute a feature

[4] `http://datamarket.azure.com/dataset/bing/microsofttranslator`

vector for the (tweet, target sub-topic) pair as described in Sub-section 5.1.2. If the classifier returns a 1, the tweet is assigned to the target sub-topic, else a new sub-topic is created and the tweet is assigned to the new sub-topic. We index the semantic representation of the new tweet and also update the semantic representation for the corresponding sub-topic.

Model 2: Compared to Model 1, in Model 2 instead of considering most frequent sub-topic among top K query results as the target sub-topic we consider the one with the highest relevance score (computed as the sum of the relevance scores across the tweets related to the sub-topic from among the top K results) as the target sub-topic.

Model 3: In this approach, we maintain two Lucene indexes: a primary index and a secondary index. Primary index stores all tweets while the secondary index only stores those tweets that are assigned to topics with less than K tweets. When a new tweet arrives, processing is the same as in Model 1, and the index used is the primary index. However, if the classifier returns a 0, we perform lookups on secondary index and follow the same procedure. We index the semantic representation of the new tweet in the primary index, and also update the semantic representation for the corresponding sub-topic. If the sub-topic has size less than K, we add the semantic representation of the new tweet in the secondary index too. Sub-topics of size $\geq K$ are periodically removed from the secondary index. The hope is that maintaining two indexes would help smaller sub-topics to get selected as the target sub-topic with a higher chance compared to Model 1 where only primary index is maintained.

Model 4: Compared to Model 3, here we use relevance score and frequency for sorting sub-topics when querying the primary index and the secondary index respectively.

Model 5: Compared to Model 3, here we use frequency and relevance score for sorting sub-topics when querying the primary index and the secondary index respectively.

Model 6: Compared to Model 3, here we use relevance score for sorting sub-topics when querying both the primary index and the secondary index.

Model 7: Compared to Model 1, no classifier is used. If the target sub-topic is less frequent in the query results, a new sub-topic is created and the tweet is assigned to the new sub-topic.

Model 8: Compared to Model 2, no classifier is used. If the target sub-topic is less frequent in the query results, a new sub-topic is created and the tweet is assigned to the new sub-topic.

5.2 Extraction of Fact Triplets from Tweets

The second task that we address in this paper is the extraction of subject-predicate-object fact triplets. In Section 5.1, we categorized tweets related to an event into various sub-topics. In this section, we will discuss how we can identify fact triplets about the sub-topics from related tweets.

Previous methods tried to extract fact triplets from well structured English sentences and Wikipedia. We aim at extracting fact triplets from tweets. Compared to full English sentences, tweets contain incomplete sentences often missing one or more of the subject, object or predicate. The aim is to tag each tweet word with one of the labels: subject, predicate, object. However, each of these may span across multiple words. Hence, we modify the problem to that of tagging each word as one of 'B-S', 'I-S', 'B-P', 'I-P', 'B-O', or 'I-O' where 'B-S' denotes beginning of subject, 'I-S' denotes an intermediate subject word, and so on. The words which do not belong to any of the labels should get labeled as 'O', i.e., others.

We formulate this problem as a sequence tagging problem and use conditional random fields (CRFs) to tag each word in the tweet. We train a CRF using the following features.

- **POS Tags**: We used the CMU POS Tagger [27] to tag the tweets. The tweet is preprocessed by removing URL Links and retweet symbols and then send to POS Tagger.

- **Typed Dependencies**: We used the Stanford Dependency parser [19] to annotate the tweet with dependency relations.

- **Capitalization** : Based on whether the first letter of the word is capitalized or not, we assign a value 'C' or 'NC' to this binary feature.

- **Tweet Words**: Each word of the tweet is treated as a feature.

- **Named Entities**: We use Twitter NLP tools [61] to get the named entities from tweets. 'B-NE' is the label for the word representing the beginning of a named entity and 'I-NE' for intermediate named entity word.

- **Chunk Tags**: We use Twitter NLP tools [61] to get the chunk tags from tweets. 'B-NP' is the beginning of a chunk and 'I-NP' is the intermediate chunk.

5.3 Experiments

In this section, we will discuss the datasets, results and comparisons with other systems.

5.3.1 Datasets

For the sub-topic detection task, we experimented with the Replab (CLEF) 2013 [6] dataset. The dataset contains ∼700 tweets for training and ∼1500 tweets for testing for 61 entities. The training and test data sets are manually labeled by annotators who are trained and guided by experts. We used an SVM classifier with RBF kernel for every event entity. The C and the γ parameters were tuned by performing a grid search between 1 to 100 for C and 0 to 9 for C. For evaluation we use precision (P), recall (R) and F1.

For the fact extraction task, we took 500 random tweets from the RepLab 2013 [6] dataset. We used 400 tweets for training and 100 for testing. In the training phase, the tweets were manually annotated by three annotators. Each word in the tweet can take one of the following labels: B-S, I-S, B-P, I-P, B-O, I-O and O. We used maximum vote strategy to get the ground truth label for each word. We found the inter-annotator agreement score using Fleiss statistic [5] to be 0.366. While this value of the Fleiss' Kappa is considered as "fair agreement" among annotators, the relatively low value shows how difficult the task is even for humans. The labeled data is available at `http://tinyurl.com/factTripletExtractionData`. We use the standard precision (P), recall (R) and F1 measures for evaluation.

5.3.2 Results

5.3.2.1 Results for Sub-topic Detection

Table 5.1 shows the accuracy results for different models proposed in this paper for K=10 and K=15. (1) We experimented with building a separate classifier model for each entity and also with building a global model. As can be observed, there was no significant difference between the two cases. (2) For the "no classifier" models 7 and 8, while the precision is high, the recall is quite low leading a poor F measure. Thus, the classifier plays a critical role. (3) Accuracy measures do not differ much when K is fixed as 10 or 15. (4) Models 3 to 6 are not better than model 2, i.e., having a separate secondary index for smaller sized sub-topics does not help overall. Detailed analysis showed that having a secondary

[5] `http://en.wikipedia.org/wiki/Fleiss'_kappa`

	Individual Model for each Entity						Global Model for all Entities					
	K=10			K=15			K=10			K=15		
	P	R	F	P	R	F	P	R	F	P	R	F
Model 1	0.684	0.294	0.392	0.700	0.281	0.379	0.731	0.276	0.381	0.743	0.269	0.374
Model 2	0.682	0.297	0.397	0.695	0.298	0.398	0.724	0.288	0.394	0.728	0.290	0.396
Model 3	0.661	0.303	0.394	0.678	0.297	0.390	0.719	0.286	0.390	0.729	0.281	0.385
Model 4	0.679	0.297	0.395	0.695	0.293	0.394	0.720	0.285	0.390	0.724	0.288	0.393
Model 5	0.677	0.294	0.392	0.681	0.292	0.391	0.718	0.285	0.389	0.727	0.284	0.388
Model 6	0.679	0.293	0.392	0.681	0.299	0.396	0.720	0.285	0.390	0.723	0.287	0.392
Model 7	0.958	0.170	0.266	0.967	0.168	0.262	0.958	0.170	0.266	0.967	0.168	0.262
Model 8	0.962	0.160	0.256	0.968	0.164	0.258	0.962	0.160	0.256	0.968	0.164	0.258

Table 5.1 Accuracy Comparison for the Proposed Models for the Sub-topic Detection Task

index did increase the number of true positives but the effect was nullified by an increase in the number of false positives too. (5) Model 2 provides the best F measure of 0.398.

The average accuracy of the SVM sub-topic detection binary classifier was 0.865. We also experimented by using various combinations of the four similarity metrics. Table 5.3 shows the results when all four similarity metrics were used. However, we observed that Model 2 provides best F1 of 0.417 when only the "String Similarity" and "Wikipedia Miner Similarity" metrics are used.

We also wanted to identify which of the eight semantic dimensions contributes most to the accuracy of the system. Table 5.2 shows the accuracy results when one of the eight dimensions is removed from the semantic representation. The last row corresponds to the accuracy of the SVM sub-topic detection classifier. Note that removing the "GCD Concepts" or the "Tweet Concepts" dimensions causes the greatest decrease in F measure and also in the recall. Even the SVM classifier performs the worst when GCD concepts are not considered. This result clearly shows that the "GCD Concepts" and the "Tweet Concepts" provide the most important signal for sub-topic detection. However, note that the best classifier accuracy is observed with a combination of all the dimensions suggesting that all the dimensions play a complementary role.

In Table 5.3, we show comparisons of the proposed system with various state-of-the-art systems. We compare against multiple methods: *CSR* is the Concept Space Representation approach proposed in [55]; *Baseline* is the memory-based learning baseline supplied by RepLab, and *UNED_ORM, REINA, LIA* are the teams from RepLab 2013 [6]. Clearly, the proposed system beats all other previous approaches by a significant margin.

Finally, Table 5.4 shows a few examples of tweets and the associated sub-topics.

	All	All-Event Phrases	All-Hashtags	All-Named Entities	All-Action Words	All-GCD Concepts	All-Tweet Concepts	All-URL Meta Concepts	All-URL Text Concepts
P	0.695	0.647	0.658	0.625	0.721	0.765	0.677	0.674	0.691
R	0.298	0.308	0.307	0.309	0.282	0.259	0.279	0.297	0.302
F	0.398	0.400	0.404	0.394	0.387	0.368	0.376	0.396	0.403
SVM Accuracy	0.865	0.679	0.734	0.680	0.649	0.643	0.855	0.784	0.752

Table 5.2 Accuracy of Sub-topic Detection when one of the Dimensions is removed

System	**Avg. P**	**Avg. R**	**Avg. F**
Model 2	0.618	0.330	**0.417**
CSR [55]	0.304	0.516	0.339
UNED_ORM	0.460	0.320	0.330
REINA	0.320	0.430	0.290
LIA	0.220	0.350	0.250
Baseline	0.150	0.220	0.170

Table 5.3 Accuracy Comparison of Systems for Sub-topic Detection

5.3.2.2 Results for Extraction of Fact Triplets

Table 5.5 shows that the proposed system for extraction of fact triplets beats the two state-of-the-art systems by a significant margin with respect to subject, predicate as well as object extraction. This is mainly because the other two systems are built for full English sentences and hence do not perform well on noisy tweet text.

The proposed system uses 6 features. We tested all possible 64 combinations of feature inclusion. We observed that the highest overall F measure is obtained when all features were used suggesting the features are all complementary. We also experimented by removing each feature from the complete feature set to understand the importance of the features. Table 5.6 shows that the chunk tags are the most important features.

Finally, Table 5.7 shows a few examples of tweets and the extracted fact triplets, and Table 5.8 shows a few examples of tweets where the system predicted incorrectly.

Tweet	Sub-topic
The volvo has a flat battery - it won't jump start for some reason	customer complain
@naclarsigh @themartygill lol, just went to craic open the volvo, half it missing lol, hows the grouse #somebodybeatmetoit	volvo owners comments / mention
For Celtics fans, that was like having a beloved 1978 Volvo with 200,000 miles finally break down, 50 yards from home.	jokes
@FabSeanJay If you want to know how crazy Nigerians are - go to Tudu & see the old ass Volvo & BMW 5series that run Accra-Lagos-Accra	car race videos / comments
@GEAGarratt BMW hand over 200 + electric vehicles to the Olympics	BMW official partner of the olympics
BMW M3 Limited Edition for UK #1 pic.twitter.com/d561hkkM	New products
2002 bmw 3 series? or 2002 bmw 5 series?. #BMW http://bit.ly/mZ4Lwp	Comparison between BMW models
Why are BMW m3's soooo fucking sexy??	Opinion about BMW
2007 BMW X5 4.8i (A Promise to Do More) $35993 - Get This BMW FREE - Contact Me	BMW vehicles for sale

Table 5.4 Example Tweet and Sub-topic Pairs

System	Subject			Predicate			Object			Overall		
	P	R	F	P	R	F	P	R	F	P	R	F
Proposed System	0.64	0.37	**0.47**	0.62	0.41	**0.49**	0.46	0.24	**0.32**	0.58	0.34	**0.43**
Reverb [24]	0.52	0.29	0.37	0.17	0.07	0.10	0.35	0.18	0.23	0.35	0.17	0.23
Triplet Extraction from Sentences [63]	0.51	0.35	0.41	0.45	0.24	0.31	0.19	0.13	0.16	0.38	0.24	0.29

Table 5.5 Accuracy Comparison of Systems for Extraction of Fact Triplets

5.4 Conclusion

In this chapter, we studied the problems of sub-topic detection and extraction of fact triplets from tweets. We designed interesting ways of semantically representing tweets, and further combined them with relevant similarity metrics to define features for the sub-topic detection classifier. We exploited various linguistic analysis techniques like Part-Of-Speech (POS) tagging, dependency analysis, etc. to learn a CRF model for extraction of fact triplets. Comparisons with various state-of-the-art mechanisms for sub-topic detection on a dataset of $\sim$134K tweets show that the proposed approach outperforms others by a significant margin providing a best F1 score of 0.417. We found that "String Similarity" and "Wikipedia Miner Similarity" metrics are the most important similarity measures. Also the "GCD Con-

Features	Subject			Predicate			Object			Overall		
	P	R	F	P	R	F	P	R	F	P	R	F
All	0.64	0.37	0.47	0.62	0.41	0.49	0.46	0.24	0.32	0.58	0.34	0.43
All - Tweet Words	0.60	0.45	0.52	0.57	0.40	0.47	0.36	0.24	0.29	0.52	0.37	0.43
All - Capitalization	0.72	0.38	0.49	0.60	0.39	0.47	0.39	0.22	0.29	0.57	0.33	0.42
All - Named Entities	0.65	0.34	0.44	0.61	0.39	0.47	0.40	0.20	0.27	0.56	0.31	0.40
All - Typed Dependencies	0.65	0.33	0.43	0.67	0.41	0.51	0.37	0.16	0.22	0.58	0.31	0.40
All - POS Tags	0.62	0.36	0.45	0.66	0.39	0.49	0.34	0.15	0.21	0.57	0.30	0.39
All - Chunk Tags	0.59	0.28	0.38	0.58	0.33	0.42	0.28	0.10	0.15	0.51	0.24	0.33

Table 5.6 Accuracy of Extraction of Fact Triplets when one of the Features is removed

Tweet	Fact Triplets
Oxford street is so stressful . #ratrace	(Oxford street, stressful, –)
I just ousted lee W . as the mayor of Oxford Street on @foursquare !	(I, ousted, lee W.)
Selena Gomez will be signing copies of her new album ' When The Sun Goes Down ' at HMV Oxford Circus , London on Tuesday 5th July from 5pm	(Selena Gomez, signing copies, new album)
Mazda Raceway will be here Sunday selling tickets from 12 - 3pm .	(Mazda Raceway, selling, tickets)
Mazda unveils 2014 Mazda6 ahead of Moscow auto show	(Mazda, unveils, Mazda6)

Table 5.7 Fact Triplets Extracted by the Proposed System

Tweet	Fact Triplets Predicted	Actual
Breaking News : Former United defender Wes Brown arrested after he leaves Anfield with 35mil in his pocket	(,{arrested,leaves},Anfield)	(Former United defender Wes Brown,{arrested,leaves},Anfield)
Real Madrid won their first Spanish Supercup against Barcelona in 1988.	(Madrid,won,)	(Real Madrid,won,first Spanish Supercup)
Firefighters battle officers in weight-loss challenge : Detective Dave Horn of Tiffin Police Department said he c ...	(Firefighters battle officers,,)	({Firefighters,Detective Dave Horn},battle,{officers,weight-loss challenge})
Breaking : Dow Jones suffers sixth-worst point drop in history	(,Breaking,history)	(Dow Jones,suffers,sixth-worst point drop)
With 35 minutes to go in MotoGP FP2 , Simoncelli is leading ahead of Lorenzo and Dovizioso	(Simoncelli,{leading,ahead},)	({MotoGP FP2,Simoncelli}, {leading,ahead},{Lorenzo,Dovizioso})

Table 5.8 Fact Triplets incorrectly predicted by the System

cepts" and the "Tweet Concepts" dimensions were found to be the most effective way of semantically representing tweets. The proposed CRF-based fact triplets extraction approach also beats the baselines significantly providing an overall F1 of 0.43. Chunk tags turned out to be the most important feature for this task. In the future, we would like to explore the usage of the fact triplets for effective summarization and knowledge base population for fresh event entities as applications of the proposed approaches. We would also like to design algorithms for effective hierarchical categorization of event sub-topics.

Chapter 6

Conclusions

6.1 Conclusions

In this thesis, we have explored a novel approach by exploiting the semantic and concept based representations of tweets for sub-topic clustering. In the *SSR* based approach, we used keywords (WordNet synonyms, URL keywords), keyphrases and Wikipedia title matching (as criterion for creating new cluster) as features. To handle the issue of overmatching of keywords due to WordNet usage, we propose the *CSR* based approach. In the *CSR* based approach, we used TweetConcepts, URLConcepts and keyphrases as features. We observe that for a few tweets, the concepts retrieved from TagMe were relatively inaccurate, and this resulted in lower reliability for CSR. SSR has higher reliability compared to CSR because most of the relationships predicted by the system are also found in gold standard. However, the sensitivity is lower for SSR because the number of discovered relationships in the system are less, as excessive keyword matching (probably because of WordNet usage) caused some sub-topics to merge into a single sub-topic. The higher sensitivity of CSR as compared to that of SSR is because CSR has higher coverage of relationships over the gold standard. CSR based approach maintains the consistency in both, identifying the number of sub-topics as well as their presence in the gold dataset for each entity. We maintain the purity of clusters by periodically cleaning up the clusters. Experiments on the RepLab (at CLEF 2013) dataset showed that the proposed approach achieves significant performance gains over the baseline and other systems using metrics like Reliability, Sensitivity and $F1$ measure.

Later we extended this study by incorporating the similarity between concepts in the *CSR* based approach. We also try to extract fact triplets from the sub-topics formed in the first step. We studied the problems of sub-topic detection and extraction of fact triplets from tweets. We designed interesting ways of semantically representing tweets, and further combined them with relevant similarity metrics to

define features for the sub-topic detection classifier. We exploited various linguistic analysis techniques like Part-Of-Speech (POS) tagging, dependency analysis, etc. to learn a CRF model for extraction of fact triplets. Comparisons with various state-of-the-art mechanisms for sub-topic detection on a dataset of $\sim$134K tweets show that the proposed approach outperforms others by a significant margin providing a best F1 score of 0.417. We found that "String Similarity" and "Wikipedia Miner Similarity" metrics are the most important similarity measures. Also the "GCD Concepts" and the "Tweet Concepts" dimensions were found to be the most effective way of semantically representing tweets. The proposed CRF-based fact triplets extraction approach also beats the baselines significantly providing an overall F1 of 0.43. Chunk tags turned out to be the most important feature for this task.

We have also studied the problem of extracting structured information for natural disaster events from Twitter. Specifically we focused on extracting attribute-value pairs and fact triplets. We solved this problem by first performing linguistic analysis like part of speech tagging and dependency parsing. After that we proposed three novel algorithms for numeric attribute-value extraction, textual attribute-value extraction, and fact triplet extraction. We also proposed an algorithm to map the extracted attributes to a schema for the corresponding event type. Experiments on $\sim$58000 tweets for 20 events show the effectiveness of the proposed approach. We summarize the results across all events for all the three settings: Only Web-links, Only Tweets, and Tweets + Web-links. The precision for Only Web-links is more because web text is more structured compared to the tweets. This results in accurate mapping of a value to a particular attribute. However, the recall is less as much of the information was not expressed in the top few Web-links, so less number of attribute-value pairs were discovered. The recall increased when both tweets and web-links were used to extract attribute-value pairs as expected. This is because some attributes were expressed in tweets and some were expressed in web-links, so in combination it resulted in more attribute-value pairs. Overall, we obtained best F1 with Tweets + Web-links. Such a structured event summary can significantly improve the relevance of the displayed results by providing key information about the event to the user without any extra clicks.

Further extensions to the thesis can be done in the following directions:

- We can do a second pass run on the formed clusters to further improve the quality of clusters. Cluster splitting and merging can be done based on the inter-cluster and intra-cluster similarity to obtain fine-grained clusters.

- Efforts can also be put in the direction of clustering jokes, irony and sarcasm related tweets separately.

- Non-informative and spam tweets can be discarded by performing a filtering step before clustering to improve the purity of the clusters.

- In extracting Tweet concepts and URL concepts, we can build our own entity linking system. The improvement in accuracy of entity linking system leads to better accuracy in finding sub-topics.

- Currently, the most frequently occuring kephrase and keyword of a sub-topic is used for labelling. We can also explore a better way in making use of the fact-triplets to better label the sub-topic.

- We can also design algorithms for effective hierarchical categorization of event sub-topics.

- We can explore the usage of the fact triplets for effective summarization and knowledge base population for fresh event entities as applications of the proposed approaches.

- We can also generalize the approaches of numeric attribute-value extraction, textual attribute-value extraction, and fact triplet extraction to other events on Twitter.

- We can improve the performance of fact triplet extraction system by incorporating more features like Previous and next word for the current word, Head word of the current token, pos of the head word token, postags of current previous and next tokens, semantic role label of the word.

- We can add these fact-triplets as extra features in finding the sub-topics along with the proposed features.

- Similar statistical models can be built for Numeric attribute-value extraction as done for fact-triplets.

- In Extracting Structured information from natural disaster events, we can first find the sub-topics and later extract structured information from the sub-topics found. This may give better mapping of candidate attribute-value pairs to target schema attribute-value pairs.
- Similar to a Semantic Web, a Semantic Twitter resource can be developed if the information extracted is either represented in RDF or OWL. The extracted the Attribute-value pairs and Fact-Triplets from Twitter events, can act as a source of semantically linking the events on Twitter. It also improves the recall in search results when queried by a user and will be capable of answering even more structured and complex queries.

Related Publications

- Sandeep Panem, Romil Bansal, Manish Gupta, Vasudeva Varma. **Entity Tracking in Real-Time using Sub-Topic Detection on Twitter**.*The 36th European Conference on Information Retrieval (ECIR 2014)*, Amsterdam, Netherlands.

- Sandeep Panem, Manish Gupta, Vasudeva Varma. **Structured Information Extraction from Natural Disaster Events on Twitter**. *The 5th Intl. Workshop on Web-scale Knowledge Representation, Retrieval and Reasoning (Web-KR 2014)*, Nov 3 - 7, 2014. Shanghai, China.

Bibliography

[1] F. Abel, I. Celik, G.-J. Houben, and P. Siehndel. Leveraging the Semantics of Tweets for Adaptive Faceted Search on Twitter. In *International Semantic Web Conference*, pages 1–17, 2011.

[2] L. Aiello, G. Petkos, C. Martin, D. Corney, S. Papadopoulos, R. Skraba, A. Goker, I. Kompatsiaris, and A. Jaimes. Sensing trending topics in twitter. *Multimedia, IEEE Transactions on*, pages 1268–1282, 2013.

[3] E. Alfonseca, K. Filippova, J.-Y. Delort, and G. Garrido. Pattern Learning for Relation Extraction with a Hierarchical Topic Model. In *Proc. of the* 50^{th} *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 54–59. Association for Computational Linguistics, 2012.

[4] E. Alfonseca, M. Pasca, and E. Robledo-Arnuncio. Acquisition of Instance Attributes via Labeled and Related Instances. In *Proc. of the* 33^{rd} *Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 58–65. ACM, 2010.

[5] E. Amigó, A. Corujo, J. Gonzalo, E. Meij, and M. de Rijke. Overview of RepLab 2012: Evaluating Online Reputation Management Systems. In *Proc. of the* 3^{rd} *Intl. Conf. of the CLEF Initiative*, 2012.

[6] E. Amigó, J. C. de Albornoz, I. Chugur, A. Corujo, J. Gonzalo, T. Martín-Wanton, E. Meij, M. de Rijke, and D. Spina. Overview of RepLab 2013: Evaluating Online Reputation Monitoring Systems. In *Proc. of the* 4^{th} *Intl. Conf. of the CLEF Initiative*, pages 333–352, 2013.

[7] E. Amigó, J. Gonzalo, and F. Verdejo. A General Evaluation Measure for Document Organization Tasks. In *Proc. of the* 36^{th} *Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 643–652, 2013.

[8] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open Information Extraction for the Web. In *Proc. of the* 20^{th} *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, volume 7, pages 2670–2676, 2007.

[9] K. Bellare, P. P. Talukdar, G. Kumaran, F. Pereira, M. Liberman, A. McCallum, and M. Dredze. Lightly-Supervised Attribute Extraction. In *Proc. of the Neural Information Processing Systems (NIPS) 2007 Workshop on Machine Learning for Web Search*, 2007.

[10] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 2003.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research (JMLR)*, 3:993–1022, Mar 2003.

[12] M. Cataldi, L. Di Caro, and C. Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, pages 4:1–4:10. ACM, 2010.

[13] A. X. Chang and C. D. Manning. SUTime: A Library for Recognizing and Normalizing Time Expressions. In *Proc. of the 2012 Intl. Conf. on Language Resources and Evaluation (LREC)*, pages 3735–3740, 2012.

[14] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[15] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi. Short and tweet: Experiments on recommending content from information streams. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1185–1194. ACM, 2010.

[16] N. Dalvi, R. Kumar, and B. Pang. Object Matching in Tweets with Spatial Models. In *Proc. of the fifth ACM Intl. Conf. on Web Search and Data Mining (WSDM)*, pages 43–52, 2012.

[17] K. S. Dave and V. Varma. Pattern Based Keyword Extraction for Contextual Advertising. In *Proc. of the* 19^{th} *ACM Intl. Conf. on Information and Knowledge Management (CIKM)*, pages 1885–1888, 2010.

[18] M.-C. de Marneffe and C. D. Manning. The Stanford Typed Dependencies Representation. In *Proc. of the COLING Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, 2008.

[19] M.-C. De Marneffe and C. D. Manning. The Stanford Typed Dependencies Representation. In *Proc. of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, 2008.

[20] Q. Diao, J. Jiang, F. Zhu, and E.-P. Lim. Finding bursty topics from microblogs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 536–544, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[21] S. Dietze, D. Maynard, E. Demidova, T. Risse, W. Peters, K. Doka, and Y. Stavrakas. Entity Extraction and Consolidation for Social Web Content Preservation. In *Proc. of the* 2^{nd} *Intl. Workshop on Semantic Digital Archives (SDA)*, pages 18–29, 2012.

[22] Y. Du, Y. He, Y. Tian, Q. Chen, and L. Lin. Microblog bursty topic detection based on user relationship. In *Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International*, volume 1, pages 260–263, Aug 2011.

[23] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-scale Information Extraction in Knowitall: (Preliminary Results). In *Proc. of the* 13^{th} *Intl. Conf. on World Wide Web (WWW)*, pages 100–110, 2004.

[24] A. Fader, S. Soderland, and O. Etzioni. Identifying Relations for Open Information Extraction. In *Proc. of the Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1535–1545. Association for Computational Linguistics, 2011.

[25] P. Ferragina and U. Scaiella. TagMe: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proceedings of the* 19^{th} *ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1625–1628. ACM, 2010.

[26] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. Part-of-speech Tagging for Twitter: Annotation, Features, and Experiments. In *Proc. of the* 49^{th} *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT)*, pages 42–47, 2011.

[27] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. Part-Of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *Proc. of the* 49^{th} *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 42–47, 2011.

[28] M. Grinev, M. Grineva, A. Boldakov, L. Novak, A. Syssoev, and D. Lizorkin. Sifting micro-blogging stream for events of user interest. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 837–837, 2009.

[29] M. Gupta, R. Li, and K. Chang. Tutorial: Towards a Social Media Analytics Platform: Event Detection and User Profiling for Microblogs. In *Proc. of the* 23^{rd} *Intl. Conf. on World Wide Web (WWW)*, 2014.

[30] M. Gupta, R. Li, and K. C.-C. Chang. Tutorial: Towards a Social Media Analytics Platform: Event Detection and User Profiling for Twitter. In *Proc. of the* 23^{rd} *Intl. World Wide Web Conf. (WWW)*, pages 193–194, 2014.

[31] M. Gupta, P. Zhao, and J. Han. Evaluating Event Credibility on Twitter. In *Proc. of the 2012 SIAM Intl. Conf. on Data Mining (SDM)*, pages 153–164, 2012.

[32] Q. He, K. Chang, and E.-P. Lim. Using burstiness to improve clustering of topics in news streams. pages 493–498, 2007.

[33] T. Hofmann. Probabilistic Latent Semantic Indexing. In *Proc. of the* 22^{nd} *Annual Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 50–57, 1999.

[34] Y. Hu, Y. Qian, H. Li, D. Jiang, J. Pei, and Q. Zheng. Mining Query Subtopics from Search Log Data. In *Proc. of the* 35^{th} *Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 305–314, 2012.

[35] T. Hua, F. Chen, L. Zhao, C.-T. Lu, and N. Ramakrishnan. STED: Semi-supervised Targeted-interest Event Detection in Twitter. In *Proc. of the 19th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 1466–1469, 2013.

[36] K. Kireyev, L. Palen, and K. Anderson. Applications of Topics Models to Analysis of Disaster-Related Twitter Data. In *NIPS Workshop on Applications for Topic Models: Text and Beyond*, Dec 2009.

[37] J. Kleinberg. Bursty and hierarchical structure in streams. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 91–101, 2002.

[38] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289. Morgan Kaufmann Publishers Inc., 2001.

[39] T. Lee, Z. Wang, H. Wang, and S. won Hwang. Attribute Extraction and Scoring: A Probabilistic Approach. In *Proc. of the 2013 IEEE* 29^{th} *Intl. Conf. on Data Engineering (ICDE)*, pages 194–205, 2013.

[40] R. Li, K. H. Lei, R. Khadiwala, and K.-C. Chang. TEDAS: A Twitter-based Event Detection and Analysis System. In *Proc. of the 2012 IEEE 28th Intl. Conf. on Data Engineering (ICDE)*, pages 1273–1276, 2012.

[41] J. Liu. Answering structured queries on unstructured data. In *In WebDB*, pages 25–30, 2006.

[42] A. Louis and T. Newman. Summarization of Business-Related Tweets: A Concept-Based Approach. In *Proc. of the* 24^{th} *Intl. Conf. on Computational Linguistics (COLING)*, pages 765–774, 2012.

[43] P. Löw. Natural Catastrophes in 2012 Dominated by U.S. Weather Extremes. http://www.worldwatch.org/natural-catastrophes-2012-dominated-us-weather-extremes-0, 2013.

[44] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller. Processing and Visualizing the Data in Tweets. *ACM SIGMOD Record*, 40(4):21–27, 2012.

[45] V. Markman. Unsupervised discovery of fine-grained topic clusters in twitter posts. In *Analyzing Microtext*, AAAI Workshops. AAAI, 2011.

[46] M. Mathioudakis and N. Koudas. Twittermonitor: Trend Detection over the Twitter Stream. In *Proc. of the 2010 ACM SIGMOD Intl. Conf. on Management of Data (SIGMOD)*, pages 1155–1158, 2010.

[47] M. McCandless, E. Hatcher, and O. Gospodnetic. *Lucene in Action, Second Edition: Covers Apache Lucene 3.0*. Manning Publications Co., Greenwich, CT, USA, 2010.

[48] D. Metzler, C. Cai, and E. Hovy. Structured Event Retrieval over Microblog Archives. In *Proc. of the 2012 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 646–655, 2012.

[49] M. Michelson and S. A. Macskassy. Discovering users' topics of interest on twitter: A first look. In *Proceedings of the Fourth Workshop on Analytics for Noisy Unstructured Text Data*, AND '10, pages 73–80, New York, NY, USA, 2010. ACM.

[50] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, pages 39–41, 1995.

[51] D. Milne and I. H. Witten. An Open-source Toolkit for Mining Wikipedia. *Artificial Intelligence*, 194:222–239, 2013.

[52] N. Nakashole, G. Weikum, and F. Suchanek. PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *Proc. of the 2012 Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1135–1145. Association for Computational Linguistics, 2012.

[53] B. O'Connor, M. Krieger, and D. Ahn. TweetMotif: Exploratory Search and Topic Summarization for Twitter. In *Proc. of the* 4^{th} *Intl. Conf. on Weblogs and Social Media (ICWSM)*, 2010.

[54] A. Pal and S. Counts. Identifying topical authorities in microblogs. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pages 45–54. ACM, 2011.

[55] S. Panem, R. Bansal, M. Gupta, and V. Varma. Entity Tracking in Real-Time Using Sub-topic Detection on Twitter. In *Proc. of the* 36^{th} *European Conference on Advances in Information Retrieval (ECIR)*, pages 528–533, 2014.

[56] S. Panem, M. Gupta, and V. Varma. Structured Information Extraction from Natural Disaster Events on Twitter. In *Proc. of the* 5^{th} *Intl. Workshop on Web-scale Knowledge Representation, Retrieval and Reasoning (Web-KR)*, 2014.

[57] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web*, pages 91–100, 2008.

[58] K. Puniyani, J. Eisenstein, S. Cohen, and E. P. Xing. Social links from latent topics in microblogs. In *Proceedings of NAACL Workshop on Social Media*, Los Angeles, 2010.

[59] D. Ramage, S. T. Dumais, and D. J. Liebling. Characterizing microblogs with topic models. In W. W. Cohen and S. Gosling, editors, *ICWSM*, 2010.

[60] J. Reisinger and M. Pasca. Low-Cost Supervision for Multiple-Source Attribute Extraction. In *Proc. of the 2009 Conf. on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 382–393, 2009.

[61] A. Ritter, O. Etzioni, S. Clark, et al. Open Domain Event Extraction from Twitter. In *Proc. of the 18th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 1104–1112, 2012.

[62] D. Rusu, L. Dali, B. Fortuna, M. Grobelnik, and D. Mladenic. Triplet Extraction From Sentences. In *Proc. of the* 10^{th} *Intl. Multiconf. "Information Society - IS 2007"*, volume A, pages 218–222, 2007.

[63] D. Rusu, L. Dali, B. Fortuna, M. Grobelnik, and D. Mladenic. Triplet Extraction From Sentences. In *Proc. of the* 10^{th} *Intl. Multiconf. Information Society (IS)*, volume A, pages 218–222, 2007.

[64] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake Shakes Twitter Users: Real-Time Event Detection by Social Sensors. In *Intl. World Wide Web Conference (WWW)*, pages 851–860, 2010.

[65] S. Sarawagi. Information Extraction. *Foundations and Trends in Databases*, 1(3):261–377, 2008.

[66] B. Sharifi, M.-A. Hutton, and J. Kalita. Summarizing Microblogs Automatically. In *Human Language Technologies: The 2010 Annual Conf. of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 685–688, 2010.

[67] S. Song, Q. Li, and H. Bao. Detecting dynamic association among twitter topics. In *WWW (Companion Volume)*, pages 605–606, 2012.

[68] K. Starbird, L. Palen, A. L. Hughes, and S. Vieweg. Chatter on the Red: What Hazards Threat Reveals about the Social Life of Microblogged Information. In *Computer Supported Cooperative Work (CSCW)*, pages 241–250, 2010.

[69] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen. Microblogging During Two Natural Hazards Events: What Twitter may Contribute to Situational Awareness. In *Intl. Conf. on Human Factors in Computing Systems (CHI)*, pages 1079–1088, 2010.

[70] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: Finding topic-sensitive influential twitterers. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, pages 261–270. ACM, 2010.

[71] Y. W. Wong, D. Widdows, T. Lokovic, and K. Nigam. Scalable Attribute-Value Extraction from Semi-structured Text. In *Proc. of the 2009 IEEE Intl. Conf. on Data Mining (ICDM) Workshops*, pages 302–307, 2009.

[72] F. Wu and D. S. Weld. Automatically Refining the Wikipedia Infobox Ontology. In *Proc. of the* 17^{th} *Intl. Conf. on World Wide Web (WWW)*, pages 635–644. ACM, 2008.

[73] F. Wu and D. S. Weld. Open Information Extraction Using Wikipedia. In *Proc. of the* 48^{th} *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 118–127, 2010.

[74] Z. Wu and M. Palmer. Verbs Semantics and Lexical Selection. In *Proc. of the* 32^{nd} *Annual Meeting on Association for Computational Linguistics (ACL)*, pages 133–138, 1994.

[75] T. Xu and D. W. Oard. Wikipedia-based topic clustering for microblogs. *Proceedings of the American Society for Information Science and Technology*, 48(1):1–10, 2011.

[76] W. Xu, S. Feng, L. Wang, D. Wang, and G. Yu. Detecting hot topics in chinese microblog streams based on frequent patterns mining. In F. L. Wang, J. Lei, Z. Gong, and X. Luo, editors, *WISM*, volume 7529 of *Lecture Notes in Computer Science*, pages 637–644. Springer, 2012.

[77] J. Yang and J. Leskovec. Patterns of Temporal Variation in Online Media. In *Proc. of the* 4^{th} *ACM Intl. Conf. on Web Search and Data Mining (WSDM)*, pages 177–186. ACM, 2011.

[78] Z. Zhai, B. Liu, H. Xu, and P. Jia. Clustering product features for opinion mining. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pages 347–354, 2011.

[79] A. Zubiaga, D. Spina, V. Fresno, and R. Martínez. Classifying trending topics: A typology of conversation triggers on twitter. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 2461–2464, 2011.