Linking Event References in Online News Streams to a Global Repository

Master Thesis

by

Igor Shevchenko

Submitted to the Faculty IV, Electrical Engineering and Computer Science
Database Systems and Information Management Group
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

as part of the ERASMUS MUNDUS programme IT4BI

at the

TECHNISCHE UNIVERSITÄT BERLIN

July 15, 2015

Thesis Advisors:
Alan Akbik, M.Sc.
Christoph Boden, M.Sc.

Thesis Supervisor:
Prof. Dr. Volker Markl

Linking Event References in Online News Streams to a Global Repository

by

Igor Shevchenko

Submitted to the Faculty IV, Electrical Engineering and Computer Science
on July 15, 2015, in partial fulfillment of the requirements for the
degree of Master of Science in Computer Science
as part of the ERASMUS MUNDUS programme IT4BI

Abstract

This thesis proposes a way to build a global repository of events from exploiting topically related clusters of news articles. An automated approach is based on the hierarchical grouping of events and event mentions. Firstly, event mentions are extracted on the sentence level from individual news articles using *OpenIE* techniques. Secondly, highly similar event mentions that refer to the same point in time are grouped within a news cluster to form unique cluster events. Finally, highly similar cluster events are grouped within the repository to form unique global events. The proposed approach is intended to extract an unrestricted set of open-domain events and all their possible textual references. Several grouping methods are implemented and evaluated. Evaluation is performed using two evaluation corpuses: adapted Event Coreference Bank Plus (ECB+) corpus and manually annotated real-word corpus. Experiments indicate the strong potential for the construction of a global repository of events. The cases in which an algorithm fails are illustrated and ways for addressing sources of errors are described.

Thesis Advisors: Alan Akbik, M.Sc., Christoph Boden, M.Sc.
Thesis Supervisor: Prof. Dr. Volker Markl

Das Vernetzen einzelner Erwähnungen von Ereignissen in einem Online-Nachrichtenstrom zu einem globalen Speicher

von

Igor Shevchenko

Eingereicht bei der. Fakultät IV für Elektrotechnik und Informatik, 15. Juli 2015,
Arbeit Eingereicht in teilweise Erfüllung der Anforderungen an die Grad Master Informatik im Rahmen des Programms Erasmus Mundus IT4BI

Zusammenfassung

Diese Masterarbeit bietet eine Lösung, um die Ereignisse thematisch zusammen zu bringen, indem thematisch miteinander verbundene Gruppen von Nachrichten analysiert werden. Ein automatisierter Ansatz basiert auf der hierarchischen Gruppierung von Ereignissen und einzelnen Erwähnungen von Ereignissen. Zunächst werden die einzelnen Erwähnungen von Ereignissen auf den Nachrichtenartikeln mit Hilfe von der OpenIE Technologie ausgenommen. Im zweiten Schritt werden ähnliche Erwähnungen von Ereignissen, die zum gleichen Zeitpunkt gehören, einer Cluster-Gruppierung zugeordnet, welche durch einzigartige Ereignisse gebildet worden ist. Schließlich werden ähnliche Cluster in einen Speicher gruppiert, welche nur einzigartigen Ereignisse angehören. Mithilfe des etwaigen Ansatzes wird ermöglicht, eine uneingeschränkte Reihe von den freizugänglichen (open-domain) Ereignissen und allen möglichen Textverweisen zu extrahieren. Mehrere Gruppierungsverfahren werden angewendet und evaluiert. Die Auswertung wird unter Verwendung von zwei Bewertungs-Korps durchgeführt: das angepasste Event Coreference Bank Plus (ECB+) Korpus und das manuell annotierte Echt Wort Korpus. Experimente bestätigen ein sehr großes Potenzial für den Aufbau eines globalen Speichers von Ereignissen. Die Fälle, in denen der Algorithmus nicht erfolgreich funktioniert hat, werden dargestellt und es werden die Methoden zur Fehlerbeseitigung beschrieben.

Thesis Advisors: Alan Akbik, M.Sc., Christoph Boden, M.Sc.
Thesis Supervisor: Prof. Dr. Volker Markl

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Berlin, July 15, 2015

Igor Shevchenko

To Lucy, the most incredible miracle in my life

Acknowledgments

I would like to thank my advisors Alan Akbik and Christoph Boden for the patience, constructive support and assistance. They are very enthusiastic people with deep interest in science with whom I have discussed many research ideas and problems. I greatly appreciate the unique way they encouraged me to reach new heights in my research.

Furthermore, I would like to express my special thanks to Prof. Dr. Volker Markl and Dr. Ralf Kutsche for the continuous supervision in all academic questions and provision of the enjoyable environment in the DIMA group at the Technische Universität Berlin. I would also like to extend my thanks to the members of the DIMA group for the technical support of the implementation part of the thesis and provision of the news crawler.

I wish to give an acknowledgement to the board members of IT4BI program, Erasmus Mundus and The European Union who gave me an opportunity to study high-quality joint master degree courses in three different European universities.

I wish to thank all fellow students of IT4BI program. Thanks to Anil, Andres, Ricardo, Tamara, Alexey, Miruna, Amit, Karim, Jake, Elena, Maximiliano, Kofi, Rizkallah, Varunya, Redion, Sergi, Hung, Poly, David, Sehrish and Oky for the friendship, support, wonderful discussions and inspirations during our challenging study.

I'm grateful to my family for the great support and continuous motivation. Thank you for the encouragement of my academic pursuit, understanding, love, and sacrifices for my education abroad. This thesis would not have been possible otherwise.

Contents

Chapter 1

Introduction

This thesis proposes a way to build a global repository of events from exploiting topically related clusters of news articles. The proposed approach is novel in the sense that it utilizes the whole news clusters as single units instead of working with isolated articles alone. Firstly, *event mentions* are extracted by utilizing day-level non-ambiguous temporal expressions within the text of news articles. Secondly, extracted *event mentions* are grouped within the news cluster to form *unique cluster events*. Finally, extracted cluster events are grouped within the whole repository to obtain *unique global events*. The extracted global events represent the output data and can be used to give an overview of the world situation.

The approach described in this thesis is intended to work on a sentence level and to be able to extract an unrestricted set of all possible events. This is achieved by using open-domain event extraction and by applying *OpenIE* techniques. Described approach does not require any training data or human supervision and can be applied on the large scale data.

This chapter introduces the thesis and the method for building a repository of events. Section 1.1 gives a motivation for creating a repository of events in comparison with existing techniques for managing an information overflow. Section 1.2 describes the problem and provides a brief overview of the method used in this thesis for event extraction. Main challenges and characteristics of input data are described. Goal of the thesis and implementation steps are listed in Section 1.3. Contributions of the thesis are highlighted in Section 1.4. Section 1.5 provides an outline of the thesis.

1.1 Motivation

Large amount of information is published daily on news portals. Many of the published news are rapidly spread across media providers by duplicating the news article with little or even no additional information. It is practically impossible for a single person to read and track published information altogether. With the shift toward electronic way of delivering news, news providers are capable to frequently update and add news articles, thus, flooding the reader with more and more information.

In addition to the high volume of news articles, most events are often repeatedly mentioned in the text with different surface forms. Some events are reported for a continuous period of time mentioning not only the new information, but also repeating the information of previous days. Worldwide news providers can report events with a delay. At the same time, local news providers can already publish new event details, thus, mixing the overall event view of online news aggregators.

This overwhelming amount of information may lead to confusion of the reader and raises challenges for effective organization and aggregation of information for human consumption. The volume of news available on the WEB is accumulating at a constantly increasing speed [89]. The resulting news availability has created the need for automated discovery and extraction of relevant information without having to search for it manually.

To answer "*news information overload*" challenges, much research has been done in the field of information retrieval resulting in document clustering and summarization techniques. News clustering [24, 91, 92] allowed automatic grouping of highly-related news articles from a multitude of sources into coherent topics. While it does not reduce the overall amount of information, it removes the need for the reader to manually search for related news articles. Text summarization [12, 83, 94] allowed automatic extraction of the most important points of the original text producing a shortened version of it. Moreover, summarization can be applied on a collection of related articles producing a single coherent summary [44]. While summarization greatly reduces the amount of information for a single news cluster, there is still a difficulty for the reader to orient among all worldwide events. This difficulty increases with the reader's intention to refresh events of the previous days or weeks.

There is a need for more intelligent system which can understand events mentioned in the text, search for interesting events, extract valuable information about events, and present events in an easy observable repository (Figure 1-1). Such a repository can provide a comprehensive overview of the world situation. It is useful by itself, creating a timeline of distinct events out of hundreds of news articles repeatedly mentioning many events.

In this thesis, a way to automatically build such an event repository is proposed. The interest is in detecting all unique events and extracting a list of all textual references used to mention the same event. Unlike previous work in event extraction, it is not our interest to populate events in a fixed set of predefined templates [14, 33, 99] or in certain specific domain, such as detecting infectious disease outbreaks [81], violent [40] and natural disaster events [36], and biomedical events [6, 35, 46, 50, 88, 90]. Instead, the aim is to identify an unrestricted set of open-domain events [7] and all their possible textual references.

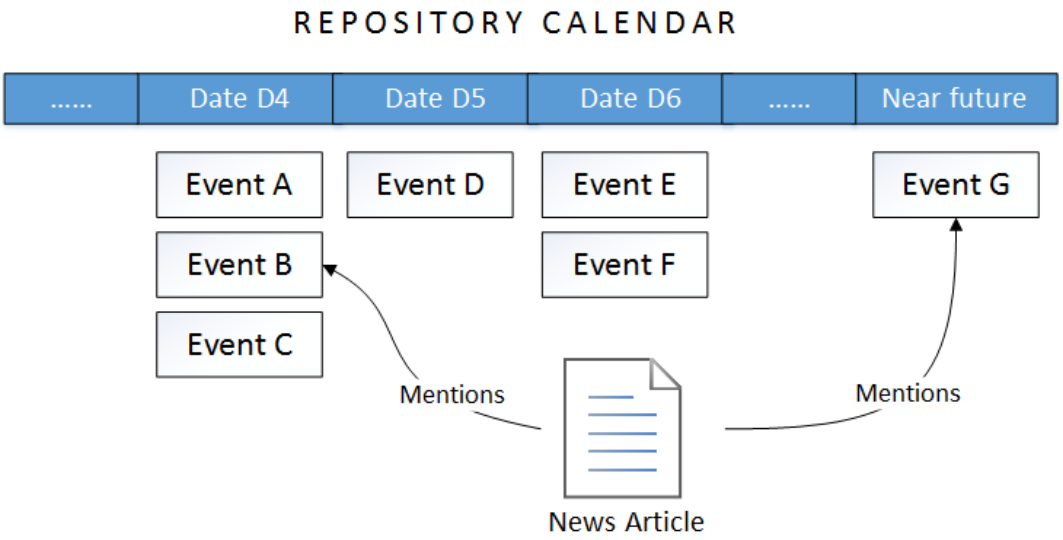

Figure 1-1. Schematic calendar view of the event repository

This is not a trivial task because events can be expressed in a very complex way. Full event description can be scattered over several sentences and articles. Additionally, descriptions of the same event can differ in specificity and granularity. For example, *shooting* and *several shots* can refer to the same actual event. The main challenges of the thesis are extracting events and resolving them to a unique event instance in the repository. Resolving process should allow soft matching based on shifts in granularity and abstraction.

1.2 Problem Statement

This section provides an overview of the event extraction process. Definition of an event used in this thesis is given and required event components are specified. Challenges of the algorithm and characteristics of the input data are described.

1.2.1 Definition of an Event

This subsection provides definitions of an event to help readers understand what an event instance is. In general, definition of an event varies across different specifications, annotation guidelines and dictionaries:

- *Oxford dictionary* defines an event as something that happens or takes place;
- The annotation guidelines of the *Automatic Content Extraction* (ACE) program define an event as a specific occurrence of something that happens, often a change of state, involving participants [59];

- *TimeML* specification defines an event as situation that happens or occurs. It can be expressed as punctual, durational or stative predicate describing states or circumstances in which something obtains or holds true [43];
- *Topic Detection and Tracking* (TDT) field defines an event as something that happens at a specific time and place [42].

There is a distinction between an event and topic, which is the broader concept of events. For instance, Yang et al. characterize "*the USAir 427 crash that occurred on 08 September 1994 in Pittsburg*" as an event, whereas "*airplane crash*" is a topic [41]. Basing on this definition, events that occurred at different locations or happened at different times are distinguished as different event instances.

In addition to *time* and *place* acting as distinctive components of an event, guidelines for *Event Coreference Bank Plus* (ECB+) corpus [4, 5] also distinguish event *action* and *participants*. Event action describes *what* happens or holds true, whereas event participants describe *who* or *what* are involved, undergo change or facilitate an event or a state. Event action and participants fit well with every of the above-mentioned definitions of an event.

In the context of this thesis, the TDT definition is used as the base definition of an event. However, an event is generalized to consist of four event components: time, place, action and participants (both human and non-human).

1.2.2 Time Component of an Event

Time component of an event is a required component in the context of this thesis. Events without a time component are not recognized and not extracted. Place, participants and action components are all utilized to enrich the representation of an event but are not strictly required for the events to have.

Time component represents the date when an event happened. Extracting this information is one of the fundamental tasks for building a repository of events. Time information can be expressed with many different forms of temporal expressions, such as "*June 11, 1989*", "*on Tuesday 18th*", "*today*", "*the second of December*", "*this morning*", "*two days ago*", "*last Wednesday*", "*last week Friday (14 October 2011)*", "*Christmas Eve*", "*tonight*", etc. Temporal expressions within the text are recognized by temporal tagger [8, 45] and resolved to an actual timestamp using article's publication date as the reference date. Logically, all temporal expressions within the news article are usually resolved by the reader considering exactly publication date as the reference point.

Correct resolution of event dates is an absolute factor determining the correctness of extracted event repository. One of the detected problems is handling of ambiguous temporal expressions and expressions that do not represent time information. Consider the sentence "Measure the intensity of your activity throughout *the day*". The phrase "*the day*" is incorrectly identified as a temporal expression being equivalent of "*today*". Another problem is incorrect resolution of temporal expressions by the temporal tagger. For example, having a detected weekday temporal expression "*on Sunday*", it can be resolved to the next Sunday whereas the correct date is the previous Sunday. On the other hand, the correct resolution of such temporal expressions as "*two weeks ago*", "*Christmas show*" or "*at 5 a.m.*" often requires understanding of the context. The described problems were solved by exploiting the tense of the verb and by checking the text of temporal expressions for ambiguity. Further discussion on resolving temporal expressions is provided in Subsection 3.2.2.

1.2.3 Extraction of Events

This subsection shortly describes the problem of extracting events after detecting temporal expressions in the sentence. Newswire text is often written in a complex and stylistically decorated way usually combining many pieces of information in a single sentence. As a result, there is overwhelming reliance on complex structures, such as in the following example: "*Two male gang members, whose names remain unreleased, were arrested last night in the case of the murder of Los Angeles County Sheriff's Deputy Juan Abel Escalante which took place outside his Cypress Park home this 2 August as he prepared to leave for work*". Such complex structures raises three major challenges: (1) identify multiple events within the same sentence, (2) choose respective event details and (3) construct a short phrase mentioning a particular event out of the whole text of a sentence.

To overcome these challenges, *OpenIE* methods [51, 70, 71] are used to extract a list of facts for each sentence. The fact consists of the predicate and a number of associated arguments. Each sentence is processed by the dependency parser and semantic role labeler in order to get syntactic and semantic representations. The resulting dependency tree has associated semantic roles and represents predicate-argument structure for each event mentioned within the text. Event detection is triggered by a predicate which can be verb [47], noun, or even adjective [1]. Labelled arguments encompass specific semantic roles they play with respect to the corresponding predicate [63]. Figure 1-2 illustrates the dependency tree with annotated semantic arguments for the example sentence "*Police confirmed Lo Presti had hanged himself on Tuesday night in Palermo prison*".

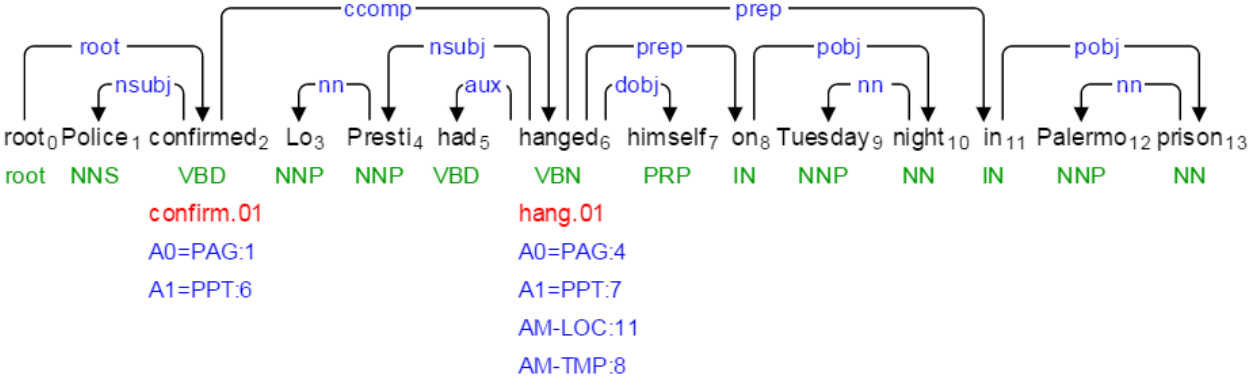

Figure 1-2. Dependency tree with semantic arguments

The dependency tree encompasses syntactic and semantic information of the sentence. There are two detected predicate-argument structures in the example above. The first verb predicate "*confirm*" has two semantic arguments *A0* and *A1*. *A0* generally represents a prototypical agent, whereas *A1* represents a prototypical theme [60] (e.g. *speaker* and *message* for the predicate *confirm*). The second verb predicate "*hang*" has four semantic arguments: two numbered arguments *A0* and *A1*, and two modifiers, *AM-TMP* and *AM-LOC*. *AM-TMP* and *AM-LOC* are used to annotate temporal and locational adjuncts, respectively. In addition to semantic roles, dependency tree represents typed grammatical relations between words (e.g. *Presti* is the nominal subject (*nsubj*) of the word *hang*) and each word is labelled with part-of-speech tags (e.g. *VBN* stands for past participle verb, *NNP* for singular proper noun).

Predicate-argument structures that do not contain temporal expressions are discarded in order to keep only those events that have an associated time component. The remaining dependency structures are analyzed to link recognized temporal expressions to appropriate predicates. For example, the temporal expression "*on Tuesday night*" refers only to the "*Lo Presti had hanged himself*" event. An algorithm traverses the dependency tree exploiting syntactic and semantic dependencies, extracts different event details and constructs an *event mention*. Each event mention represents a textual representation used to mention a particular event. From the example above, the outcome is a single event mention "*Lo Presti, hanged, himself, on Tuesday night* (e.g. resolved to *2015-03-15*) *in Palermo prison"*. On the next step, different textual representations used to mention the same actual event are grouped together within the news cluster to from *unique cluster events*. Finally, extracted cluster events are grouped within the whole repository to obtain *unique global events*. The process of grouping and different grouping methods are described in Chapter 4.

1.2.4 Retrieving News Clusters

The approach described in this thesis requires topically related clusters of news articles as the input. However, the scope of the thesis does not describe the process of grouping news articles into clusters. News clusters should be retrieved from existing clustering systems. The English language news clusters can be crawled from Google News[1]. Google News aggregates articles from a variety of news providers worldwide, and groups similar information content into the same cluster. News clusters are extended over time and can span several days. New articles are added to existing clusters if they follow the same event or content topic. The quality of produced clusters is observed to be very high. Each cluster is a collection of news articles typically speaking about a *single topic* that is reported by hundreds of different online sources. Articles in a cluster, therefore, describe similar information content and reference the same events using different words.

The same events can be reported multiple times, in different forms, with different level of details, both within the same article and within topically related articles. By utilizing news clusters as single units, the algorithm takes advantage of these alternate descriptions to improve global event extraction by eliminating inconsistencies of event representations. The typical size of a news cluster is about 150 articles, which are published over 1-2 days. Hot or breaking events can have large clusters containing more than a thousand of articles. Bigger news clusters potentially contain bigger varieties of textual representations of the same event. The required information for each news article utilized by the algorithm is:

- URL – URL of the article, which is used as a unique identifier;
- Publication date – the date of the publication of the article;
- Cluster ID – the ID of cluster this article belongs (assigned by Google News);
- Category – the news category of the article;

1.3 Thesis Goal

The goal of the thesis is to examine an unsupervised algorithm to build a global repository of events from exploiting topically related clusters of news articles. Given a news cluster as an input, the algorithm extracts *event mentions* on a sentence level from the text of news articles. Then event mentions are grouped into unique *cluster events* while eliminating different textual representations of the same event. Finally, *cluster events* extracted from all news clusters are grouped into unique *global events* within the repository.

[1] https://news.google.com/

Each extracted cluster and global event consists of a unique ID, resolved date when an event happened, list of textual representations mentioning this event, and location where the event took place. For readability purposes, a representative phrase of an event is also identified. In order to accomplish the thesis goal, the following implementation steps are defined:

1. Retrieve clusters of news articles by crawling Google News website;
2. Extract *event mentions* from news articles:
    a. Recognize temporal expressions within the text and resolve them to the standardized form using temporal tagger;
    b. Extract open-domain facts using *OpenIE* techniques;
3. Develop methods for grouping *event mentions* into *cluster events*;
4. Develop methods for grouping *cluster events* into *global events*;
5. Determine representative phrase for *cluster events* and *global events*;
6. Present extracted event repository using simple WEB interface;

Crawler of Google News website was provided by the DIMA group at the Technische Universität Berlin and was not implemented as the part of this thesis. Extraction of *event mentions* is based on the work "*Extracting a Repository of Events and Event References from News Clusters*" by Silvia Julinda [89] of the 2012-2014 IT4BI cohort. Her work provided the necessary proof of concept for *event mentions* extraction. However, many parts of the initially proposed approach were reworked and improved. Improved and changed parts are listed in Section 6.2 with clear indication of the improvement consequences.

1.4 Thesis Contributions

This thesis provides the following contributions:

- Novel approach to build a global repository of events from exploiting topically related clusters of news articles as single units. Experiments indicate the strong potential with 82.5% precision, 62.5% recall and 74.8% purity values;
- Improved extraction of *event mentions* on the sentence level using *OpenIE* techniques. Achieved mostly by text preprocessing, removing ambiguous temporal expressions, and resolving insufficient semantic roles;
- Solid evaluation of the approach using adapted *Even Coreference Bank Plus* (ECB+) corpus [4, 5] and manually annotated real-world corpus. Analysis of errors and identification of future directions for improvements;

- Presentation of the extracted event repository using calendar view. The created repository provides a comprehensive overview of the world events and can be used for the future event-linking information extraction effort.

1.5 Thesis Outline

This thesis consists of six chapters. Chapter 2 provides a theoretical background for event extraction and illustrates prerequisites for understanding the extraction algorithm. Dependency parsing, semantic role labelling, and previous work related to the thesis are discussed. Chapter 3 provides a detailed explanation of each step of the developed algorithm. Such concepts as recognition and resolution of temporal expressions, dependency-based semantic role labelling, extraction and subsequent grouping of *event mentions* are detailly illustrated. Different methods for grouping *event mentions* into *cluster events* and *cluster events* into *global events* are described in Chapter 4. The experimental results are presented and analyzed in Chapter 5. Used evaluation corpuses and metrics are described and formalized. Sources of errors, limitations and shortcomings of the developed method are illustrated. Finally, Chapter 6 gives a summary of the thesis and lists the directions for future work.

Chapter 2

Background

This chapter introduces theoretical foundations of different components used in this thesis and intended to illustrate the prerequisites needed to understand the proposed algorithm. The chapter begins with definition of information extraction (Section 2.1) and open-domain information extraction (Section 2.2). Section 2.3 describes different approaches for event extraction with a number of related projects compared to the approach in this thesis. Section 2.4 describes and compares two syntactic parsing methods: dependency parsing and constituency parsing. Semantic analysis and semantic role labelling are discussed in Section 2.5. Used NLP libraries and other tools are described in Section 2.6.

2.1 Information Extraction

Information extraction (IE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents [62]. In most of the cases this activity concerns processing human language texts by means of natural language processing. Information extraction identifies a predefined set of concepts in a specific domain, ignoring other irrelevant information, where a domain consists of a corpus of texts together with a clearly specified information need [95]. The presence of specific information need and target application domain are identifying factors.

IE systems extract clear, factual information out of unstructured text and put it in a semantically precise form that allows further processing. Even in a limited domain, IE is a non-trivial task due to the complexity and ambiguity of natural language. There are many ways of expressing the same fact, which can be distributed across multiple sentences, documents or even knowledge repositories [95]. Information extraction usually includes such tasks as:

- Named entity recognition;
- Coreference resolution;
- Relations extraction;
- Event extraction.

2.2 Open-Domain Information Extraction

Traditional information extraction assumes that the structures to be extracted (the types of named entities, the types of relations, etc.) are predefined. In some cases, however, there is a need to extract all potentially useful facts from a large and diverse corpus such as the WEB [71].

Open-domain information extraction (*OpenIE*) system does not require any human input, makes a single run through the data and generates a large set of relational tuples. As the goal of the thesis is to extract an unrestricted set of all possible events from online news streams, it falls to the field of open-domain information extraction and uses *OpenIE* techniques. *OpenIE* focus on domain independent extraction and rely on the redundancy of the text.

2.3 Event Extraction

Event extraction refers to the task of identifying events in unstructured text and deriving detailed and structured information about them, ideally identifying *who did what to whom, when, where, through what methods (instruments), and why* [39]. Event extraction is considered to be the hardest of the information extraction tasks.

Three are several approaches for event extraction. First, there are data-driven approaches, described in Subsection 2.3.1, which aim to convert data to knowledge through the usage of statistics and machine learning. Second, there are knowledge-driven approaches, discussed in Subsection 2.3.2, which extract knowledge through representation and exploitation of expert knowledge. The hybrid event extraction approaches, discussed in Subsection 2.3.3, combine knowledge and data-driven methods. Each of the event extraction approaches is supported by a short explanation and description of example systems. The main differences of the approach used in this thesis in comparison to example systems are indicated.

2.3.1 Data-Driven Approaches

Data-driven approaches rely solely on quantitative methods to discover statistical relations in natural language [29]. Statistical relations are facts that are supported by statistical evidence. These facts are words or phrases statistically associated with each other. A large set of annotated data is required to build and test statistical models that approximate linguistic phenomena. Data-driven approaches are not restricted to probability theory only. They can encompass all quantitative approaches to automated language processing [29]. Some examples of data-driven methods are Term Frequency-Inverse Document Frequency (TF-IDF) metric, N-grams, and clustering.

Several usages of data-driven event extraction can be found in the literature. Hardy at al. applied several well-known machine learning algorithms (e.g. multi-nominal logistic regression model, probabilistic Naïve Bayes classifier) for discovering events and their components [34]. They used a set of easy-to-collect features such as part-of-speech, sentence length, named entities, and patterns of syntactic chunks to train the classifier. Full annotated corpus consisted on 3996 instances of 11 event types (e.g. political, financial, criminal, law). The classifier was trained using different sizes of training corpus and the output was evaluated for correct recognition of different event details. Another example is automatic extraction and classification of events from emails [56]. The events are classified into different categories: official meetings, personal meetings, and others, using *TF-IDF* similarity measure.

The advantage of data-driven approaches is that no domain knowledge is required, neither rich linguistic resources. However, discovered statistical relations do not necessarily imply semantically valid relations [29]. Additionally, large amount of annotated data is required in order to get statistically significant results. This approach is not preferred for achieving the goal of the thesis as substantial amount of human effort is required to annotate the news corpus.

2.3.2 Knowledge-Driven Approaches

Knowledge-driven approaches consider meaning of text and use linguistic, lexicographic as well as human knowledge for event extraction [31]. This approach needs little training data but more domain and expertise knowledge. Knowledge is represented as patterns expressed with rules. Patterns are created by exploiting linguistic and human knowledge with regards to the domain and are usually extracted from the training data by human expert. The patterns that use lexical representations and syntactical information in combination with regular expressions are called lexico-syntactic patterns [65, 66], whereas the patterns which also make use of semantic information are called lexico-semantic patterns [49]. This subsection gives an overview of the three example knowledge-driven systems.

NEXUS

News cluster Event eXtraction Using language Structures (NEXUS) is an event extraction system utilized for populating violent incident knowledge bases [40]. It automatically extracts security-related facts from clusters of online news articles. For each cluster it tries to detect and extract only the main event by analyzing all articles of a cluster.

In an initial step, each article of the cluster is linguistically preprocessed in order to get more abstract representation. This encompasses sentence boundary detection, domain-specific

dictionary look-up (e.g., recognizing numbers, quantifiers, persons), identification of unnamed person groups (e.g. six civilians), simple chunking, labeling of action words and morphological analysis. In the subsequent steps, the pattern engine applies a set of hand-coded extraction rules on each article and core templates learned from annotated training data (3415 templates in total). The last step consists of cross-article cluster-level information fusion in order to produce fully-fledged event description.

Different from our approach, NEXUS system extracts only one event per news cluster. Date of this event is identified by the date of the cluster. However, news clusters tend to mention many different events on a variety of days. The goal of this thesis is to extract all existing events together with the multitude of corresponding event mentions.

Chinese News Fact Extractor

Chinese News Fact Extractor [99] is another example of a knowledge-driven event extraction system. The system is intended to recognize 33 ACE [59] event types from Chinese online news. Events and their components are extracted by combining rule-based method (verb-driven) and a supervised machine learning method. By utilizing an article's headline, the system identifies informative topic sentences and extracts main event only from these sentences. In their interpretation, an extracted event is a summarization of a news article. Our approach is similar to this system by limiting the scope of event extraction to specific sentences. This results in reduced computation overload. However, our approach is not restricted only to the main event of a news article and works on all sentences with non-ambiguous day-level temporal expressions.

Event Extraction using Semantic Parsing

Another example made by Exner et al. is system that automatically extracts unrestricted set of events from Wikipedia using semantic parsing [76, 86]. Semantic Role Labeler (SRL) is used to extract the predicate in the sentence together with its arguments, or roles, such as agent, theme, temporal and locational adjuncts. Subsequent steps use regular expressions to extract event components from extracted arguments. Extracted events without time, place or agent are discarded. Time component is identified using Named Entity tagger and time phrases that do not contain date expressions, such as "*three days ago*", are discarded. Furthermore, the system automatically links extracted event components to the external resources *DBpedia* database [15] and *GeoNames* web service. Our approach is similar to this system by doing semantic parsing. However, our definition of an event is less restrictive and only requires the time component. Additionally, it resolves detected temporal expressions to an actual timestamp.

Summary

Knowledge-driven approaches require less training data than data-driven approaches and the patterns are powerful to extract very specific information [29]. Drawback of this approach is that domain knowledge is required to define extraction patterns, which are usually hand-crafted and hand-tuned. Additionally, it is difficult to scale-up or reuse available patterns to cover more situations [58]. Majority of existing systems using knowledge-driven approach are domain dependent, such as biological domain [11, 35] and financial domain [26]. Also, a lot of effort is required to create accurate templates defining possible events for each target domain.

Our proposal for event extraction is similar to these systems by using syntactic and semantic information. However, proposed system does not use fixed set of event templates in order to identify an unrestricted set of events in an open domain, temporal expressions are resolved to an actual timestamp, and duplicated events are grouped into a single event instance.

2.3.3 Hybrid Approaches

Besides the advantages of both approaches, there is increasing number of researchers that combine two approaches to overcome their disadvantages and generate better results [29]. The combination of these two approaches is called hybrid approach. For example, data-driven methods can be utilized to overcome the lack of expert knowledge in knowledge-driven approaches [31, 78]. Another application is to reinforce statistical methods by combining with lexical knowledge [98]. Two recent systems that utilize hybrid approach for event extraction are *TwiCal* [7] and *NewsSpike* [20].

TwiCal

TwiCal [7] is the first open-domain event extraction and categorization system for Twitter. Events are extracted in 4-tuple representation which includes a named entity, event phrase, calendar date, and event type (Figure 2-1). This representation closely matches the way important events are typically mentioned within the text of twitter messages. Similar to our approach, the date when an event happened is resolved from temporal expressions within the text of a message using message publication date as a reference date. However, *TwiCal* utilizes a less strict definition of ambiguous temporal expressions [7].

TwiCal is a hybrid event extraction system due to the nature of its processing pipeline, namely because of such components as *event tagger* and *event classifier*. The task of *event tagger* is to determine event phrase, while *event classifier* assigns a particular event type.

Entity	Event Phrase	Date	Type
Steve Jobs	died	10/6/11	DEATH
iPhone	announcement	10/4/11	PRODUCTLAUNCH
GOP	debate	9/7/11	POLITICALEVENT
Amanda Knox	verdict	10/3/11	TRIAL

Figure 2-1. Examples of events extracted by TwiCal. Figure taken from [7]

Event tagger exploits initial expert knowledge to recognize event triggers. *Event tagger* is trained on a corpus of 1000 tweets (19,484 tokens) with annotated event phrases [7]. Each tweet is annotated using contextual features (e.g. part-of-speech tags, adjacent words), dictionary features (e.g. event terms from WordNet [30, 85], Brown Clusters), and orthographic features (e.g. prefixes, suffixes, punctuation).

The *event classifier* categorizes the events into types based on latent variable models. The latent variable models automatically discover the event types which reduces the amount of manual effort required to annotate a large corpus of tweets [7]. Human expert then manually inspects the discovered types and assigns them with labels (such as sports, politics, and product releases, etc.) and marks event words which assign the highest probability to that type.

Comparing to *TwiCal*, our proposed system is intended to work on complicated and details-rich news sentences, which are different from short and self-contained tweets. Therefore, in addition to part-of-speech tagging, there is also a need for syntactic and semantic parsing. Additionally, our system is intended to provide more event details, but without categorizing events into types.

NewsSpike

NewsSpike [20] is an open domain relation extraction system from news streams. Although the focus of *NewsSpike* is to generate semantically equivalent paraphrases for entity relations, the system uses relevant approach for event extraction.

NewsSpike applies open information extraction system called *Reverb* [9] to obtain a set of (a_1, r, a_2, t) tuples, where (a_1, a_2, t) suggests a real-world event and relation r is likely to describe that event, while t denotes the timestamp. However, this timestamp does not represent the date when an event happened, but instead denotes an article's publication date. Different to our approach, *NewsSpike* does not utilize temporal expressions within the text.

Afterwards, the system applies temporal heuristic rules [20] combined with probabilistic graphical model to generate semantically equivalent paraphrase clusters, like {*step down from, resign from, cut ties with*}. These paraphrase clusters than can be used to detect textual references likely to speak about the same event.

Summary

In hybrid event extraction systems, the training data is still needed due to the usage of data-driven methods, but the required amount is typically smaller than with purely data-driven systems [29]. Complexity of hybrid systems and the amount of required expertise increases due to the combination of multiple techniques. Hence, this approach is more suitable for researches and recommended only for advanced users [29].

2.4 Syntactic Analysis

The purpose of syntactic analysis is to determine the structure of the input sentence. Such formalizations are aimed at making computers understand relationships between words (and indirectly between corresponding people, things, and actions) [82].

This section gives an overview of two syntactic parsing approaches. Subsection 2.4.1 describes dependency parsing, which is used in this thesis. An alternative method of constituency parsing is shortly described in Subsection 2.4.2. Finally, the Subsection 2.4.3 provides comparison of parsing approaches and lists reasons for choosing the dependency parsing as an approach for syntactic analysis.

2.4.1 Dependency Parsing

Dependency parsing is an approach to analyze the syntactic structure of a sentence inspired by the theoretical linguistic tradition of dependency grammar [87]. This grammatical tradition is said to culminate with the seminal work of a French linguist Lucien Tesniere in 1959. The idea is expressed in the following way [61]:

"The sentence is an *organized whole*, the constituent elements of which are *words*. [1.2] Every word that belongs to a sentence ceases by itself to be isolated as in the dictionary. Between the word and its neighbors, the mind perceives *connections*, the totality of which forms the structure of the sentence. [1.3] The structural connections establish *dependency* relations between the words. Each connection in principle unites a *superior* term and an *inferior* term. [2.1] The superior term receives the name *governor*. The inferior term receives the name *subordinate*. Thus, in the sentence *Alfred parle* [...], *parle* is the governor and *Alfred* the subordinate. [2.2]"

As indicated by the statements above, the basic idea of dependency grammar is that the syntactic structure of a sentence essentially consists of words that are linked by binary asymmetrical relations [87] (Figure 2-2). This binary asymmetrical relation between the pair of

words is called *dependency relation*. A dependency relation holds between a syntactically *subordinate* word and a *governor* word. More common names in the literature are *head* for the *governor*, and *dependent* for the *subordinate* [68]. The *head* plays the larger role in determining the behavior of the pair. In general, the *dependent* is the modifier, object or complement.

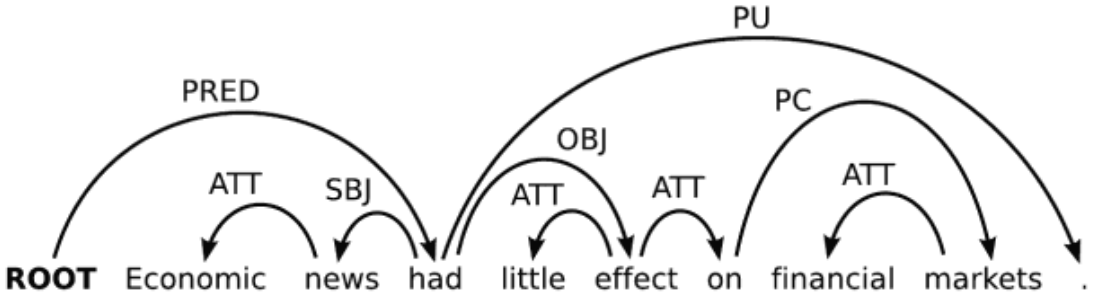

Figure 2-2. Dependency structure of a sentence. Figure taken from [87]

The dependency structure is a tree (directed acyclic graph) with the main verb representing a root of a sentence [68]. Dependency relations are represented by directed arcs pointing from the *head* to the *dependent* and are assigned with a specific *dependency type*. For example, the verb *had* is the head of the dependent noun *news* with the dependency type *subject* (SBJ). By contrast, the noun *effect* is also dependent of the verb *had* with the dependency type *object* (OBJ). Every real word of the sentence should have a syntactic *head*, but *dependent* words may be optional [87]. In this case, the verb *had* does not satisfy this formal definition because it lacks a syntactic head. Thus, an artificial word *ROOT* is inserted to the beginning of each sentence and the verb *had* is labeled as the dependent of *ROOT* [87].

While most head-dependent structures have a straightforward analysis in dependency grammar, there are also constructions that have a relatively unclear status. This group includes complex verb groups, subordinate clauses, auxiliary verbs and prepositions [53]. For these constructions, there is no general consensus regarding what should be considered as the dependent and as a head.

Another kind of problematic construction for dependency grammar is *coordination*. Consider an example sentence "*They operate ships and banks*" [87]. Although, the phrase "*ships and banks*" is clearly identified as a direct object of the verb *operate*, it is not clear how to determine relationship between nouns as both are equally good candidates for being heads. Figure 2-3 shows examples of possible dependency structures for coordination.

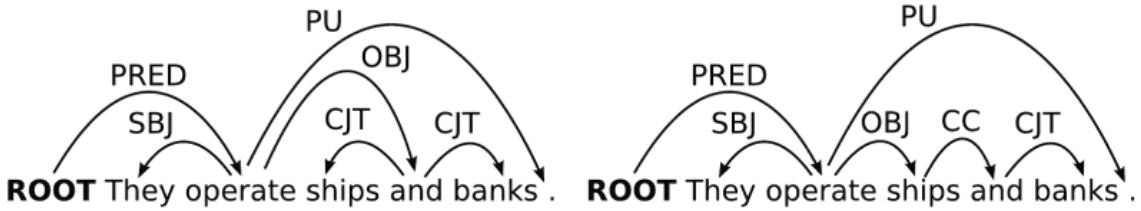

Figure 2-3. Possible dependency structures for coordination. Figure taken from [87]

An example on the left represents the coordination conjunction as a head of the coordination structure. Both nouns "*ships*" and "*banks*" are dependent of the word "*and*" with dependency type CJT (special type that does not describe the grammatical function of the conjuncts). The word "*and*" acts as an *object* for the verb "*operate*" populating its meaning to the dependent nouns.

An example on the right represents the coordination conjunction as dependent of the first noun "*ships*", which acts as an object for the verb *operate*. The word "*and*" is labeled as the dependent of "*ships*" with type CC (coordination) denoting its grammatical function. This example may look closer to the way humans perceive information while reading the text. The decision to choose the solution depends on the specific grammar or corpus annotation guidelines as dependency relations may vary from one framework to another [87].

The task of a *dependency parsing* is to take a given input sentence and impose on it the appropriate set of *dependency relations* [68]. Kübler et al. defines dependency relations as dependency graphs and formulates the parsing problem as mapping words in a sentence S, to its dependency graph G as follows [87]:

Let $S = w_0, w_1, \dots, w_n$ be a sentence represented by a sequence of tokens. Let $R = \{r_1, \dots, r_m\}$ be a finite set of possible *dependency relation types* that can hold between any two words in a sentence. A *dependency graph* $G = (V, A)$ is a labeled directed graph, consisting of nodes V, and arcs A, such that for sentence S and label set R the following holds:

1. $V \subseteq \{ w_0, w_1, \dots, w_n \}$
2. $A \subseteq V \times R \times V$
3. if $(w_i, r, w_j) \in A$ then $(w_i, r', w_j) \notin A$ for all $r' \neq r$

A dependency graph is thus a set of labeled dependency relations between the words of a sentence. To illustrate above definition, consider the dependency graph in Figure 2-2, which is represented by:

1. $G = (V, A)$
2. $V = V_s = \{ROOT, Economic, news, had, little, effect, on, financtial, markets, .\}$
3. $A = \{(ROOT, PRED, had), (had, SBJ, news), (hab, OBJ, effect), (had, PU, .),$
   $(news, ATT, Economic), (effect, ATT, little), (effect, ATT, on),$
   $(on, PC, markets), (markets, ATT, financial)\}$

The nature of dependency relations (w_i, r, w_j) is not always straightforward to define and differs across linguistic theories [87].

2.4.2 Constituency Parsing

Constituency parsing breaks up the sentence into constituents (phrases), which are then recursively broken into smaller constituents. This representation is arguably the most widely used type of syntactic representation in both theoretical and computational linguistics [87]. While the dependency parsing represents head-dependent relations between words, classified by *functional* categories such as subject (SBJ) and object (OBJ), the constituent parsing represents the grouping of words into phrases, classified by *structural* categories such as *noun phrase* (NP) and *verb phrase* (VP) (Figure 2-4).

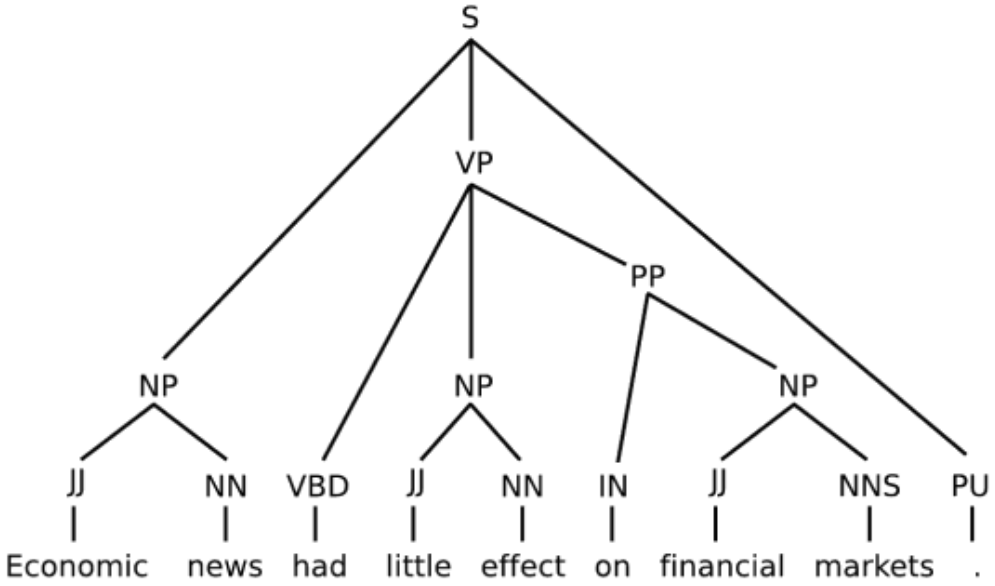

Figure 2-4. An example of constituency parsing. Figure taken from [87]

The grammaticality of a sentence is represented by a *phrasal* tree structure with each node representing a *phrase element* and each terminal node representing a word. The root of the tree is the entire sentence (denoted with S), and also called *clause*. The sentence is understood in term of a division of a clause into *noun phrase* (subject) and *verb phrase* (predicate plus object). The phrases by themselves do not represent independent sentences. In the given example, a sentence consists of a noun phrase (NP), verb phrase (VP) and punctuation (PU).

A grammar of a language defines set of rules to form grammatically correct phrases from real words depending on their part-of-speech tags [75]. For example, noun phrase, among other cases, can be formed from adjective (JJ) and noun (NN) allowing "*Economic*" and "*news*" to be grouped together. The phrase can include more than two other elements and can mix phrases with terminal nodes, for example as verb phrase in Figure 2-4. The part-of-speech (POS) tags used in this thesis are from Penn Treebank POS target [72]. Description of POS tags used in examples is given in Appendix A.3.

2.4.3 Comparison of Parsing Approaches

The major difference between syntactic parsing approaches is that dependency parsing creates one node per word, instead of hierarchy of mid-level phrases as in constituency parsing. Also the encoded information is different: dependency parsing operates with *functional* categories, while constituent parsing operates with *structural* categories. Dependency parsing for syntactic parsing provides such advantages as:

- Dependency relations are closer to the semantic relationships needed for the next stage of interpretation [52, 68];
- Dependency parsing is much faster than constituent parsing which makes it more suitable for large-scale data processing [51]. Faster processing is explained by the more constrained nature of the dependency parsing [87] and reduced search space because it links together only nodes that already exist without creating an unknown number of mid-level hierarchies;
- Dependency grammar is better suited for languages with free or flexible word order, making it possible to analyze typologically diverse languages within a common framework [53]. However, this does not mean that constituency parsing is more suited for English language because dependency parsing has been successfully applied to English language texts in [28, 51, 69];
- Dependency parsing provides transparent encoding of predicate-argument structure of a sentence [53] (Figure 2-5). Even isolated fragments of parsing output can be directly interpreted;

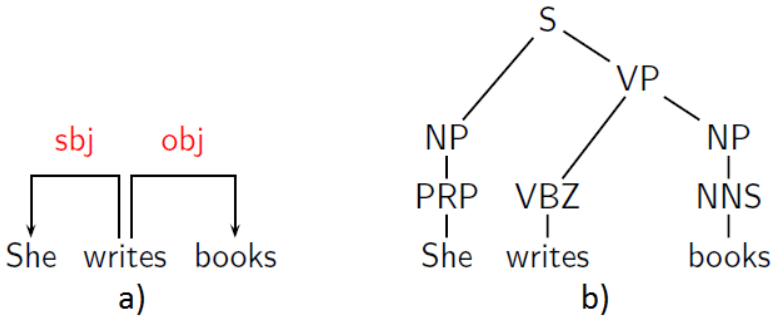

Figure 2-5. a) Dependency parsing directly encodes subject and object of a sentence b) Constituency parsing encodes sentence into noun phrase and verb phrase

- The dependency representation is more compact and simpler than constituency structure and, thus, easier to understand. Dependency graphs consist of nodes and a set of directed arcs representing labeled dependency relations. On the other hand, a sentence represented with constituency parsing has a more complicated tree structure containing phrasal nodes as the grammatical elements;

- Dependency parsing is applicable to many languages with minor modifications [51] and has led to the development of accurate syntactic parsers [53];

The disadvantage of dependency parsing is less expressive representation with respect to certain aspects of syntactic structure [68]. For example, it is impossible to distinguish in a pure dependency representation between an element modifying the head of a phrase and the same element modifying the entire phrase. It is also possible to convert dependency graph to phrasal structure by letting each head word represent a phrase consisting of all its underlying nodes [87]. However, it contains less explicit information about the constituents of the sentence as compared to the phrases in the constituent parsing.

As the goal of the thesis is to extract events from sentences, it was decided that formalism representing functional relations between words is more useful for extraction of events and different event details. Additionally, dependency parsing provides direct encoding of predicate-argument structure which is relevant for semantic interpretation. Dependency parsing is robust and performs well even on long sentences, which is often the case when working with newswire resources. For these reasons, the dependency parsing is used for syntactic parsing in this thesis. More specific details on the used parsing library are given in Subsection 2.6.1.

2.5 Semantic Role Labelling

The previous section described the parsing of the sentence into its corresponding syntactic representation. However, it does not represent the semantic relations between text constituents. In order to fully understand context of the sentence and answer "*who did what to whom where and when*" it is necessary to perform sentence-level semantic analysis. Having the possibility to answer these questions would allow detection and extraction of events being described within the sentence.

Semantic Role Labeling is the task of discovering predicate-argument structure of each predicate in a given input sentence and labelling the arguments with semantic roles they play with respect to the predicate [63]. In theory, a predicate can be any lexical category (e.g., verb, noun) that forms its own predicate argument structure [51]. To give an example, a sentence "*Peter sold his lunch to Betty in school yesterday*" results in identification of the predicate *sell* involving *Peter* as the seller (Agent), *Betty* as the buyer (Recipient), and *lunch* as the thing sold (Theme). In addition to that, the *school* is identified as the location where the selling event took place, and *yesterday* as the time when the selling event happened. Such relationships (e.g. agent, recipient) are called as *semantic roles*.

The number and type of semantic roles associated with a predicate are determined by the semantics of the predicate itself. A set of widely recognized semantic roles is described in [6]. Semantic roles are different from the grammatical relations of syntactic parsing described in Section 2.4. Grammatical relations (e.g. subject, object, indirect object) are morphosyntactic describing relations between words in a sentence. On the other hand, semantic roles (e.g. agent, theme, recipient) are conceptual notions [96] describing roles the participants play in events and situations. To illustrate the differences, consider the following examples [38] in Table 2-1.

Table 2-1. Difference between semantic roles and grammatical relations

Sentence	Phrase	Grammatical Relation	Semantic Role
Bob opened the door with a key	Bob	Subject	Agent
The key opened the door	The key	Subject	Instrument
The door opened	The door	Subject	Patient

There is many-to-few mapping between semantic roles and grammatical relations [10]. Consider two example sentences: "*Anna killed the king*" and "*Anna was killed by the king*". In both examples *Anna* is the subject of the sentence, but in the first example *Anna* acts as an Agent, while in the second as a Patient of the predicate *kill.* In general, the number of possible semantic distinctions vastly outnumbers those actually lexicalized in a language [73].

Semantic role labels are provided by several lexical resources such as *FrameNet* [19], *VerbNet* [57], and *PropBank* [18]. These lexical resources were created trying to solve disadvantages of previous semantic corpuses but can be seen as complementing each other. The main difference is the granularity of semantic role labels [63]. Unlike *FrameNet* and *VerbNet* that provide situation-specific semantic roles, *PropBank* uses more coarse-grained semantic meta-roles numbered sequentially from *A0* (*ARG0*) to *A5* (*ARG5*). It is also possible to map to more fine-grained labels, such as agent, theme, recipient, through predefined mappings [64]. It is worth mentioning that the same semantic role (e.g. *A0*) can only be interpreted in a verb-specific manner. However, *A0* is generally considered to be a prototypical Agent, while *A1* is a prototypical Theme/Patient across verbs [60].

The definition of an event used in this thesis (Subsection 1.2.1) aligns very well with the description of semantic roles. For instance, event action is represented by the predicate while event participants can be obtained using the agent, patient and theme. The parts of the sentence describing the time and place of an event can be obtained using the temporal and locational semantic arguments. The fact that semantic role labelling is performed for each predicate within the sentence allows recognition of several events within the same text. The description of the semantic role labels used in this thesis is given in Appendix A.1.

2.6 NLP tools

This section provides an overview of the used NLP libraries and other tools. *ClearNLP* library which is used for dependency parsing and semantic role labelling is described in Subsection 2.6.1. *WordNet* database as the source of synonyms is described in Subsection 2.6.2.

2.6.1 ClearNLP

ClearNLP is an open-source library[2] used in this thesis for dependency parsing and semantic role labelling. *ClearNLP* provides fast and robust modular NLP components that have been proven to provide higher accuracies and parsing speed with respect to other state-of-the-art systems [51]. Better performance, higher accuracies and ease-of-use were the main reasons for choosing *ClearNLP* for syntactic and semantic parsing.

ClearNLP dependency convention uses *Stanford* dependency approach [23] as the core structure, but also integrates *CoNLL* dependency approach [48] to improve some limitations, such as add long-distance dependencies, enrich object predicates and to minimize unclassified dependencies [51]. The *Stanford* dependency approach is chosen as the core structure because it gives more fine-grained dependency labels and is more widely used than *CoNLL* dependency approach. Additionally, *ClearNLP* does not enforce projectivity. Although, projectivity provides some advantages, it is dropped from the properties of a well-formed dependency graph in order to represent correct dependency relations. Even in rigid word order languages such as English, non-projective dependencies are necessary to represent long-distance dependencies [51].

Rues and heuristics for finding the dependency heads were completely reanalyzed to handle relations not included in the previous dependency approaches [51]. This resulted in a newly created list of dependency labels called *CLEAR* dependency labels. *CLEAR* dependency labels are mostly inspired by the *Stanford* dependency approach, partially borrowed from the *CoNLL* dependency approach and extended with several newly introduced labels.

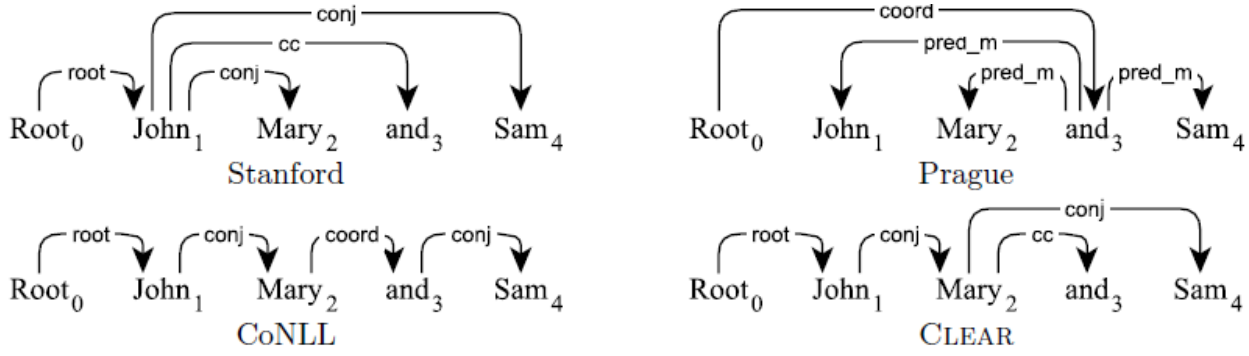

Figure 2-6. Different ways of representing coordination. Figure taken from [51]

[2] https://github.com/clir/clearnlp

An example of handling coordination by different dependency approaches is represented in Figure 2-6. The *Stanford* dependency approach makes the leftmost conjunct the head of all other conjuncts and conjunctions. In contrast, the *Prague* dependency approach assigns the head to the rightmost conjunction [67]. The *CoNLL* dependency approach assigns the head according to the order of words, making each preceding conjunct or conjunction the head of its following conjunct or conjunction. *CLEAR* approach is similar to the *CoNLL*, but conjunctions do not become the heads of conjuncts. This way, conjuncts are always dependents of their preceding conjuncts whether or not conjunctions exist in between.

For the semantic role labelling, *ClearNLP* uses semantic roles from the *PropBank* [3]. Semantic roles are added to the dependency trees produced by the dependency parser as the first step. Dependency-based semantic role labelling is faster and more straightforward comparing to the constituent-based semantic role labelling [51]. However, unlike constituent-based SRL that maps semantic roles to entire phrases, dependency-based SRL maps semantic roles to head-tokens only (Figure 2-7).

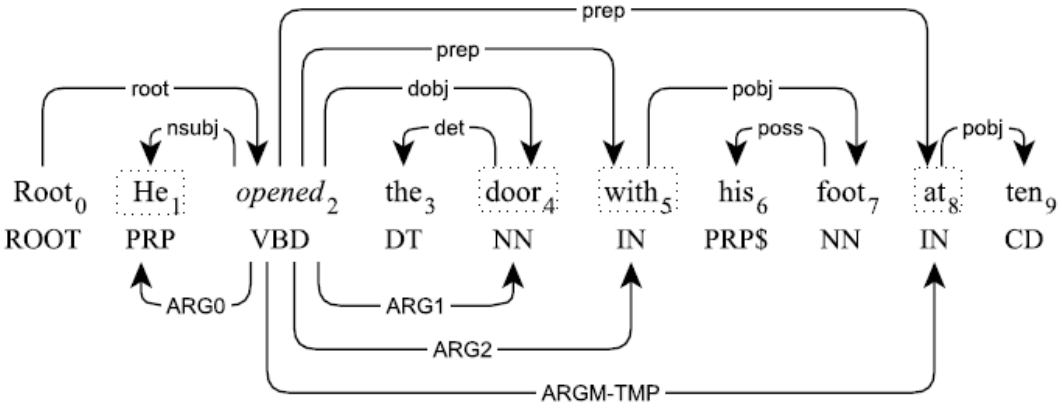

Figure 2-7. Dependency tree annotated with semantic arguments. Top arcs show syntactic dependencies. Bottom arcs show semantic dependencies. Figure taken from [51]

In the given example sentence "*He opened the door with his foot at ten*", there is one predicate *open* with four semantic arguments: three numbered arguments *ARG0, ARG1, ARG2* and one temporal argument *ARGM-TMP*. In the given context, the argument *ARG0* and *ARG1* represent the *opener* and *thing opening*, respectively, while *ARG2* represents an *instrument*. The time when the thing opened is represented by *ARGM-TMP* argument.

Semantic labels are assigned only to the head-tokens of dependency relations (heads are indicated by the dotted boxes). The subtrees of the head-tokens are then considered as semantic arguments. For instance, *the door* is a semantic argument of the predicate *open* with semantic role *ARG1*. However, the semantic role is assigned only to the head word *door*. This leads to a concern about getting the actual semantic phrase, but it was shown that it is possible to recover the original chunks from the head-tokens with minimal loss [52].

The semantic phrase can be constructed by traversing the syntactic subtree. This actually provides flexibility for extracting the representations of event participants as unnecessary details can be omitted. For example, to get the *instrument* phrase "*with his foot*" it is necessary to get the head-token of *ARG3* (*with*) and traverse the head-dependencies while correctly ordering dependent nodes.

2.6.2 WordNet

WordNet [30] is an online lexical database providing a large repository of English lexical items. It was designed to establish the connections between four major open-class syntactic categories: nouns, verbs, adjectives and adverbs. The smallest unit of information in a *WordNet* is *synset*, which represents a specific meaning of a word. It includes the word, its explanation, usage examples and list of synonyms.

The specific meaning of one word under one syntactic category is called a *sense*. The word has as many *senses* as it has different meanings or interpretations. Each distinct *sense* of a word belongs to a different *synset*. *Synsets* represent groups of words with synonyms meaning within the same lexical category (e.g. nouns). However, not all *synset* members are interchangeable in all contexts [16]. Each *synset* has a gloss that defines the concept it represents and usually includes few sentences illustrating the usage. For example, the words *car*, *auto*, *automobile*, *machine* and *motorcar* constitute a single *synset* that has the following gloss: "*motor vehicle with four wheels; usually propelled by an internal combustion engine"*.

Synsets are interlinked by means of conceptual-semantic and lexical relations. The result is a network of meaningfully related words and concepts. Some examples of relations are antonyms (*light-dark*), governor-subordinate (*furniture-chair*), part-whole (*finger-hand*), backward entailment (*divorce-marry*), manner-specificity (*whisper-talk*), presupposition (*buy-pay*), morpho-syntactic relations, etc.

WordNet is a lexical resource widely used for a range of natural language processing tasks [16]. Nevertheless, a much larger variety of semantic relations can be defined between words and word senses than are currently incorporated into *WordNet*. For instance, thematic and semantic roles of nouns functioning as arguments of specific verbs are not encoded [27], but could be useful for training semantic labelling. In the context of this thesis, *WordNet* is used as the source of synonyms to detect different representation of the same event.

Summary

This chapter has discussed and compared related work to the approach for event extraction used in this thesis. Necessary theoretical prerequisites for different processing components in this thesis have been introduced. Dependency parsing was chosen as the method for syntactic parsing. The main reasons in favor of dependency parsing are that it is faster than constituency parsing, output is more suitable for semantic parsing, and encoded functional dependencies are more useful for the extraction of event details. The benefits of semantic parsing for extracting different event components were discussed. Finally, the chosen *ClearNLP* library for dependency parsing and semantic role labelling was described.

Chapter 3

Extraction of Global Repository of Events

This chapter describes the algorithm used to build a global repository of events. The single diagram showing all details can be rather confusing, that is why the algorithm will be presented step-by-step. Each step is illustrated by the diagram and supported by the detailed explanation of the process. However, before going to the details, it can be advantageous firstly to look at the high-level view of the extraction process (Figure 3-1) to understand the overall information flow.

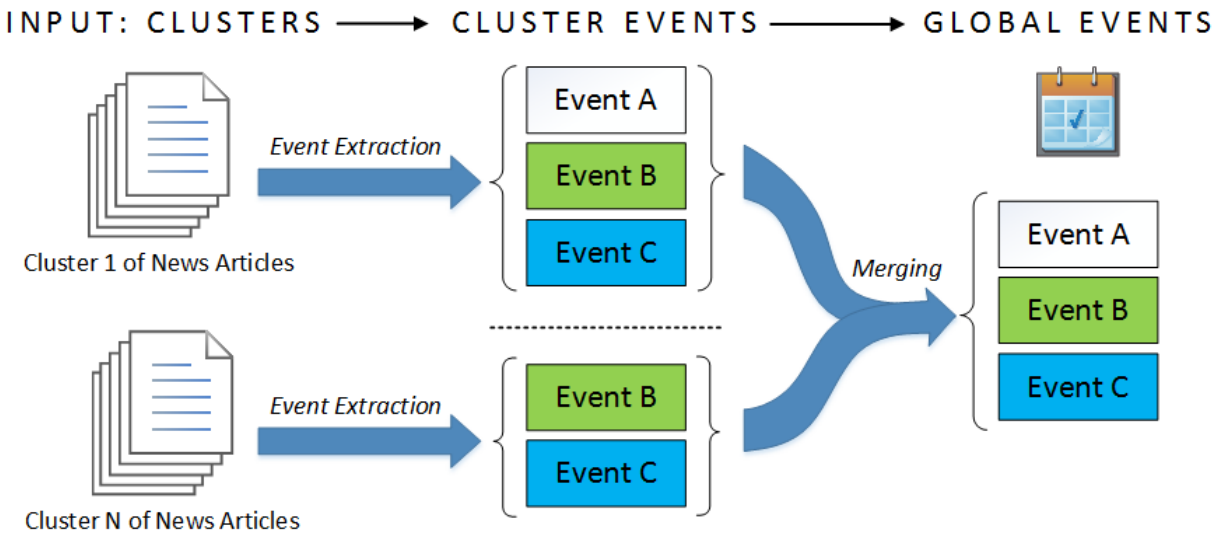

Figure 3-1. High-level view of global events extraction

The algorithm takes as input topically related clusters of news articles. The first major step is to extract unique events for each news cluster individually. In the context of this thesis, such events are called *cluster events*. The second major step of the algorithm is to extract unique events within the whole repository. Such events are called *global events*. From the preliminary investigation, the same event can be repeatedly mentioned across many news clusters. Therefore, the principal aim of the second extraction is to merge these duplicated *cluster events* into unique *global events*. Global events are stored in the database and then can be displayed to the user using the calendar view to provide a comprehensive overview of the world situation. Figure 3-2 provides a high-level view on the process of cluster events extraction.

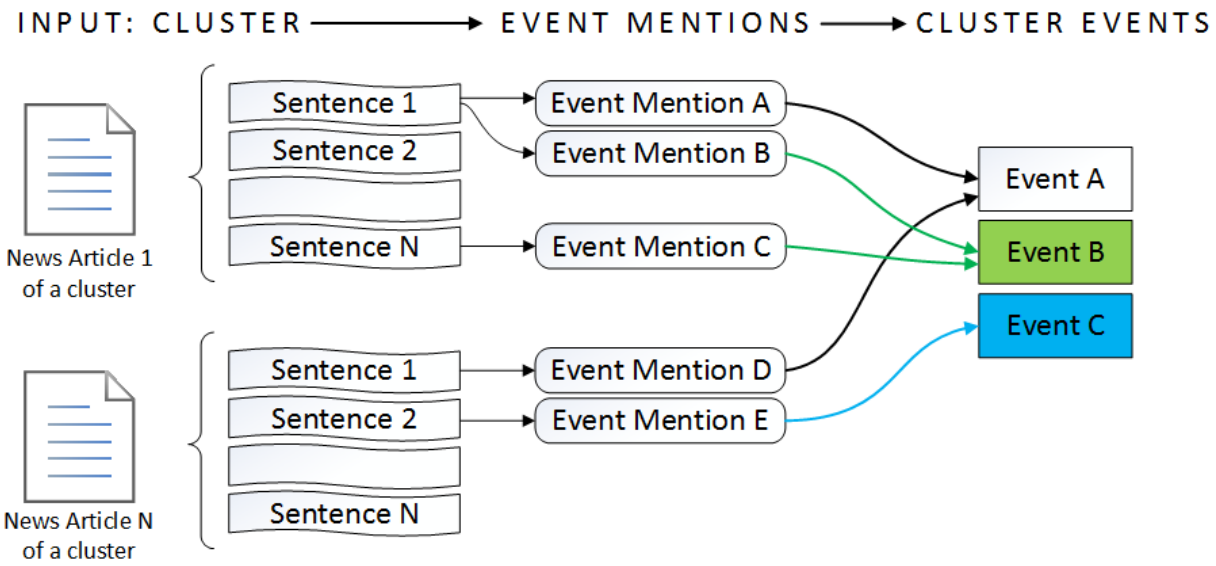

Figure 3-2. High-level view of cluster events extraction

Extraction of *cluster events* takes as input individual news clusters. *Cluster events* are created by grouping together highly similar *event mentions* that refer to the same point in time. *Event mentions* are defined as textual references used to mention a particular event. *Event mentions* are extracted on the sentence level from individual news articles. In general, each sentence can describe zero or more events, either different or identical. Consequently, each event is likely to be referred by many *event mentions*, both within the same article and across other articles of the same cluster as well as another cluster.

The whole algorithm is divided into five principal steps: (1) Retrieval of input clusters, (2) Extraction of cluster event mentions, (3) Extraction of cluster events, (4) Extraction of global events, (5) Determination of the event representative. Some of these principal steps are further subdivided into logical substeps. Individual steps are detailly described in following subsections.

3.1 Retrieval of Input Clusters

As it was mentioned in the previous section, the algorithm takes as input topically related clusters of news articles. The characteristics of the input clusters and the required information were described in Subsection 1.2.4. The process of building global repository of events starts with downloading input clusters from a dedicated news crawler (Figure 3-3).

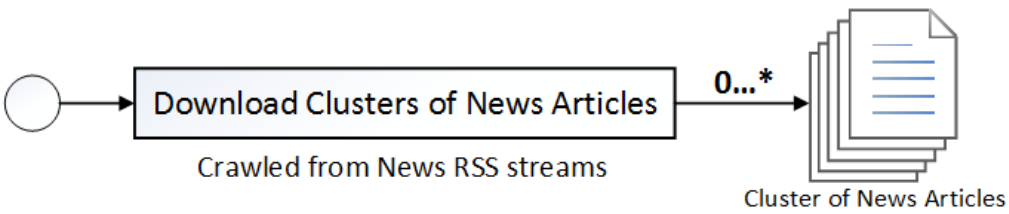

Figure 3-3. Retrieval of news clusters

The developed algorithm also supports retrieval of individual news articles and new clusters from locally stored TSV or JSON files. The algorithm downloads all the available news clusters and only then starts the extraction process. If the news cluster was already processed previously but was extended with new articles then only new articles are processed.

3.2 Extraction of Cluster Event Mentions

The process of *cluster event mentions* extraction is rather complicated and will be presented step-by-step. The input to the process is a single cluster of news articles. The output is a collection of all *event mentions* within the cluster (Figure 3-4). The output is created by combining all *event mentions* extracted from all individual news articles of the cluster.

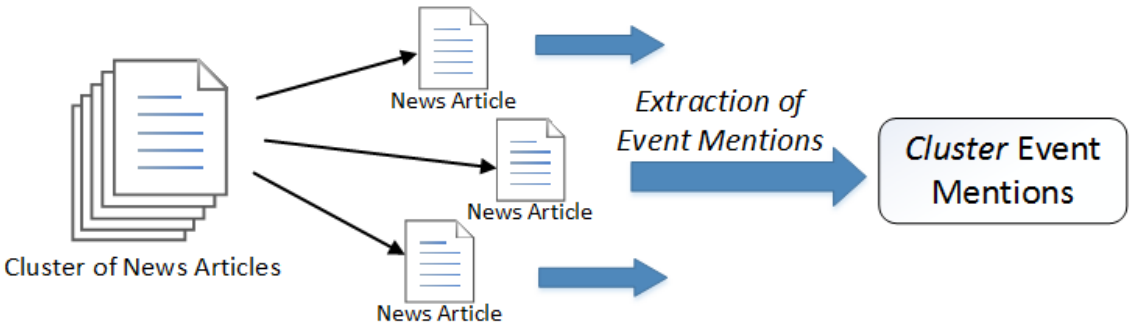

Figure 3-4. High-level view of cluster event mentions extraction

News articles are processed one-by-one in the order of articles' publication date and time. The process of event mentions extraction from individual news articles is further subdivided into the following substeps: (1) Sentence segmentation, (2) Temporal tagging, (3) Dependency-based semantic role labelling, and (4) Construction of event mentions.

During the first step, the text of an article is split into separate sentences. After that, each sentence is passed to the temporal tagger to detect temporal expressions. All sentences that contain temporal expressions are passed to the dependency parser and semantic role labeler, while others are skipped. The obtained dependency trees and semantic roles are used by the algorithm to extract *event mentions* within the sentences. In subsequent subsections each of these steps is described in more details.

3.2.1 Sentence Segmentation

Event mentions are extracted on the sentence level. Therefore, it is required to segmentize the text of news articles into separate sentences. Segmentation of the text is done by the *Stanford CoreNLP*[3] text segmenter, which detects sentence boundaries utilizing punctuation splitting and

[3] http://nlp.stanford.edu/software/corenlp.shtml

separation of affixes. However, it was noticed that text splitting into the sentences is not always correct due to the absence of, excess of or unusual punctuation, presence of links and even abbreviated weekdays (e.g. "*on Mon.*"). Additionally, the text of the article may contain links, advertisements, captions of photos, publication or update statements (e.g. "*the article was first published on March 13, 2013, 2:29 PM*") that do not provide any information gain, rather add noise to the text and errors to the extraction process.

Erroneous sentence splitting results in erroneous events extraction. Therefore, the text of the article is preprocessed before splitting it into the sentences (Figure 3-5). Text preprocessing includes text cleaning and normalization of dates.

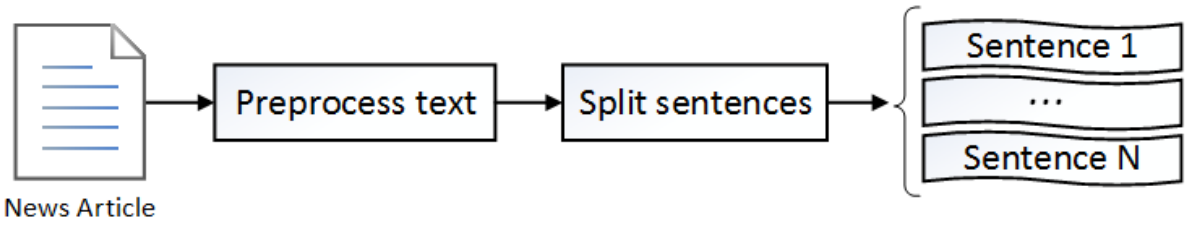

Figure 3-5. Segmentation of news article into sentences

Text Cleaning

The process of text cleaning removes unwanted information embedded inside the text of the articles and tries to resolve unusual punctuation. Text cleaning includes removal of:

- URLs;
- Unusual text punctuation;
- Metadata sentences;

Presence of URLs inside the text can cause wrong segmentation of text into sentences. For example, having a text "*Purchase was announced on www.example.com yesterday*", text segmenter can split the text into two sentences: (1) "*Purchase was announced on www.example.*" and (2) "*com yesterday*". Such splitting can cause unpredictable loss of information about events in the text. The solution is to detect URLs in the text and either completely remove them or substitute dots in the links with special keyword, e.g. "*www[dot]example[dot]com*".

Misleading or unusual punctuation can also cause incorrect segmentation of sentences. Some examples of such punctuations are dashes used instead of commas and doubled punctuation symbols (e.g. "--", ",,"). The presence of such punctuation causes even bigger obstacles for syntactic and semantic analysis. For example, consider the sentence "*A gang of four thieves -- two of them disguised as women -- on Thursday stole nearly all the jewels on display at the Harry Winston boutique*". The presence of unusual punctuation ("--") results in impossibility for syntactic analyzer to determine the structure of the sentence returning empty results.

Metadata sentences of the text are sentences that do not serve as the main content of an article. Therefore, such sentences should not be used for event extraction. It includes:

- Publication statements (e.g. "*Story first published on: 10 September 2014 20:12, IST*");
- Update statements (e.g. "*Last updated 05:00, 11 September 2014*");
- Social network sharing (e.g. "*follow us on Twitter and on Facebook*");
- Captions of photos (e.g. "… *Photo: Monica Davey, European Pressphoto Agency*");

Metadata sentences are not usually surrounded by punctuations. While splitting the text into sentences, metadata sentences stick to regular sentences, adding non-related information to the potential events described within the text. Metadata sentences are detected and removed by recognizing typical phrases in the beginning, end or within the text of an article.

Dates Normalization

Preprocessing of the text also includes normalization of dates. Normalization of dates is necessary to improve recognition of temporal expressions by temporal tagger (Subsection 3.2.2) and, especially, syntactic analyzer (Subsection 3.2.3). Dates are normalized to the form which is likely to be detected by both of them. Otherwise, not all parts of a temporal expression are recognized as a single phrase or temporal expression is assigned with wrong semantic role.

Dates normalization includes:

1. Month mentions are normalized to the full month names (e.g. "*Sep*", "*Sep.*" and "*Sept*" are normalized to "*September*");
2. Weekday mentions are normalized to the full weekday names (e.g. "*Wed*" and "*Wed.*" are normalized to "*Wednesday*");
3. Mentions of absolute dates within the text are normalized to the "*dd MM yyyy*" format. Day suffixes ("*st*", "*nd*", "*rd*", "*th*") are removed. Because of this step, expression "*on January 15th*" is normalized to "*on 15 January*"). This normalization is necessary to improve dates detection. Otherwise, having the text "*on January 15th*", temporal tagger can detect only "*on January*" that will be resolved to the month granularity;
4. Brackets around the dates are substituted with commas (e.g. "*yesterday (July 12)*" is changed to "*yesterday, July 12,*"). This is necessary because temporal tagger recognizes two separate temporal expressions within the text: (1) "*yesterday*" and (2) "*July 12*", which can latter result in duplicated event mentions. Additionally, syntactic analyzer does not add text in braces to the dependency trees;

5. Removal of commas before the year and after the weekdays in full date mentions (e.g. "*Meeting on Tuesday, 10 September, 2014*" becomes "*Meeting on Tuesday 10 September 2014*"). This is again done because of the imperfect work of syntactic analyzer. For example, having the text "*Car accident happened on 10 September, 2014 in Krakow*", the syntactic analyzer assigns "*10 September*" with a temporal semantic role while the year "*2014*" is assigned with a different role or can be not included into the dependency tree at all. Such behavior is caused by the presence of comma before the year.

3.2.2 Temporal Tagging

The task of temporal tagging is to identify sentences that have explicit non-ambiguous temporal expressions and resolve these expressions to actual timestamps. Temporal expression can be a single word (e.g. *today*) or a sequence of words (e.g. *on Friday morning*), and be represented in relative times (e.g. *last Sunday*) or absolute times (e.g. *15 July 2015*).

Temporal expressions are recognized and resolved by the English temporal tagger *SUTime* which is available as a part of a *Stanford CoreNLP* pipeline [8]. *SUTime* is chosen because it is a part of a single *Stanford* annotation pipeline, which is also used for text segmentation in this thesis. Beneficially, evaluation results show that *SUTime* system outperforms state-of-the-art techniques [8], producing high quality results compared to other systems [37, 45]. *SUTime* performs annotation of temporal expressions using *TIMEX3* [97] format that is suitable for further manipulation and usage. Input to the temporal tagger is a text of a single sentence. Relative temporal expressions within the text are resolved using article's publication date as a reference date (Figure 3-6).

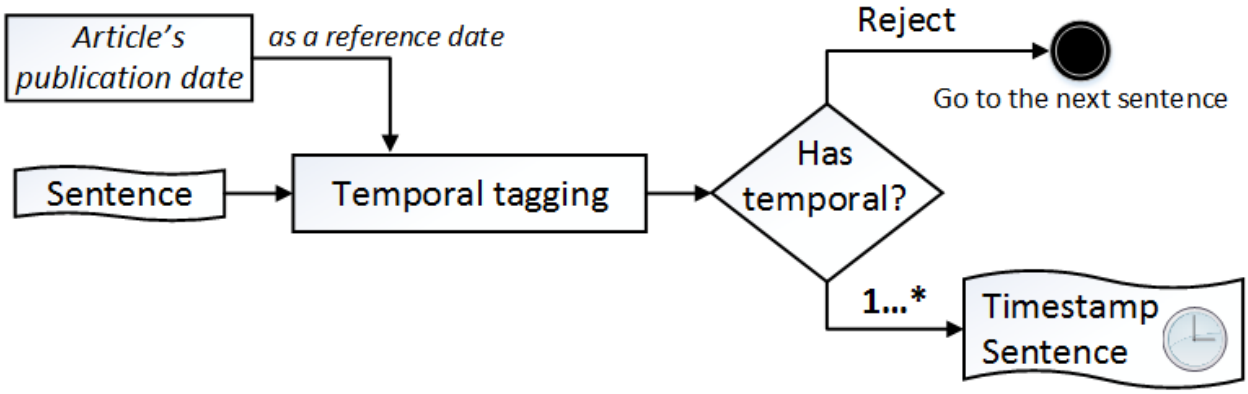

Figure 3-6. Temporal tagging of sentences

For each of the detected and resolved temporal expressions, the process of temporal tagging creates a *timestamp sentence*. As sentence can have more than one temporal expression within the text, several *timestamp sentences* can be created from a single sentence. Sentences that do not contain temporal expressions are rejected.

Timestamp sentence is a wrapper around the sentence and includes such information as:

- Text of the input sentence;
- Text of the recognized temporal expression within the sentence;
- Resolved date of the recognized temporal expression;
- Reference date used to resolve the temporal expression.

Depending on the role of the temporal expression in the sentence, *SUTime* assigns one of the supported temporal relation roles, such as *DATE*, *TIME*, *DURATION*, *RANGE* and *SET*. Table 3-1 shows examples of recognized temporal expressions, assigned *TIMEX3* role, as well as normalized resolved value using Monday, 14 April 2015 as the reference date (2015-04-13).

Table 3-1. Examples of resolved temporal expressions by SUTime

Temporal Expression	Temporal Role	Normalized Value
12 December 1974	DATE	1974-12-12
last Monday	DATE	2015-04-06
next Monday	DATE	2015-04-20
Christmas Eve	DATE	2014-12-24
Martin Luther King Day	DATE	2015-01-10
Winter of 2014	DATE	2014-WI
August	DATE	2015-08
now	DATE	PRESENT_REF
the day	DATE	2015-04-13
the day after tomorrow	DATE	2015-04-15
Saturday morning	TIME	2015-04-18TMO
early Tuesday morning	TIME	2015-04-14TMO
7:18 pm	TIME	2015-04-13T19:18
this afternoon	TIME	2015-04-13TAF
3 days	DURATION	P3D
a few months	DURATION	PXM
10 to 15 hours	DURATION	PT10H/PT15H
from 2010 to 2014	RANGE	2010/2014
every year	SET	P1Y
every third Saturday	SET	XXXX-WXX-6

Granularity of Temporal Expressions

In the context of this work, time component of an event is a required component. Temporal expressions with role *DURATION*, *RANGE* and *SET* are discarded because they do not provide specific information about the time when an event happened.

Some news articles provide highly detailed temporal information. The most specific temporal expressions include time up to a minute scale (e.g. "*12:07 a.m. 15 July 2015*"), while others indicate part of the day (e.g. *morning*, *late evening*, *night*). Although "*minute*", "*hour*" or

"*part-of-the-day*" scales can be used to indicate event date and time, vast majority of news articles do not provide such detailed information. The majority of events are reported on the day level, referring to the past, present or future date. Therefore, in order not to lose recall of events, the granularity of event dates is chosen to be on the "*day*" level. The normalized value of a resolved temporal expression is parsed to obtain the day, month and year information. The decision was made not to generalize expressions that provide more detailed temporal information about an event, but to keep this information for future reference. Temporal expressions that refer to larger periods (weeks, seasons, years, etc.) are discarded.

Elimination of Ambiguous Temporal Expressions

The known limitation of *SUTime* and other rule-based temporal taggers is poor handling of ambiguous phrases [8]. The system considers the phrase as a temporal expression while the actual semantic meaning is different. For example, consider the sentence "*Chris Weitz presented today the second film of the studio's TWILIGHT film franchise*". Although it is clear that "*TWILIGHT*" here is not a temporal expression, a temporal tagger recognizes it as a date, resolving to *2015-04-13TEV* using *2015-04-13* as a reference date.

Additionally, even from valid temporal expressions it is not always possible to correctly resolve the date. For example, consider the sentence "*William Jonson was pulled over in the early morning hours and arrested on suspicion of DUI, police said on interview on Tuesday"*. In this example, "*early morning hours*" is recognized as a temporal expression, which is in fact correct recognition. However, correct resolution of the particular temporal expression requires knowledge of the interviewing event. Therefore, in the context of this thesis, event extraction utilizes only exact non-ambiguous day-level temporal expressions. "Exact" means that temporal expressions have to be supported by an absolute date or explicit relative date. Otherwise, the temporal expression is considered as ambiguous. Table 3-2 shows some examples of ambiguous and equivalent non-ambiguous temporal expressions.

Verification of temporal expression for ambiguousness is made by a set of regular expressions. Regular expressions detect the presence of numbers, weekdays (e.g. "*Monday*", "*Tue.*", "*Wed*"), months (e.g. "*August*", "*Jun*", "*Sep.*"), ambiguous (e.g. "*morning*", "*night*", "*year*", "*daylight*") and explicit (e.g. "*this*", "*last*", "*next*", "*yesterday*", "*tonight*") temporal components. Depending on the combination of detected components, it is decided if the temporal expression is ambiguous to find the exact date. All uppercase words are also ignored (e.g. *TODAY* (*name of the newspaper*)). Temporal expressions that are recognized as ambiguous are rejected and not utilized for the construction of *timestamp sentences.*

Table 3-2. Ambiguous and equivalent non-ambiguous temporal expressions

Ambiguous temporal expressions	Non-ambiguous temporal expressions
at 5 a.m.	at 5 a.m. **today**
at 6 p.m.	at 6 p.m. **this Monday**
at 15:45	at 15:45 **yesterday**
around 15:45	around 15:45 **two days ago**
early morning	early morning on **15 May**
late evening	late **this** evening
in the early morning	in the early morning on **last Tuesday**
one year ago	**this day** one year ago
two weeks ago	10 days ago
more than month ago	10th of previous month
the same night	**tonight**
the day	**today**
daylight	**today**
wed *(wedding)*	–
the eighth day	–
last minute	–

Temporal Expressions Referring to the Reference Date

Majority of news articles are published within the interval of 1-2 days after the event happened. However, some articles are published on the same date as the event happened. If this is the case, articles usually utilize such temporal expressions as *today* or *this afternoon* to refer the event date. However, some news articles refer to events by the weekday name, even knowing that it happened within the given publication day; while the event itself is described using the past tense verbs.

This represents a borderline situation when the temporal expression is represented by the weekday (e.g. *Friday*) and because the article is published on the same date, the reference date is represented by the same weekday (e.g. *Friday*). This situation triggers a caution, as the temporal expression can be potentially resolved to the date on the previous week, while the correct resolved date is the same as the reference date. The given situation can be represented by the sentence "*The fire service was called at 06:00 to the Waitrose store in Wellington, on Sunday.*" that was published on the same day (*on Sunday*).

Fortunately, investigation of the resolution of such cases showed that event dates are resolved to the correct date. Further investigation showed that temporal tagger by default resolves temporal expressions to the dates on the same week to which the reference date belongs. This behavior can be overridden by using explicit weekday specifiers, such as *last*, *this* and *next* (e.g. *this Friday*).

3.2.3 Dependency-Based Semantic Role Labelling

All sentences that contain non-ambiguous temporal expressions are passed to the syntactic and semantic analysis. The purpose of analysis is to determine the structure of the input sentence and identify the semantic relations between different sentence parts. The analysis is ideally aimed at answering the question "*Who did what to whom, when, where, through what methods (instruments), and why?*" In the context of this work, event extraction is primarily focused on such event details as subject (*Who*), object (*What*), time (*When*) and place (*Where*) of the predicate (*did*). To extract the required information, the sentence is annotated by dependency-based semantic role labelling (Figure 3-7). Event details are then represented by and extracted from the resulting semantic arguments.

The input to the analysis is a *timestamp sentence* from the temporal tagging step. The output is a collection of dependency trees with annotated semantic roles. Resulting dependency trees represent open domain facts that happened on the resolved date of the *timestamp sentence*. The analysis starts with part-of-speech tagging and dependency parsing. The received dependency trees are then passed to the semantic labeler. The whole process is performed using *ClearNLP* as a language-processing library.

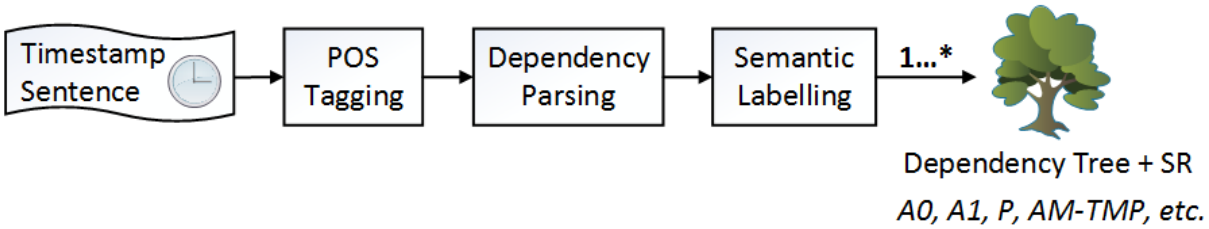

Figure 3-7. Dependency-Based Semantic Role Labelling

Based on observations, grammar of sentences in news articles has an overwhelming reliance on complex structures. Sentences are often written in syntactically rich way, encompassing many event details, statements and facts. Consider the following example: "*According to John Childs and the security advisory Microsoft also published today, the vulnerability affects all supported versions of IE, from the 12-year-old IE6 to the not-yet-officially-released IE11, the browser that will accompany Windows 8.1 when it ships 18 October*". There are four facts within the same sentence: (1) *Microsoft published security advisory today*, (2) *The vulnerability affects all supported versions of IE*, (3) *IE browser will accompany Windows 8.1* and (4) *Windows 8.1 ships on 18 October*. While extracting facts from the sentence it is important to link temporal expressions to the correct facts. Inability to do so will result in the incorrect event extraction.

Verification of the Temporal Argument

Syntactic and semantic analysis takes as input *timestamp sentence*, which is a wrapper around the sentence and includes information about the recognized temporal expression. Syntactic parsing returns dependency tree for each predicate within the sentence. Returned dependency trees are then filtered out to leave only those trees that have a required temporal expression as the temporal argument (*AM-TMP*). This way the resolved date of the temporal expression is linked to the factual events represented within the sentence. If there is no *AM-TMP* argument within the dependency tree then such dependency tree is rejected.

In order to filter dependency trees it is required to compare the text of the *AM-TMP* argument and the text of the temporal expression. The simplest way is to check if *AM-TMP* simply contains the required text within its argument phrase. For example, consider the recognized temporal expression "*11 March 2015*" from the sentence "*An Atlanta judge sentenced Brian Nichols to life in prison on 11 March 2015*". Dependency tree for this sentence has *AM-TMP* argument equal to "*on 11 March 2015*". One thing to notice is that *AM-TMP* argument is not always the same as the text of temporal expression. However, in the described example, temporal expression is fully contained within the *AM-TMP* phrase. Nevertheless, due to imperfection of semantic analysis, some parts of temporal expression can be assigned with different semantic roles. For example, consider a sentence "*Burglar was standing on crosswalk around 7 p.m. Wednesday night*" (Figure 3-8). Temporal tagger recognizes "*7 p.m. Wednesday night*" as a temporal expression. However, dependency tree marks only "*Wednesday night*" with *AM-TMP* argument, while "*7 p.m.*" is marked with *AM-LOC* argument.

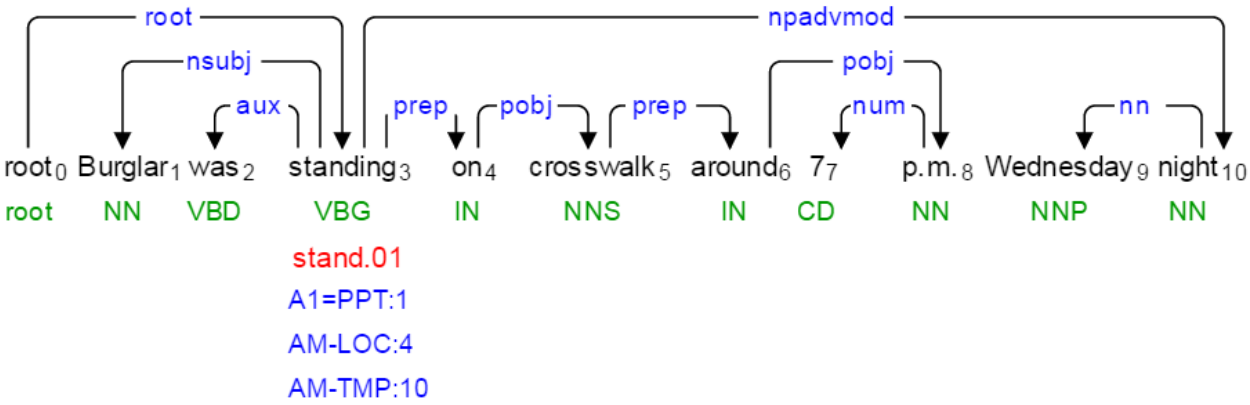

Figure 3-8. Verification of the temporal argument

In order not to miss events due to such erroneous behavior, a more complicated intersection approach is implemented. The algorithm returns, preferably maximum, date mention from the temporal expression that also appears in the given test phrase. The algorithm always returns textual expression that appears in both test phrase and required temporal expression. First, the temporal expression is verified if it is fully contained in the test phrase. Second,

absolute dates (e.g. "*10 August*", "*10 August 15"*, "*10 August 2015"*) are extracted from the temporal expression and these dates are verifies for containment in the test phrase. Finally, weekdays (e.g. "*today"*, "*Yesterday*", "*Monday*") are extracted from the temporal expression and also verified for containment. For the weekdays, the weekday surrounding is also extracted (e.g. "*last*" in "*last Monday*" or "*evening*" in "*Friday evening*"). Described verifications are made in the order they listed. Algorithm returns the common date mention, which can be the initial temporal expression, date, weekday or an empty string.

Elimination of Subordinate Temporal Arguments

Subordinate clause (also called dependent clause) is a clause that provides the main clause with additional information. It cannot stand as a complete sentence because it does not express a complete thought. However, it contains both a subject and a verb. It is then unclear whether temporal expression refers to the verb in the main clause or to the verb in the subordinate clause.

To illustrate the problem, consider the sentence "*Actor restored reputation after he supported a church, which was damaged on Friday*". There are three predicates found in the sentence: *restore*, *support* and *damage*. Two of them have temporal arguments. The first predicate *restore* has a long temporal argument "*after he supported a church, which was damaged on Friday*" (Figure 3-9) and the third predicate *damage* has a temporal argument "*on Friday*". The recognized temporal expression "*on Friday*" belongs to both temporal arguments. However, it refers only to the moment when a church was damaged and is not valid for the predicate *restored*.

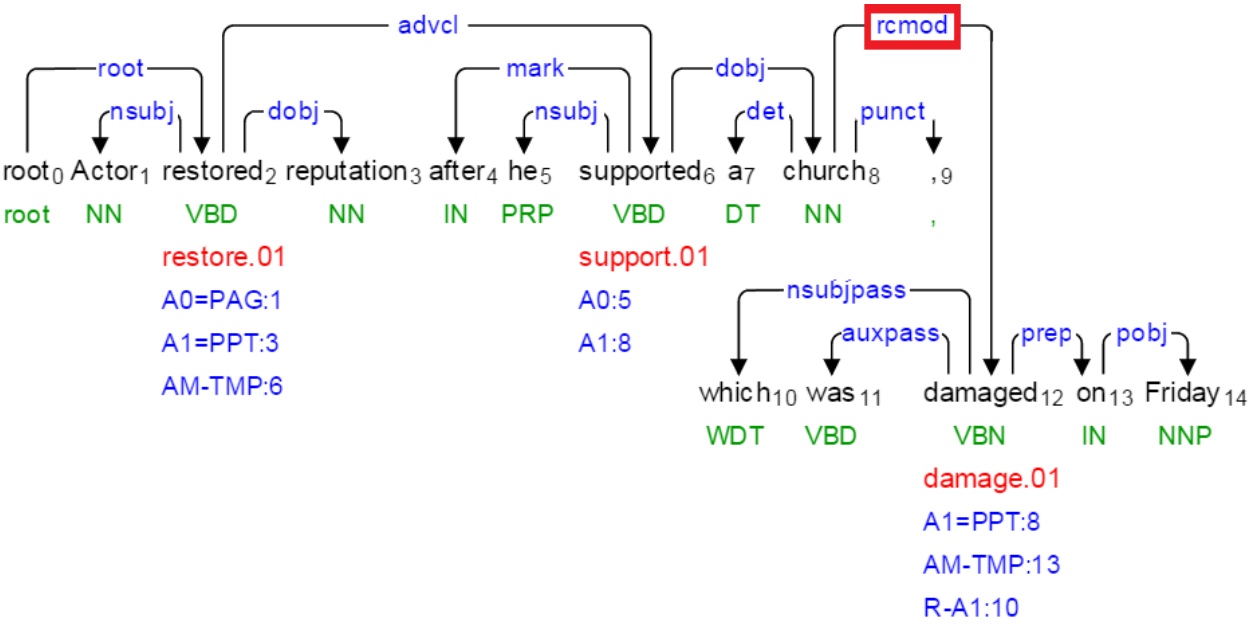

Figure 3-9. Subordinate temporal argument

The presented problem is solved by considering the relative placement of the temporal expression and the predicate. The temporal expression must appear in the same clause as the predicate in order to represent an associated time. Otherwise, temporal expression is ignored for the particular predicate. In the given example, temporal expression "*on Friday*" appears in the relative clause modifier (*rcmod*), while predicate *restored* appears in the main clause (*root*). In this case, temporal expression for the predicate *restored* is ignored. On the other hand, predicate *damaged* appears in the same relative clause as the temporal expression. In this case, the temporal expression represents a valid time for the correct predicate. The descriptions of dependency type labels used are given in Appendix A.2.

Adjustment of the Resolved Dates using Tense of the Predicate

Unfortunately, temporal tagger does not always correctly resolve temporal expressions. For example, consider the sentence "*The 4th parliamentary election of Turkmenistan ended on Sunday evening*" which was published on *Monday 2015-04-13*. The temporal expression "*on Sunday evening*" is resolved to the "*Sunday 2015-04-19"* whereas the correct event date is "*Sunday 2015-04-12*" (e.g. *yesterday*).

The resolved date can be verified and adjusted by considering the tense of the corresponding predicate. In the given example, the temporal expression is linked to the predicate *ended*, which has the *VBD* as part-of-speech tag (Figure 3-10). *The Penn Treebank* [72] defines *VBD* as the past tense of a verb. By knowing the tense of the predicate, it is possible to check whether the resolved date refers to the date ahead of or prior to the reference (publication) date. In the current example, the resolved date *2015-04-19* is six days ahead of the publication date. It can be automatically adjusted to the correct date *2015-04-12* by subtracting one week.

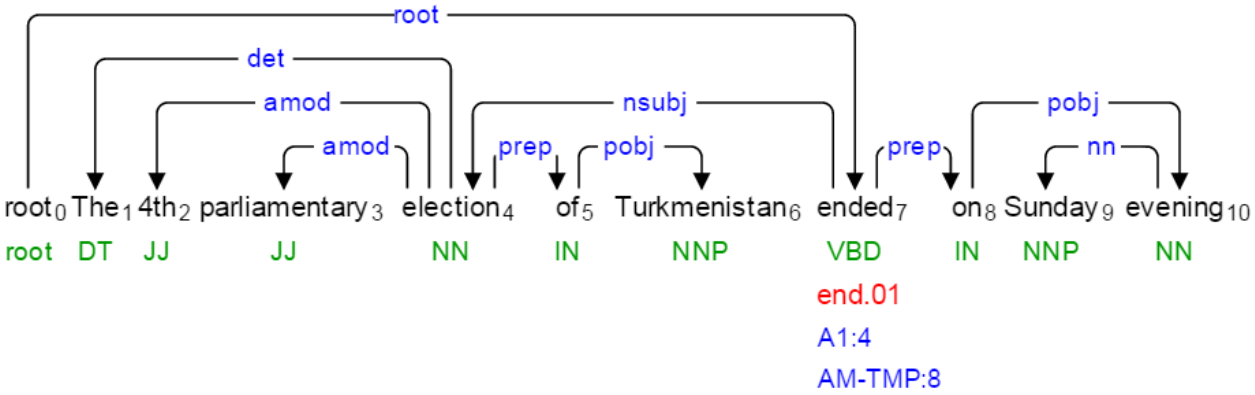

Figure 3-10. Usage of VBD predicate to adjust the resolved date

Correct adjustment of the resolved date depends on the granularity of the temporal expression. If the temporal expression represents a weekday, such as "*on Sunday"* in the given example, then it is required to adjust the resolved date by one week depending on the future or

past tense of the predicate. However, if the temporal expression represents an absolute date, such as "*on Tuesday, 14 April*", then it is required to adjust the resolved date to the *April 14* of the previous, current or next year. The adjustment procedure is illustrated in Algorithm 1. The algorithm adjusts the resolved date only if it contradicts to the tense of the predicate. If the temporal expression represents full date mentions, no date adjustment is needed.

Algorithm 1. adjustResolvedDate(ts, pos)

Input: ts – instance of *timestamp sentence*
pos – part-of-speech of the predicate
Output: adjusted *resolved* date

```
1:  let expr be ts.temporalExpression
2:  let resolvedDate be ts.resolvedDate
3:  let referenceDate be ts.referenceDate
4:
5:  if expr represents full date mention then
6:      return resolvedDate
7:
8:  else if expr represents absolute date then
9:      if (resolvedDate is after referenceDate) and (pos is past tense) then
10:         resolvedDate = resolvedDate.minusYears(1)
11:     else if (resolvedDate is before referenceDate) and (pos is future tense) then
12:         resolvedDate = resolvedDate.plusYears(1)
13:
14: else if expr represents weekday then
15:     if (resolvedDate is after referenceDate) and (pos is past tense) then
16:         resolvedDate = resolvedDate.minusWeeks(1)
17:     else if (resolvedDate is before referenceDate) and (pos is future tense) then
18:         resolvedDate = resolvedDate.plusWeeks(1)
19: return resolvedDate
```

3.2.4 Construction of Event Mentions

This section describes the construction of *event mentions* and identifies semantic arguments that can be used to extract important event details. After completion of the dependency-based semantic role labelling, each *timestamp sentence* has a number of semantically annotated dependency trees. The aim is to extract from these dependency trees the human-readable event phrase, subject, object and location.

Based on observations, text of the sentences quite often represents a statement about an event made by a third person or a party. For example, consider the sentence "*The five patients were still receiving treatment on Monday, Lu said"*. In this example the actual event is contained in the statement and not the declaration itself. Temporal tagging of the sentence recognizes a

temporal expression *on Monday*. Syntactic analysis returns two dependency trees: one for the predicate *receive* and another for the predicate *say*. However, because the second predicate is not associated with the temporal expression, the corresponding dependency tree is rejected. Now consider the case when both predicates contain temporal arguments as in the sentence "*The five patients were still receiving treatment on Monday, Lu said on interview on Tuesday*". Syntactic analysis returns two valid dependency trees: one for the predicate *receive* with temporal expression *on Monday*, and another for the predicate *say* with temporal expression *on Tuesday*. The second dependency tree represents another event for another date with object speaking about the treatment of five patients. The proposed algorithm constructs *event mention* for each pair of predicate with temporal expression. Filtering of *reporting* events is not performed.

Extracting Event Phrase

Event phrase is represented by the ordered sequence of semantic arguments. Semantic arguments are ordered in the same way they appear in the original sentence. Text of semantic arguments is extracted by getting head-tokens of arguments and traversing the dependency subtrees one node at a time in the sorted order. The resulting event phrase is typically shorter than the original sentence.

The returned *event mention* consists of the semantic arguments, which together represent an event phrase, and a *timestamp sentenc*e (Figure 3-11), which was used for the extraction of dependency tree. *Timestamp sentence* represents temporal information for an event, such as the date when an event happened. Additionally, it is used to trace the origin of *event mention* and link it to the news article.

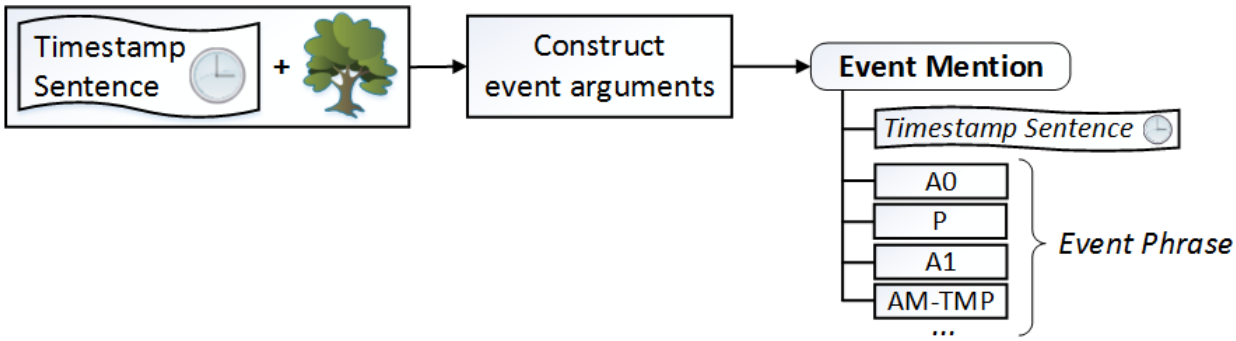

Figure 3-11. Construction of event mentions

Extracting Event Details

Event details (e.g. subject, object, location) can be extracted from the appropriate semantic arguments. Table 3-3 provides a mapping of semantic arguments that can be used to represent one or another event detail.

Table 3-3. Mapping event details to semantic arguments

Event Details	Semantic Arguments
Subject	A0
Object	A1, C-V, AM-PRD
Location	AM-LOC, AM-DIR
Time	AM-TMP

Subject of an event can be extracted from the *A0* semantic argument as it clearly represents the role agent of the predicate. Object-related information, on the other hand, can be represented by several semantic arguments. *A1* semantic argument represents patient or theme of the predicate, so is used as a primary source when extracting an object. Such semantic arguments as *AM-PRD* and *C-V* can also be considered as object-related arguments. Relevance of these arguments can be demonstrated by the following examples. In the sentence "*Famous star showed around naked on Wednesday*" the phrase "*naked*" is considered as an object of the predicate *showed* (Figure 3-12) and marked with *AM-PRD* semantic argument.

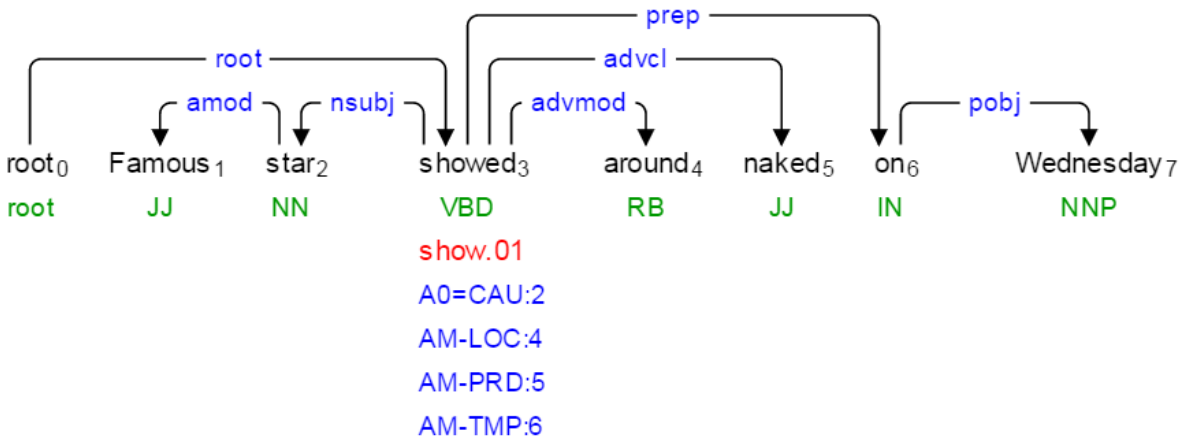

Figure 3-12. Example of AM-PRD argument

In the sentence "*John Boehner weighed in himself on Tuesday, saying that all children ought to be vaccinated*" the phrase "*in himself*" is considered as an object of the predicate *weighted* (Figure 3-13) and marked with *C-V* semantic argument.

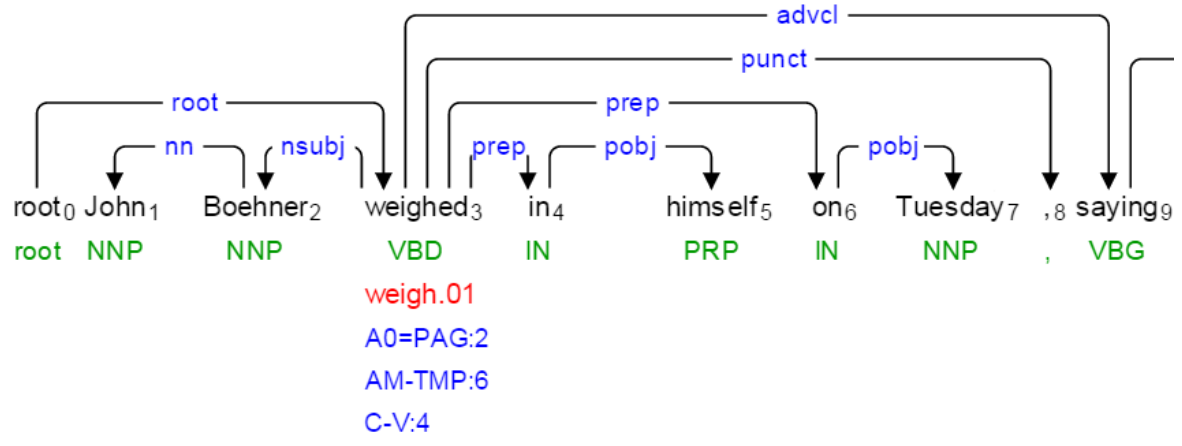

Figure 3-13. Example of C-V argument

Location information can be extracted from such semantic arguments as *AM-LOC* and *AM-DIR*. An example of *AM-LOC* argument is given using the sentence "*Wladimir Klitschko defended WBO and IBF titles in Germany today*" (Figure 3-14). The phrase "*in Germany*" is considered as the location for the predicate *defended*. Example of *AM-DIR* argument is given using the sentence "*On Monday a group of people were marching toward the London 67th Police station*" (Figure 3-15). The phrase "*toward the London 67th Police station"* is considered as the direction of the predicate *marching*. Named Entity Recognizer can be applied on these arguments to extract names of locations.

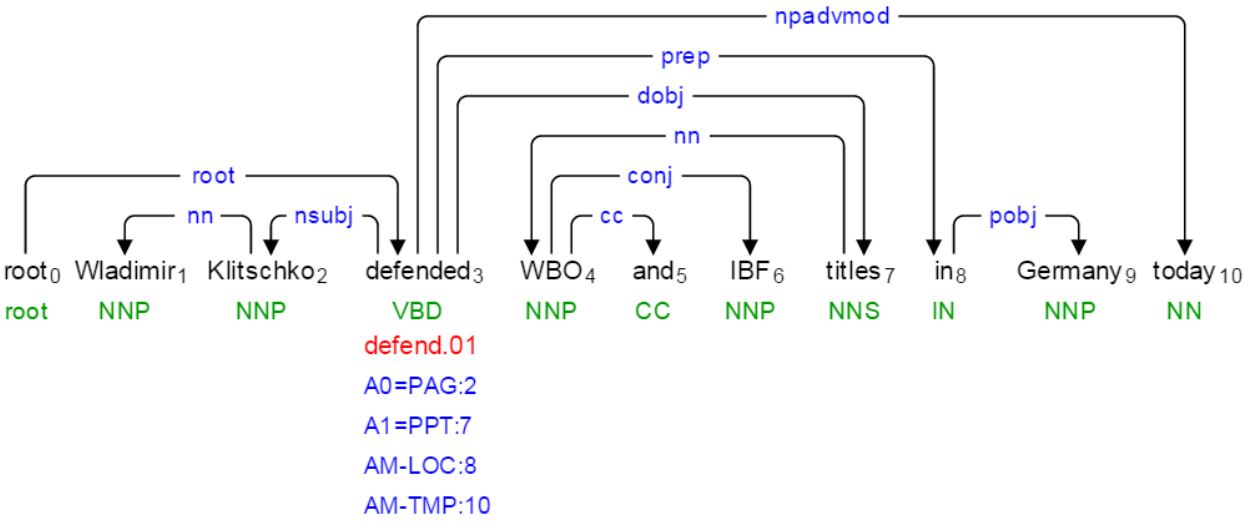

Figure 3-14. Example of AM-LOC argument

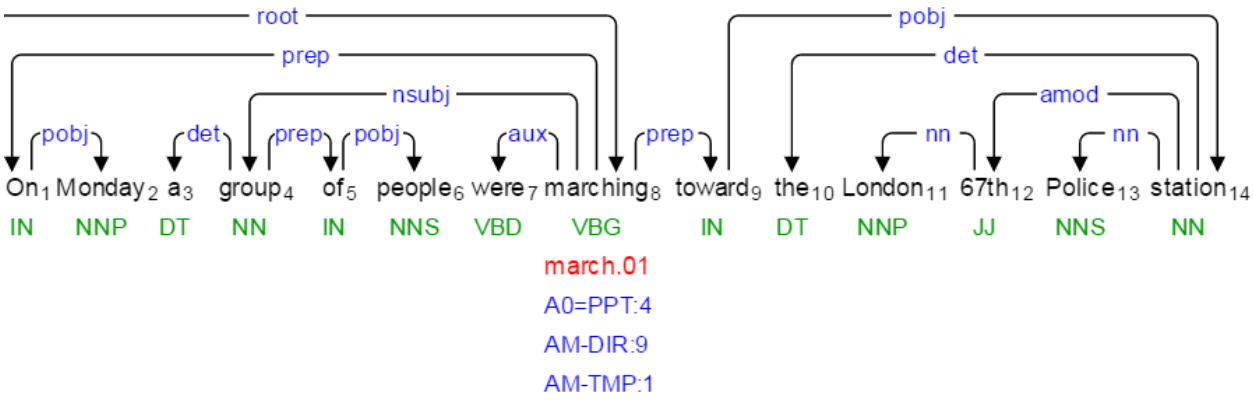

Figure 3-15. Example of AM-DIR argument

Time information can be obtained directly from the *event mention*. Recognized temporal expression and resolved date are stored separately from the semantic arguments. The process of syntactic and semantic analysis ensured that each dependency tree contains *AM-TMP* argument that corresponds to the temporal expression. In general, the text of *AM-TMP* argument contains extra temporal details. Some examples are "*on Saturday night after pointing a gun at two plainclothes officers*", "*during a shooting on Friday afternoon*" and "*following an announcement made yesterday*".

3.3 Extraction of Cluster Events

This section provides an overview of c*luster events* extraction process (Figure 3-16). *Cluster events* are unique events within the particular news cluster. The input to the process is the collection of all *event mentions* extracted from all news articles of a news cluster. These *event mentions* are grouped into *cluster events* using one of the supported grouping methods described in Chapter 4. *Cluster events* track the identifier of a cluster they are extracted.

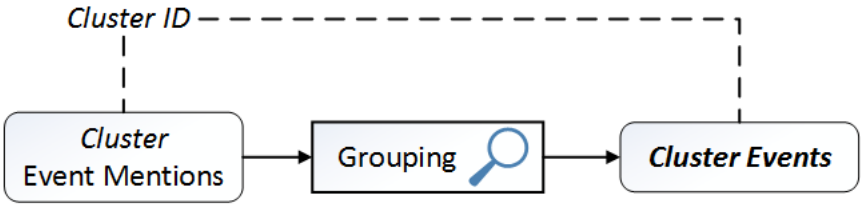

Figure 3-16. Extraction of cluster events

3.4 Extraction of Global Events

This section provides an overview of *global events* extraction process (Figure 3-17). *Global events* are unique events within the repository. The process of extraction is very similar to the previous step. The main difference is that an input to the process is the collection of all *cluster events* extracted from all available news clusters. These *cluster events* are grouped into *global events* using one of the supported grouping methods described in Chapter 4. Extracted *global events* are stored in the database and then displayed to the user in the calendar view.

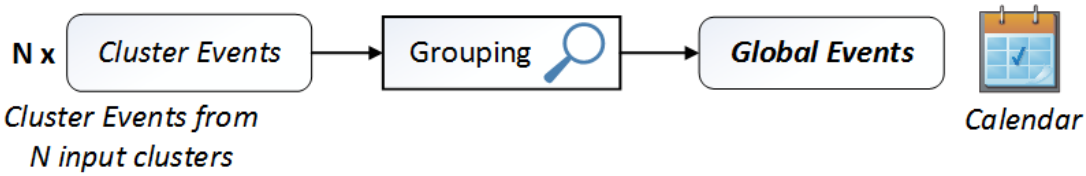

Figure 3-17. Extraction of global events

3.5 Determination of the Event Representative

This section describes an approach to select representative phrase for *cluster events* and *global events*. Each of the resulting events consists of a unique ID, normalized temporal expression denoting the date when an event happened, and a list of textual representations.

By its nature, *cluster events* are just a collection of *event mentions* grouped together. Each *event mention* contains a collection of semantic arguments along with a temporal expression. In other words, *cluster events* are represented by a list of predicate-arguments tuples which presume to refer the same actual event.

The representative event phrase is identified as follows:

- First, the most common predicate across all *event mentions* is identified;
- Then the algorithm selects the shortest *A0* and *A1* semantic arguments from a subset of *events mentions* that have the most common predicate as their predicate;
- Location is identified as the most frequent *AM-LOC* across all *event mentions*.

The representative event phrase is constructed by concatenating the shortest *A0* argument, the most common predicate, the shortest *A1* argument and the most frequent location. Temporal expression is not included to the representative phrase as it's assumed to be outputted in the separate field. Such semantic arguments as *AM-DIR*, *C-V* and *AM-PRD* are not considered during the representative phrase construction. They can be beneficiary as a secondary source of information in case *A1* or *AM-LOC* arguments are missing. However, considering that *cluster events* consist of a large variety of *event mentions*, the chance of *A1* or *AM-LOC* argument to be missing is relatively small. Representative phrase for *global events* is constructed in the same way considering all *event mentions* from all grouped *cluster events*.

Summary

This chapter has discussed the algorithm for building a repository of *global events* out of topically related clusters of news articles. An algorithm consists of five principal steps that were discussed in details. The process begins with retrieval of news clusters to be used as input data. The principal goal of the next step is to extract *event mentions* out of individual news articles. The step is performed by recognizing temporal expressions within sentences using temporal tagger *SUTime*. Detected temporal expressions are resolved to actual timestamps using article's publication date as a reference date. The text of articles is preprocessed before the temporal tagging in order to maximize the recognition of temporal expressions and facilitate the following syntactic and semantic analysis. Text preprocessing includes text cleaning and dates normalization. Ambiguous temporal expressions are filtered out. Sentences that have temporal expressions of the "day" granularity are passed to the dependency-based semantic role labelling. Resulting dependency trees represent open domain facts that happened on the resolved date of temporal expression. Annotated semantic arguments are analyzed and mapped to appropriate event details. Event phrase is constructed by ordering the list of extracted semantic arguments. In subsequent steps, *cluster event mentions* are grouped into *cluster events*, and *cluster events* into *global events* using one of the supported grouping methods. Event representative is identified for each event using the most common predicate and head arguments.

Chapter 4

Grouping Methods

This chapter describes different methods of grouping (1) *event mentions* into *cluster events*, and (2) *cluster events* into *global events*. Each of the grouping methods can take as input either *event mentions* or *cluster events* and provide as output *cluster events* or *global events*, respectively. However, in the context of the provided descriptions, it is assumed that the task of *event mentions* grouping is performed. All grouping methods assume that inputted *event mentions* (1) are from the same cluster, and (2) happened on the same day (have the same resolved date). The necessary adaptations for grouping *cluster events* into *global events* are described in Section 4.4.

4.1 Using Proper Nouns

This grouping method represents the simplest approach which is used as a baseline metric for other grouping methods. The collection of inputted *event mentions* is represented as a graph. Each *event mention* represents a unique vertex. An undirected edge between two *event mentions* is created if these *event mentions* have enough number of common proper nouns or both of them do not have proper nouns at all.

Let $ProperNouns(A)$ be the set of proper nouns of an *event mention* A. Proper nouns include names, titles, organization and locations mentioned within the corresponding event phrase. Let N be the parametrized required minimum number of common proper nouns. An undirected edge between two *event mentions* A and B is created if:

$$|ProperNouns(A) \cap ProperNouns(B)| \geq N$$

$$or$$

$$(ProperNouns(A)\ is\ empty\ AND\ ProperNouns(B)\ is\ empty)$$

The second condition is necessary to group *event mentions* that do not have proper nouns at all. An example of such *event mention* is "*The fire destroyed church on yesterday evening*". All such *event mentions* are grouped into a single *cluster event*. Otherwise, they will constitute isolated event instances greatly affecting evaluation results. An additional intuition is that

grouping of such *event mentions* into one *unclassified* event ("*Rag Bag*") is less harmful than introducing disorder into a clean event [25]. After the graph is constructed, the grouping method finds connected components in a graph. Each extracted connected component represents a separate *cluster event* with vertices being the *event mentions*.

4.2 Using Word Alignment

Word alignment is the process of identifying words that correspond in meaning in the source and target phrases. By experimenting with monolingual word alignment tool *Jacana.Align* [100], the similarity of two *event mentions* is determined as the number of aligned content words. The given tool was selected as it provides the state-of-the-art performance and faster than previous works [100]. The speed of execution is crucial as there is a need to compute word alignment between many pairs of *event mentions* for larger news clusters.

Single run of the tool models an alignment from the source sentence to the target sentence. In order to obtain the full alignment information it is required to run the aligner in both directions. Word alignment uses the broad specter of similarity features and able to determine words that have the same meaning considering the context of sentence. Figure 4-1 shows an example of alignment of the source sentence "*They married last year*" to the target sentence "*They got married one year ago*" in which all words from the source sentence are aligned to the words of the target sentence.

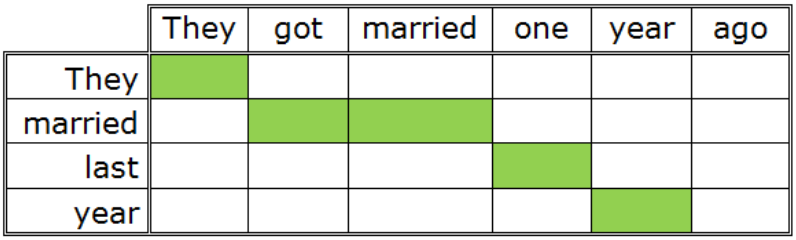

Figure 4-1. Example of word alignment

As well as in the previous grouping method, the collection of inputted *event mentions* is represented as a graph. Let N be the parametrized required minimum rate of aligned words. Let $AlignedWords(A, B)$ be the number of aligned words from the event phrase of the source *event mention* A to the event phrase of the target *event mention* B. Let $WordCount(E)$ be the number of words in the *event mention* E. An undirected edge between two event mentions A and B is created if:

$$\frac{max(\ AlignedWords(A,B),\ AlignedWords(B,A)\ )}{min(\ WordCount(A),\ WordCount(B)\ )} \geq N$$

The numerator defines the maximum number of aligned words from the two respective comparisons of event phrases. The denominator defines the minimum number of words in either of event phrases. The obtained result defines the rate of aligned words taking into account possible one-to-many mappings of a single word. After the graph is constructed, the grouping method finds connected components in a graph. Each extracted connected component represents a separate *cluster event* with vertices being the *event mentions*. Table 4-1 shows examples of *event mentions* grouped by event date and word alignment.

Table 4-1. Examples of event mentions grouped by event date and word alignment

Event Date	Event Mentions
2015-04-18	Beloved pin-up icon Bettie Page passed away
	Legendary pinup queen Bettie Page died
	Pin-up queen and pop culture icon who died
2015-04-18	Gray 16 died after police bullets
	Kimani Gray who was fatally wounded by police
	Sixteen-year-old Kimani Gray killed on Saturday

4.3 Using Cosine Similarity

This section describes two methods for grouping *event mentions* basing on the cosine similarity of their event phrases. Cosine similarity is a well-known similarity measure and its explanation is not included into the scope the thesis. The particular weighting scheme used to represent vectors of *event mentions* is the subject of discussion that follows.

One grouping method that uses cosine similarity to group *event mentions* into *cluster events* is a bottom-up group-average agglomerative clustering [17]. Bottom-up clustering threats each *event mention* as a singleton cluster and then successively merges pairs of clusters until all clusters have been agglomerated into a single cluster that contains all *event mentions*. Each agglomeration choses the best merge available at the step. The steps of agglomerations are represented using a dendrogram, which allows reconstructing history of merges (Figure 4-2a).

By cutting the dendrogram at some predefined level of similarity, the one can obtain a set of disjoin clusters each of which contains *event mentions* with average similarity higher than the cut similarity. Each disjoint cluster then represents resulting *cluster event*. For example, the cut of the dendrogram at similarity equals to 0.5 produces two *cluster events*, while cutting at similarity 0.25 produces only one single *cluster event* (Figure 4-2a). Group-average clustering approach was selected because it evaluates cluster quality based on all similarities between *event mentions*, thus, avoiding such drawbacks as *chaining* and *outliers* [17].

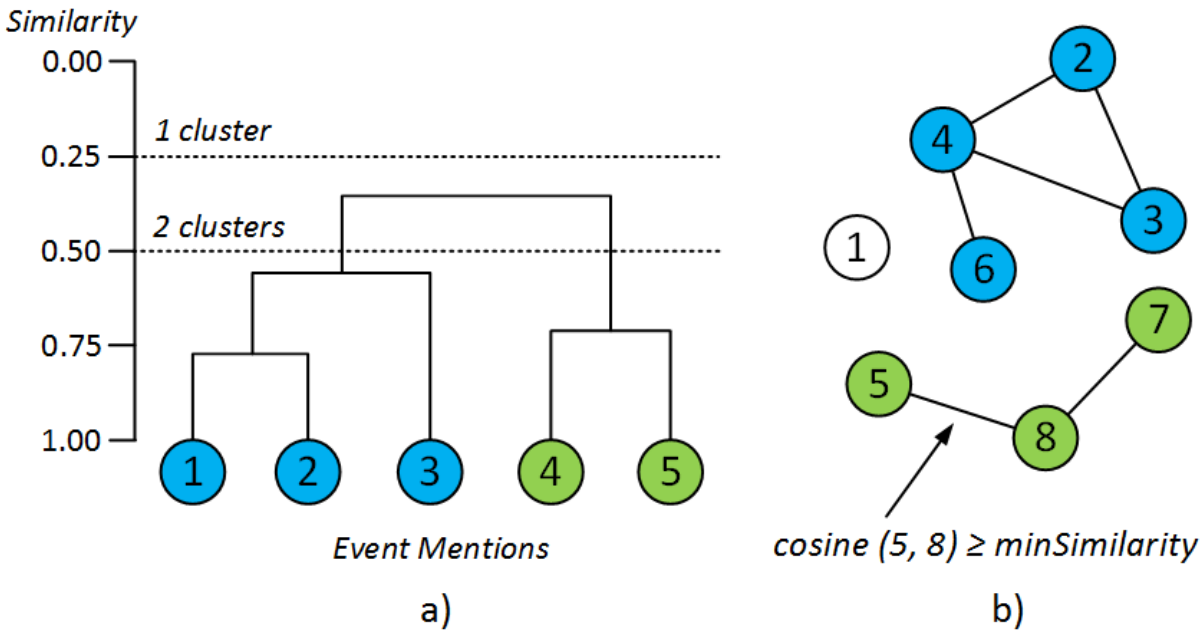

Figure 4-2. a) Dendrogram of the bottom-up agglomerative clustering
b) Graph with edges between event mentions defined by the minimum cosine similarity

Another grouping method uses cosine similarity in combination with connected components. The collection of inputted *event mentions* is represented as a graph. Let $minSimilarity$ be the parametrized required minimum similarity. Let $cosine(A, B)$ be the cosine similarity between *event mentions* A and B. An undirected edge between two event mentions A and B is created if:

$$cosine(A, B) \geq minSimilarity$$

After the graph is constructed, the grouping method finds connected components in a graph (Figure 4-2b). Each extracted connected component represents a separate *cluster event* with vertices being the *event mentions*. This grouping method is added to track the influence of usage of different features (proper nouns, word alignment, cosine similarity) while using the same underlying grouping mechanism.

4.3.1 Computing Cosine Similarity

Cosine similarity of *event mentions* is computed as a weighted sum of (1) cosine similarity of vectors representing lemmatized words and (2) cosine similarity of vectors representing proper nouns of corresponding *event mentions*. The two vectors are disjoint. First one represents event content, whereas the second represents event participants. Vectors of proper nouns include names, titles, organizations, locations and weekdays. Stop words and punctuations are removed. The resulting cosine similarity of *event mentions* A and B is calculated as:

$$cosine(A,B) = 0.4 * cosine.\boldsymbol{Lemmas}(A,B) + 0.6 * cosine.\boldsymbol{ProperNouns}(A,B)$$

Where $cosine.\boldsymbol{Lemmas}(A,B)$ is the cosine similarity of vectors representing event lemmas, whereas $cosine.\boldsymbol{ProperNouns}(A,B)$ is the cosine similarity of vectors representing proper nouns. Each of these similarities returns score in range $[0,1]$. The defined formula makes sure the *event mentions* have the same participants but also similar content. Identification and usage of proper nouns significantly improves similarity calculation and clustering [74]. However, the usage of named entities alone is not sufficient. The proper nouns have higher coefficient because it has positive effect on clustering and evaluation results. If either of *event mentions* does not have proper nouns then the overall cosine similarity is computed as the cosine similarity of lemmatized event phrases:

$$cosine(A,B) = 1.0 * cosine.\boldsymbol{Lemmas}(A,B)$$

4.3.2 Token Weighting Scheme

Event mention is represented using vectors of lemmatized words and proper nouns along with their weights. One vector stores proper nouns, while another stores the rest of the words. Token weights can be represented either by *TF.IDF* scores or by the *log-likelihood test* values. *Log-likelihood test* is said to perform better than *TF.IDF* or *chi-square* when dealing with varying text sizes [2], but this was not observed during the evaluation. The potential reason for this is that an algorithm works on the sentence level, where each word is usually reported only once. Therefore, there is no significance of the differences of a word's frequency in event phrases of varying size.

*Log-likelihood value*s were computed using word frequency lists based on 3 months of news. Usually log-likelihood values are filtered by the *p-value* in order to limit the size of the vector to the most important words. However, as in this work an algorithm deals with sentence-long event phrases of varying size, there are already small-sized vectors. Restricting the vectors to most important words in a corpus would result in even smaller vectors and potential loss of information, so no filtering by *p-value* was performed. Moving forward, *TF.IDF* is used as the default weighting scheme as it provides better similarity results.

4.3.3 Computing Word Frequencies

Vectors of *event mentions* include word lemmas based on their part-of-speech. This result in the vectors that can have the same lemma several times, but these instances are represented by different par-of-speech tags (Table 4-2).

Table 4-2. Examples of the same lemma with different part-of-speech tags

Lemma	Part-of-speech	Example sentence
surprising	Adjective	She earned a surprising amount of money
surprise	Verb	The news really surprised me
surprise	Noun	I prepared a surprise for you
beat	Verb	He beat Becker in the tennis championship
beat	Noun	Feel the beat of her heart
beat	Adjective	So beat I could flop down

Word frequencies and document counts, which are used for computing *TF* and *IDF* scores, are calculated for lemmas based on their part-of-speech tags. Additionally, they are calculated for each news category separately (Figure 4-3). This allows calculation of more appropriate *TF.IDF* scores for news-category-specific words.

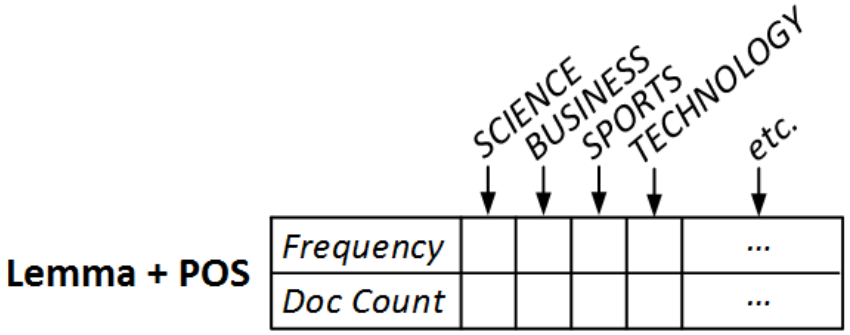

Figure 4-3. Word frequencies and document counts are calculated for each news category

4.3.4 Using WordNet

Event mentions can represent the same event using different words and with different level of details. This constitutes a difficulty for the similarity calculation as *event mentions* talking about the same actual event can be represented by disjoint feature vectors. This issue is resolved by considering synonyms (*synsets*) for lemmas using *WordNet* lexical database [30]. Part-of-speech information is utilized to obtain an appropriate set of synonyms, thus, limiting the scope of the search. Table 4-3 gives examples of synonyms obtained for the lemma "*surprise*" with two possible part-of-speech tags. Word sense disambiguation is not performed at this step. Synonyms are obtained only for nouns and verbs. Usage of other part-of-speech tags did not give any performance gain but only increased execution time.

Table 4-3. Examples of synonyms based on part-of-speech tags

Lemma	Part-of-speech	Synonyms
surprise	Verb	impress, affect, strike, act, move, attack, assail
surprise	Noun	amazement, astonishment, change, alteration, modification
surprising	Adjective	*not performed*

The feature vectors of *event mentions* are extended using the following approach. Having a pair of *event mentions* as input, as the first step an algorithm finds sets of lemmas that appear only in one *event mention* and not in the other. The second step is to find synonyms for these lemmas for each *event mention* separately. The algorithm then intersects two sets of obtained synonyms to find synonyms common for both *event mentions*. These common synonyms are then used to enrich feature vectors.

The weight of the synonym is the same as the weight of its source lemma in corresponding *event mention*. In case the same synonym satisfies several source lemmas then the weight of the synonym is an average of the corresponding weights. Only one synonym per lemma is used to enrich feature vectors. *WordNet* is utilized only for the vectors of lemmas and not used for the vectors of proper nouns.

4.4 Grouping of Cluster Events

Grouping of *cluster events* into *global events* requires several adaptations for each of the described grouping methods. The first difference is that there is only a requirement for the inputted *cluster events* to happen on the same date.

The set of proper nouns of a *cluster event* is just a union of proper nouns from all *event mentions* that belong to this *cluster event.* Grouping method using proper nouns then can be applied without any change to the implementation.

To group *cluster events* into *global events* using word alignment it is required that at least third part of all *event mention* pairs from different *cluster events* has a required minimum rate of aligned words. The specified rate was defined by considering sophisticated nature of word aligner and a broad specter of used similarity features.

To group *cluster events* using cosine similarity it is necessary to define feature vectors for *cluster events*. List of proper nouns and list of lemmas of a *cluster event* is created by combining, respectively, lists of proper nouns and lemmas from all *event mentions* that belong to this *cluster event.* Feature vectors are then created in the same way as for the *event mentions* utilizing either *TF.IDF* or *log-likelihood* weighting scheme. *WordNet* for grouping of *cluster events* is not utilized. This is explained by the fact that *cluster events* encompass a much higher variety of different textual representation, so the usage of synonyms is not clarified.

Summary

This chapter described different methods of grouping *event mentions* into *cluster events*, and *cluster events* into *global events*. Each of the grouping methods define specific grouping verification in addition to the requirements, in case of grouping *event mentions* into *cluster events*, of being from the same news cluster and describing events that happened on the same date. These additional verifications can be the requirements to have the minimum number of common proper nouns (grouping by proper nouns), the minimum rate of aligned words (grouping by word alignment) or the minimum required cosine similarity (grouping by cosine similarity). Grouping by proper nouns represents the simplest grouping methods used as the baseline metric for other grouping methods. Cosine similarity using *TF.IDF* weighting scheme provides better grouping results compared to the *log-likelihood test*. Each of the described grouping methods is evaluated in the subsequent chapter.

Chapter 5

Evaluation and Results

This chapter presents an evaluation of the proposed algorithm for the construction of a global repository of events. The quality of a constructed repository is determined by the quality of the extracted *cluster events* and *global events*. Therefore, the essence of this chapter is to conduct an evaluation of the event extraction process using different grouping methods.

The chapter is organized as follows: Section 5.1 describes the used evaluation corpuses and the process of annotation, and analyzes their representativeness in relation to intended language population. Section 5.2 formalizes the used evaluation metrics. Section 5.3 provides the quantitative evaluation of *cluster events* and *global events* extracted by different grouping methods. Sources of errors are analyzed and discussed in Section 5.4. Section 5.5 presents a simple WEB interface used to represent extracted *global events* as calendar entries. The summarizing overview of the most important aspects is given in the end of the chapter.

5.1 Evaluation Corpus

The only possible way to evaluate the extraction of *cluster events* and *global events* is to compare the output of the algorithm against events that were manually extracted on the same data. This process is usually done by annotating the *evaluation corpus* to mark which sentences represent which events with subsequent comparison of the extraction results using a *scoring program*. The comparison includes identification whether all annotated events were extracted by the algorithm and whether all *event mentions* were grouped to correct events.

This section describes two corpuses that are used to evaluate the construction of a global repository of events. Subsection 5.1.1 gives detailed explanation of how *Event Coreference Bank Plus* (ECB+) corpus was adapted to represent the required input. Subsection 5.1.2 explains how *real-word data* was annotated to be used as an evaluation corpus. Representativeness, comparison and usage of evaluation corpuses are described in Subsection 5.1.3.

5.1.1 Event Coreference Bank Plus

Event Coreference Bank Plus (ECB+) corpus [4, 5] is the largest publicly available annotated corpus for event coreference resolution. This thesis has a different objective than event coreference, but it is possible to reuse available annotations. ECB+ corpus is an extended version of *Event Coreference Bank* (ECB) corpus [22] used in many recent studies of event coreference resolution [13, 21, 32]. The new corpus in total includes 984 news articles describing 43 topics (seminal events) capturing descriptions of double event instances per topic. The corpus is annotated with event classes, specific types of entities, participants, time, locations, as well as with inter- and intra-document coreference between them. The ECB+ corpus annotation is an *event-centric* annotation [4]. All event components are annotated from the point of view of an event action. Annotation includes marking of:

- Involved participants as opposed to any participant mentioned within the sentence;
- Time when an action happened as opposed to any time expression mentioned in a text;
- Location in which the action was performed in contrast to a locational expression that does not refer to the place where an action happened.

For example, having a sentence "*People hate Mondays*" *event-centric* annotation does not mark *Mondays* as a time component of an event. In this case it is annotated as a non-human participant of an action *hate*.

Reusing Time Annotations

As in the context of this thesis, time component of an event is a required component, there is a particular interest in the available time annotations within the ECB+ corpus. The time component of events marks an explicit temporal expression. ECB+ time annotation includes annotation of four major time types: *DATE*, *TIME*, *DURATION* and *REPETITION* [4]. Each of the time types can be additionally marked as a *coreferent time expression*. Two or more time expressions corefer with each other if they refer to the same time. The following list provides examples of time types taken from the ECB+ annotation guidelines [4]:

1. *DATE* tag refers to calendar time:

June 11, 1989	*On Tuesday 18th*
Yesterday	*This summer*
Summer, 2002	*The second of December*
Last week	*Two days ago*

2. *TIME* tag captures expressions referring to a specific time of the day:

Ten minutes to three	*At 9 a.m. Friday, October 1, 1999*
At five to eight	*The morning of January 31*
At twenty after twelve	*Between 8 a.m. and 10 a.m.*
(late) Last night	*Today morning*

3. *DURATION* tag is meant for temporal expressions denoting durations:

2 months	*All last night*
48 hours	*20 days in July*
Three weeks	*3 hours last Monday*
For over a year	*Up to 30 days*

4. *REPETITION* tag is used for sets of times describing repeated events:

Often	*Every 2 days*
Frequently	*On the second Tuesday of every month*
Every Tuesday	*The second time*
Twice a week	*Two-time*

Any of the four time types described above does not correlate well with the specification of the temporal expressions utilized by the event extraction algorithm. Each of these types can represent valid exact non-ambiguous day-level temporal expressions as well as non-day-level granularity expressions or ambiguous ones (Table 5-1).

Table 5-1. Examples of valid and invalid temporal expressions for each ECB+ time type

Time type	Valid examples	Invalid example
DATE	*June 11, 1989; Yesterday*	*This summer; Last week*
TIME	*The morning of January 31*	*Ten minutes to three*
DURATION	*3 hours last Monday*	*Three weeks*

In this thesis, event extraction utilizes only exact non-ambiguous temporal expressions of the day granularity. Therefore, the annotated set of temporal expressions has to be filtered to keep only those annotations that are valid for the extraction algorithm. The real significant benefit of the provided time annotations is their event-centric nature.

Temporal expressions of the type *REPETITION* are fully filtered from the annotated corpus because they do not represent the date of a single event, but rather represent repeated events that are not supported by the event extraction. Surprisingly, the type *DURATION* can also include valid temporal expressions and also needs to be inspected.

Each of the annotated temporal expressions of type *DATE*, *TIME* and *DURATION* was manually inspected and discarded if it did not represent valid temporal expression. Table 5-2 provides initial and resulting number of time annotations for each time type in the whole corpus.

Table 5-2. Number of time annotations in the ECB+ corpus

Time type	Initial number of annotations	Number of annotations after filtering
DATE	1332	885
TIME	672	523
DURATION	313	2
REPETITON	95	0

Time annotations that were discarded are mostly represented by the time, week, month, season and year temporal expressions (e.g. "*at around 9 a.m.*", "*last week*", "*this season*", "*next March*", "*in September*", "*month ago*", "*1940s*"). Some entries were filtered because they do not represent time components of an event (e.g. expressions like "*today's challenges*", "*to spend the night*" and "*the date has not been confirmed*"). The remaining temporal expressions represent an absolute or relative reference to a specific date (e.g. "*on Saturday night*", "*March 14, 2013*", "*10:24 a.m. Friday*", "*today*", "*this morning*", "*last week Friday*" and "*two days ago*").

Filtering Metadata Sentences

Some articles of ECB+ corpus contain *metadata sentences* inside the text. *Metadata sentences* include URL, header, as well as publication and update statements (Table 5-3). Such information is usually located in the beginning and, sometimes, at the end of the text.

Table 5-3. Example of metadata sentences in ECB+ articles

http://www.nydailynews.com/new-york/7-nypd-bullets-killed-teen-kimani-gray
PUBLISHED: WEDNESDAY, MARCH 13, 2013, 2:29 PM
UPDATED: THURSDAY, MARCH 14, 2013, 9:31 AM
Violent protests in Europe borne out of deeper frustrations

ECB+ corpus includes annotation of these sentences as well. There is a significant amount of time annotations for sentences that are just publication statements (e.g. "*Published: Wednesday, March 13, 2013, 2:29 PM*"). Such sentences do not represent real-world events and should not be used for event extraction. *Metadata sentences* and time annotations that refer to them has to be filtered from the corpus. The event extraction algorithm has mechanisms for filtering *metadata sentences*. However, the aim of the evaluation corpus is to focus evaluation purely on event extraction. Therefore, the filtering of *metadata sentences* was done manually with the aim of removing any possible evaluation bias. The remaining number of exact non-ambiguous day-level temporal expressions is 942 expressions in the whole corpus.

Reconstructing Publication Dates

The algorithm for construction of a global repository of events requires an article to have a publication date. Unfortunately, the ECB+ corpus does not provide publication dates for articles in a separate field, but it is available within *metadata sentences*.

Publication date of many articles was extracted from *metadata sentences*, but not for all. For other articles it was found by searching the article in the Internet or was guessed on the basis of article's temporal expressions. The reconstructed publication dates were verified by checking that coreferent temporal annotations across articles still refer to the same date. As the result of this verification all available coreferent temporal annotations were correctly resolved. The information about the link and header was found in the same way.

Resolving Temporal Expressions

Each temporal expression was manually resolved using article's publication date as a reference date. The purpose of manual resolution of temporal expressions is to have an evaluation source to estimate the accuracy of temporal tagging.

Each of the resolved temporal expressions refers to one of 181 distinct dates. The resolved date for particular temporal expression can be in the past or future relative to the publication date or be the same as publication date. Temporal expressions that are marked as coreferent always point to the same date.

Reconstructing Text of Articles

Event extraction algorithm takes as input unstructured free text of a news article. However, the ECB+ corpus is distributed using annotated XML files. Each XML file contains tokenized text of a news article and corresponding event annotations. Punctuation characters are included as separate tokens. For example, an expression "*face-to-face*" is annotated using five tokens. The provided annotation lacks any description of how to reconstruct the original text. Nevertheless, with the aim of a pure evaluation of the extraction algorithm, it is required to reconstruct the original text of an article to mimic the required input.

Text reconstruction is made using a set of rules that cover basic punctuation grammar. These rules look at the neighborhood of tokens while correctly inserting whitespaces between them. As event extraction algorithm is mainly focused on sentences with temporal expressions, these sentences were additionally manually investigated to make sure they are correctly reconstructed.

Cluster Forming

Construction of global repository of events takes as input topically related clusters of news articles. ECB+ corpus describes 43 topics capturing descriptions of double event instances per topic. Because these double event instances happened on different dates due to different articles' publication dates, each topic is represented by two separate news clusters. The whole ECB+ corpus therefore contains 86 news clusters. Articles inside the news clusters are ordered by publication date and time.

Assignment of News Categories

Grouping methods using cosine similarity utilize news category of articles to give more weight to the news-category-specific words. ECB+ annotation does not provide such information, so news categories were manually assigned to each ECB+ topic on the basis of articles content. Table 5-4 summarizes the distribution of articles per each news category.

Table 5-4. Distribution of articles per news categories in ECB+ corpus

News category	Number of articles	Number of clusters
BUSINESS	45	4
ENTERTEINMENT	105	8
POPULAR	207	18
SPORTS	182	16
SPOTLIGHT	117	12
TECHNOLOGY	99	8
US NEWS	84	6
WORLD NEWS	143	14

Annotation of Events

After having resolved temporal expressions, the task of the human annotator was to annotate unique events for each date. This annotation marks *global events* within the ECB+ corpus. There can be several events on the same date. However, each event instance can happen only on one date. For each event, human annotator also marks sentences that speak about this event. Coreferent temporal expressions, due to their event-centric nature, have all been marked with the same event instance.

Annotation resulted in 250 unique events within the corpus. The discovered disadvantage of the created evaluation corpus it that there are many dates having only one event instance happening on that date. This reduces the representativeness of the evaluation corpus with regards to the large-volume streams of news.

5.1.2 Real-World Data

Algorithm for construction of a global repository of events has also been evaluated using real-world streams of news. Real-world evaluation corpus consists of 5 clusters with 1434 news articles published over 4 days. The evaluation corpus in total represents 407 *global events*. Pure manual annotation of the corpus is very time consuming process and requires large human involvement. Therefore, the annotation process included the usage of the developed event extraction algorithm to speed up the annotation and potentially minimize human errors.

The annotation process was performed with the usage of the developed temporal tagger. Temporal tagger recognizes exact non-ambiguous temporal expressions and automatically resolves them using article's publication date as the reference date. The task of the human annotator is to (1) manually verify extracted temporal expressions and remove invalid ones, and (2) manually verify and correct resolved dates.

Having sentences with recognized temporal expressions as input, the annotation process extracts *event mentions* and group them into *cluster events* using group-average agglomerative clustering with 70% minimum cosine similarity. The resulting *cluster events* represent highly similar *event mentions*. The task of the human annotator is to manually investigate each extracted *cluster event* and remove from them non-related *event mentions*. Duplicated *cluster events* that actually speak about the same event are manually grouped into the same *cluster event*.

After all *cluster events* for all input news cluster are manually verified, the annotation process groups highly similar *cluster events* into *global events*. This also uses agglomerative clustering with 70% minimum cosine similarity. The task of the human annotator is to manually verify each extracted *global event* and merge duplicate *global events* into one event instance if such were detected. The annotation process tracks the origin of *global events* to sentences and annotates each sentence with mentioned events.

5.1.3 Comparison and Usage of Evaluation Corpuses

Event Coreference Bank Plus evaluation corpus was used through the whole implementation part of the thesis. It is useful corpus to evaluate and tune the recognition and resolution of temporal expressions, as well as extraction of *event mentions*. It allowed making initial comparison of different grouping methods on a small scale data. On the other hand, real-world evaluation corpus is a representative of a real-world news streams making it much more useful for final evaluation.

Based on our observations, construction of a global repository of events from real-world evaluation corpus possesses harder settings due to bigger variety of textual representations. It is an important factor because if a corpus is not a representative of the sampled language population, one cannot be sure that the results of experiments obtained on it can be generalized onto the intended language population [54]. Table 5-5 compares two evaluation corpuses.

Table 5-5. Comparison of evaluation corpuses

Aspect	ECB+ Corpus	Real-World Corpus
Size of clusters, articles	4…21	141…807
Size of articles, sentences	3…52	7…182
Participants per article	Few	Many
Annotation	Tool-based	Manual
Text quality	High	Noisy

Real-world news clusters are much bigger than clusters in ECB+ corpus. The biggest annotated news cluster has 807 articles, but, in general, news clusters with up to 2000 articles are observed. Real-world news articles are also much longer varying from 7 sentences to 182 sentences. In general, they contain bigger number of participants and event details. The combination of these aspects makes event extraction from such news clusters much more difficult even for human annotator. In addition, text of real-word evaluation corpus is considered as noisy. It can contain unrelated materials in the form of advertisements or recommended articles. However, the real problem of a noisy text is the presence of misspelled or mistakenly-merged words. Such words are useless when computing similarity of *event mentions*.

Representativeness of the annotated real-world news clusters can be characterized by the amount of unique events they contain per day. Figure 5-1 shows the distribution of unique events and *event mentions* for a small cluster tracking a discovered 1-week-old murder case. Figure 5-2 shows the same information for the larger cluster about a presentation of Apple Smart Watch. The crossed area denoted the dates when the articles of a cluster were published.

As observed from figures, a huge number of events happened during the publication dates of articles. Larger clusters tend to have bigger number of extracted dates resulting in a long tail distribution. Dates that are far apart from the publication dates usually have only a few events without a significant amount of *event mentions*. However, because the evaluation corpus contains several clusters published over the same period of time, each date tend to be represented by at least two events. There is no discovered correlation between the number of *event mentions* and the number of events on the same date. Annotated real-world news clusters have a good representativeness of the real-world news streams having several unique events for the vast majority of dates and, thus, can be considered as a suitable evaluation corpus.

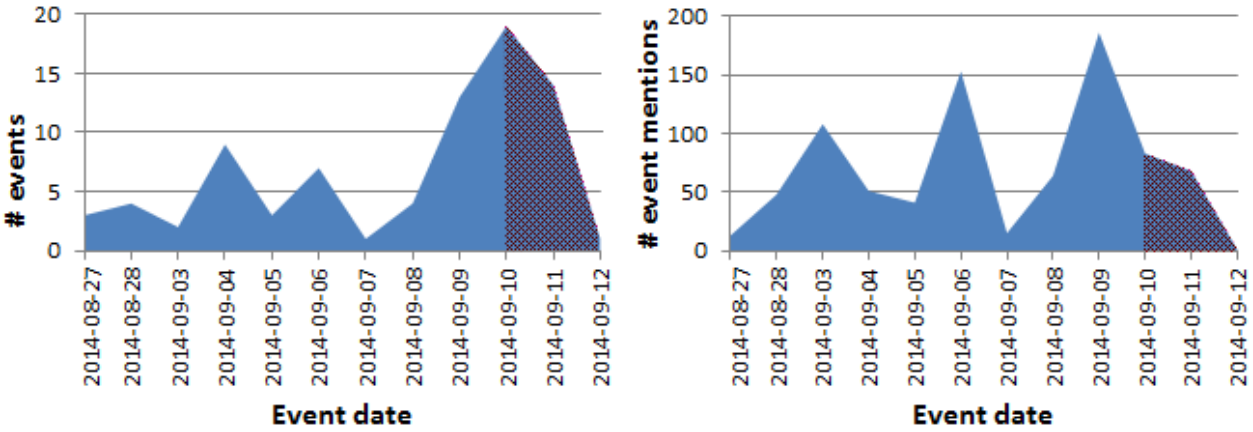

Figure 5-1. Distribution of events and event mentions in annotated small cluster with 141 articles tracking a discovered 1-week-old murder case

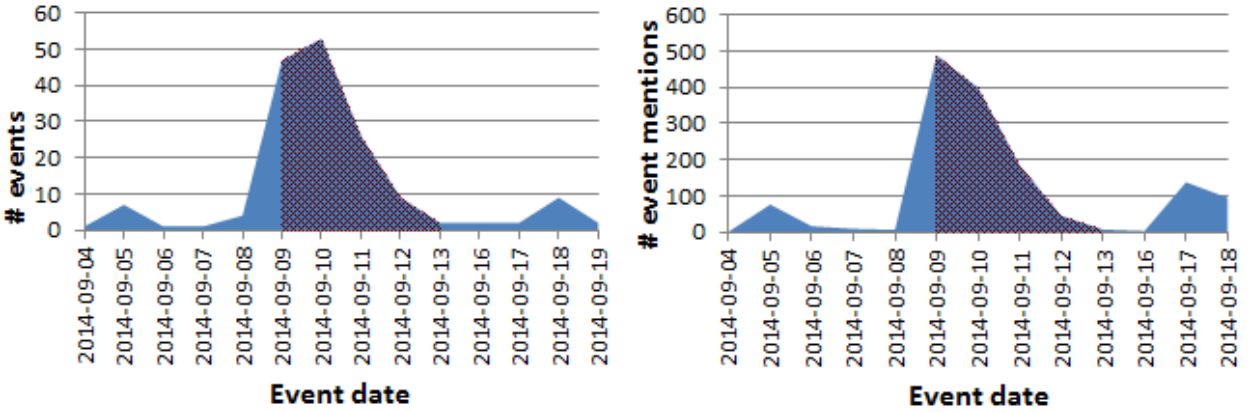

Figure 5-2. Distribution of events and event mentions in annotated large cluster with 807 articles about presentation of Apple Smart Watch

5.2 Evaluation Metrics

This section describes and formalizes three metrics used to evaluate event extraction process: precision and recall (Subsection 5.2.1) and purity (Subsection 5.2.2).

5.2.1 Precision and Recall

Precision and *recall* have been used commonly to measure the effectiveness of information retrieval and information extraction systems [39, 93]. Given a user's information need represented by a query, the set of returned results is either *relevant* or *irrelevant* with respect to this information need. *Recall* represents the proportion of retrieved relevant results among all relevant results in the corpus, while *precision* represents the proportion of retrieved results that are relevant. The extent to which a system produces all the appropriate output is defined by *recall*, and only appropriate output is defined by *precision*. Therefore, *recall* and *precision* can be seen as a measure of completeness and correctness, respectively [39].

In the context of this work, the extracted events represent an information need. An event contains and is represented by a set of sentences mentioning this event. The set of relevant sentences for each event is defined according to an annotated evaluation corpus, representing ground truth or a "gold-standard". A retrieved sentence is said to be irrelevant if it does not align with an event or if it has been assigned as mentioning an incorrect event. The *recall* and *precision* metrics for the evaluation of a single event can be defined formally as follows.

Let S be a set of all sentences in a corpus, $Rel \subseteq S$ be the subset of relevant sentences mentioning an event e, and $Res \subseteq S$ be the subset of retrieved sentences for e.

- *Precision* is fraction of relevant sentences that are retrieved for an event. Precision 1.0 means that all retrieved sentences are relevant and represent actual *event mentions*:

$$precision(e) = \frac{|Res \cap Rel|}{|Res|}$$

- *Recall* is fraction of retrieved sentences that are relevant for an event. Recall 1.0 means that all relevant sentences in a corpus are retrieved:

$$recall(e) = \frac{|Res \cap Rel|}{|Rel|}$$

Each extracted event can be considered as a cluster. This turns the task into an evaluation of a classification process. In classification, the *precision* and *recall* are computed using contingency tables with notation of true positives, false positives and false negatives. True positives describe the relevant retrieved sentences and false positives describe the irrelevant retrieved sentences. False negatives describe the relevant sentences which are not discovered or retrieved. Thus, the computation of *precision* and *recall* values is defined as follows:

$$precision(e) = \frac{true\ positives}{true\ positives + false\ positives}$$

$$recall(e) = \frac{true\ positives}{true\ positives + false\ negatives}$$

The above definitions represent calculation of *precision* and *recall* values for individual events. The resulting *precision* and *recall* for the extraction of all the events in the corpus is represented by an average value. Events are treated equally. If an event is not retrieved by the system, but annotated in the evaluation corpus, the *precision* and *recall* for such events results in 0.0 values. The calculation of *precision* and *recall* values for both *cluster event* and *global event* is made by the same formulas. The only difference is in the scope of the evaluation corpus.

5.2.2 Purity

Quality of clusters can be represented by their *purity*. *Purity* is a simple and transparent evaluation measure that penalizes both false positives and false negatives [17]. Event can be considered as a cluster of sentences grouped together. The biggest subgroup of sentences that refer the same actual event determines the event this cluster represents. For example, consider two clusters with 10 members in total (Figure 5-3). The majority class and number of members of the majority class for these clusters are: (x, 4) and (y, 3), respectively. Therefore, cluster A represents an event "x" and cluster B represents an event "y".

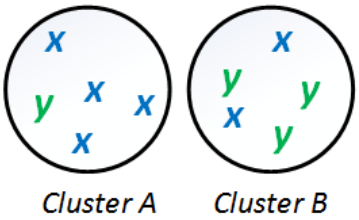

Figure 5-3. Purity as evaluation metric of cluster quality

To compute purity, each cluster is assigned to the majority class within the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned members divided by the total number of members [17]. Formally:

$$purity(\Omega, C) = \frac{1}{N}\sum_{k} max\left|\omega_k \cap c_j\right|$$

where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$ is a set of clusters and $C = \{c_1, c_2, \ldots, c_j\}$ is the set of classes. If all members refer to the correct events then the purity is 1.0. For the example depicted in Figure 5-3, the *purity* value is computed as $(1/10) * (4 + 3) = 0.7$

5.3 Evaluation Results

This section describes the evaluation of the proposed algorithm, in particular of such steps as temporal tagging (Subsection 5.3.1), as well as extraction of *cluster events* and *global events* (Subsection 5.3.2). An analysis in performed on the evaluation set in order to determine *precision*, *recall* and *purity* for event extraction, and estimate the accuracy of temporal tagging.

Event extraction process is evaluated using two annotated evaluation corpuses (ECB+ and real-world corpus) and four grouping methods: 1) the baseline grouping method using proper nouns ("*Proper Nouns*"), 2) grouping method using word alignment ("*Word Alignment*"), 3) grouping method using cosine similarity with group-average agglomerative clustering ("*Cosine-Clustering*"), and 4) grouping method using cosine similarity with connected components ("*Cosine-Components*").

5.3.1 Evaluation of Temporal Tagging

This subsection describes evaluation of the temporal tagging using annotated *Event Coreference Bank Plus* (ECB+) corpus. Correct extraction of events is the basis for the construction of a global repository of events. This, in turn, depends on the (1) correct recognition of exact non-ambiguous day-level temporal expressions within the text, and (2) correct resolution of detected temporal expressions to actual timestamps.

Temporal tagger was evaluated using a collection of all sentences from the ECB+ corpus. For each input sentence the temporal tagger recognizes temporal expressions and resolves them to specific dates. The output is then compared with manually annotated answers. The output is marked as erroneous if temporal tagger recognizes more non-ambiguous temporal expressions then there exist in reality, or if temporal expression is resolved to a wrong date. The initial evaluation results show 95.3% correct temporal tagging. The usage of the tense of the predicate to adjust the resolved date has a positive effect and improves the accuracy to 98.2%.

The fraction of incorrectly recognized temporal expressions constitutes only a small part of errors. This can be the result of tuning an algorithm to the evaluation corpus. Textual diversity of the real-world news streams can contain examples of ambiguous temporal expressions that are not covered by the temporal tagger.

5.3.2 Evaluation of Event Extraction

Basing on the event extraction algorithm, *global events* are composed of *cluster events*. Therefore, the quality of the resulting *global events* depends on the quality of the extracted *cluster events*. Thus, evaluation using each of the available evaluation corpuses starts by determining the best grouping method for the extraction of *cluster events*. Then the task is to determine the best performing combination of cluster and global events grouping methods. Evaluation corpuses and their characteristics are described in Section 5.1.

Evaluation using ECB+ Corpus

The ECB+ corpus was used as the first evaluation corpus during the implementation part of the thesis. It contains 250 unique events within 984 articles in 86 news clusters and allowed making initial comparison of different grouping methods on a small scale data with low textual diversity. Table 5-6 shows the best obtained evaluation results for the extraction of *cluster events* using different grouping methods. The best obtained results for particular evaluation metric are marked in bold.

Table 5-6. Best evaluation results for extraction of cluster events using ECB+ corpus

Grouping method	Precision	Recall	Purity	Events Count
Proper Nouns	**0.931**	0.783	**0.961**	294
Word Alignment	0.919	**0.874**	0.954	**255**
Cosine-Clustering	0.918	**0.876**	0.954	**254**
Cosine-Components	0.911	**0.879**	0.953	**249**

First thing to notice is that *word alignment* and *cosine similarity* grouping methods provide very similar evaluation results. Also *word alignment* annotation appeared to be very slow computing about 15 sentence pairs per second (on the mid-end laptop with *12 GB* of *RAM* and *1.9 GHz CPU*). The slow process of alignment annotation makes it practically inapplicable for running on the real-word news streams.

Word alignment and *cosine similarity* grouping methods provide the best *recall* values extracting the count of events that is very close to the required one. Surprisingly, the highest *precision* and *purity* values were obtained by the simplest grouping method using just proper nouns. The disadvantage of "*proper nouns*" method is that it has extracted much bigger number of events resulting in the lowest *recall* and duplicated event instances.

In general, all grouping methods showed very high evaluation results. This is explained by the presence of many dates having only one event happening on that date. Table 5-7 shows the best obtained evaluation results for the extraction of *global events* using different combinations of cluster and global events grouping methods.

Table 5-7. Best evaluation results for extraction of global events using ECB+ corpus

Grouping method for cluster events	Grouping method for global events	Precision	Recall	Purity	Events Count
Word Alignment Cosine-Clustering Cosine-Components	Word Alignment Cosine-Clustering Cosine-Components	0.869	**0.816**	0.943	**251**
Proper Nouns	Cosine-Clustering	**0.890**	0.758	**0.947**	273
Cosine-Clustering	Proper Nouns	0.810	0.773	0.876	240

In general, extraction of *global events* shows reduced values for all evaluation metrics. Different combinations of *word alignment* and *cosine similarity* grouping methods show very similar evaluation results having on average 86.9% *precision*, 81.6% *recall*, 94.3% *purity*, and producing almost perfect number of extracted events. Combinations of *proper nouns* and other grouping methods do not provide satisfactory results. For example, (*proper nouns*, *cosine-clustering*) provides the highest *precision* and *purity*, but results in duplicated event instances (indicated by 273 extracted events). Usage of *proper nouns* for grouping of *global events* does not provide any remarkable evaluation results as well.

The developed algorithm provides high evaluation results due to the reduced representativeness of the ECB+ corpus with regards to the large-volume streams of news. There is a need for the evaluation on the real-word data.

Evaluation using Real-World Data

Real-world evaluation corpus contains 407 unique events within 1434 articles in 5 news clusters published over 4 days. The representativeness of the corpus is detailly described in Subsection 5.1.3. Table 5-8 shows the best obtained evaluation results for the extraction of *cluster events* using different grouping methods. The best obtained results for particular evaluation metric are marked in bold.

Table 5-8. Best evaluation results for extraction of cluster events using real-world data

Grouping method	Precision	Recall	Purity	Events Count
Proper Nouns	0.783	0.233	0.505	745
Word Alignment	*not practically applicable*			
Cosine-Clustering	**0.888**	0.638	**0.831**	574
Cosine-Components	0.802	**0.651**	0.648	**465**

The baseline grouping method using just *proper nouns* shows much less impressive evaluation results when applied on more representative evaluation corpus. *Proper nouns* method provides good *precision*, but absolutely not acceptable *purity* and *recall*. The best results are obtained using the minimum of 2 common proper nouns. Basing on the evaluation using ECB+ corpus, *word alignment* grouping method is practically inapplicable on real-world news streams. Therefore, this grouping method is excluded from the evaluation.

The best evaluation results using both *cosine similarity* methods are obtained using 60% minimum similarity. *Cosine-components* produce the closest count of events to the required one, but extracted events are mixed events due to low *purity*. Agglomerative clustering ("*cosine-clustering*") provides the best *precision* and *purity*. The high number of extracted events is explained by the big diversity of textual representations referring the same event, which possesses the difficulty for similarity calculation. Table 5-9 shows the best obtained evaluation results for the extraction of *global events* using different combinations of grouping methods.

Table 5-9. Best evaluation results for extraction of global events using real-world data

Grouping method for cluster events	Grouping method for global events	Precision	Recall	Purity	Events Count
Cosine-Clustering	Cosine-Components	**0.887**	0.605	**0.838**	569
Cosine-Components	Cosine-Clustering	0.781	**0.621**	0.626	**461**
Cosine-Clustering	Cosine-Clustering	0.825	**0.625**	0.748	489

Cosine-components grouping method, when applied after *cosine-clustering*, does not provide any extraction gain. The number of extracted events is reduced by 5 instances slightly lowering evaluation metrics. The method provides the highest *precision* and *purity*. However, the extracted *global events* contain big amount of duplicated event instances. Presented results are the best results obtained using the (60%, 70%) minimum similarity for the (*cosine-clustering*, *cosine-components*) methods, respectively.

Cosine-clustering grouping method, when applied after *cosine-components*, also does not provide any extraction gain. The number of extracted events is reduced by 4 instances providing the lowest *precision* and *purity*. This combination of grouping methods provide the most mixed extracted *global events*, which is bigger unwanted situation comparing to duplicated event instances. Presented results are the best results obtained using the (60%, 60%) minimum similarity for the (*cosine-components*, *cosine-clustering*) methods, respectively.

Combination of (*cosine-clustering*, *cosine-clustering*) methods provides high *precision* (82.5%), high *purity* (74.8%) and the best *recall* (62.5%). The extracted *global events* represent the tradeoff between duplicated event instances and mixed events. This combination is considered as providing the best extraction results. The extracted events are pure and precise enough for the reader to recognize the main information content. Presented results were obtained using (60%, 50%) minimum similarity, respectively. The fact that grouping of *cluster events* into *global events* uses smaller minimum similarity is the subject of discussion that follows.

Basing on observations, some events are mentioned by a lot of news articles, while some events are mentioned only by a few. There is an interest in evaluating how an algorithm extracts events with different popularity. The *popularity* of an event is defined by the number of *event mentions* describing this event. Popularity implies bigger number of *event mentions*, which also potentially results in bigger textual diversity. Figure 5-4 shows *precision* and *purity* when extracting *cluster events* using real-world evaluation corpus and *cosine-clustering* with 60% minimum similarity. The x-axis defines the minimum popularity of *cluster events* being considered during an evaluation.

Increase of event popularity results in relatively stable *precision* and big loss of *recall*. The loss of *recall* is explained by the increasing textual diversity with the increase of event popularity. The same event can be represents using different words and different level of details. The initial significant loss of *precision* and *recall* (when moving from 0 to 5 on x-axis) is explained by the presence of singleton events having a small number of *event mentions*. Singleton events posse low textual diversity and, thus, have high *precision* and *recall*, boosting the overall evaluation results.

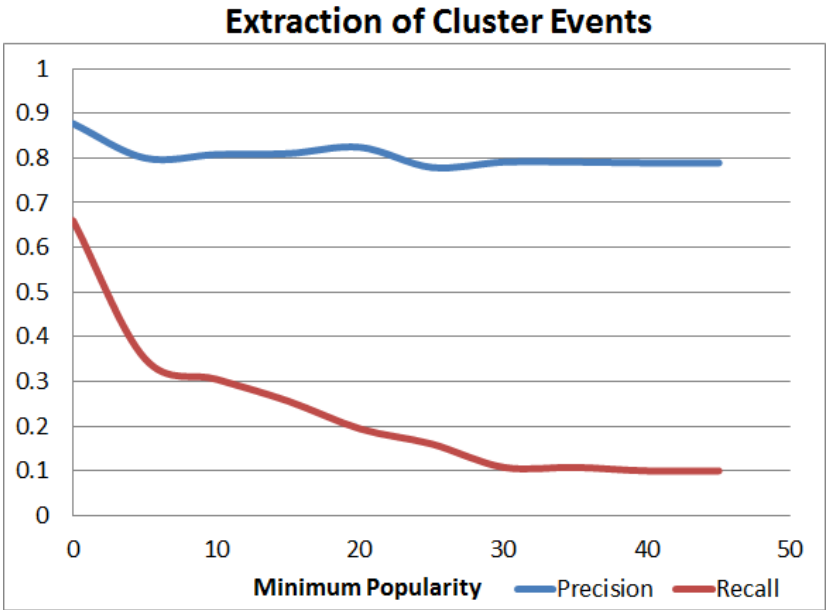

Figure 5-4. Evaluation of cluster events extraction based on popularity

The low *recall* indicates that event extraction results in duplicated event instances. However, these duplicated event instances are then merged during the step of grouping *cluster events* into *global events* (Figure 5-5). The resulting *global events* have high *precision* and increased *recall*.

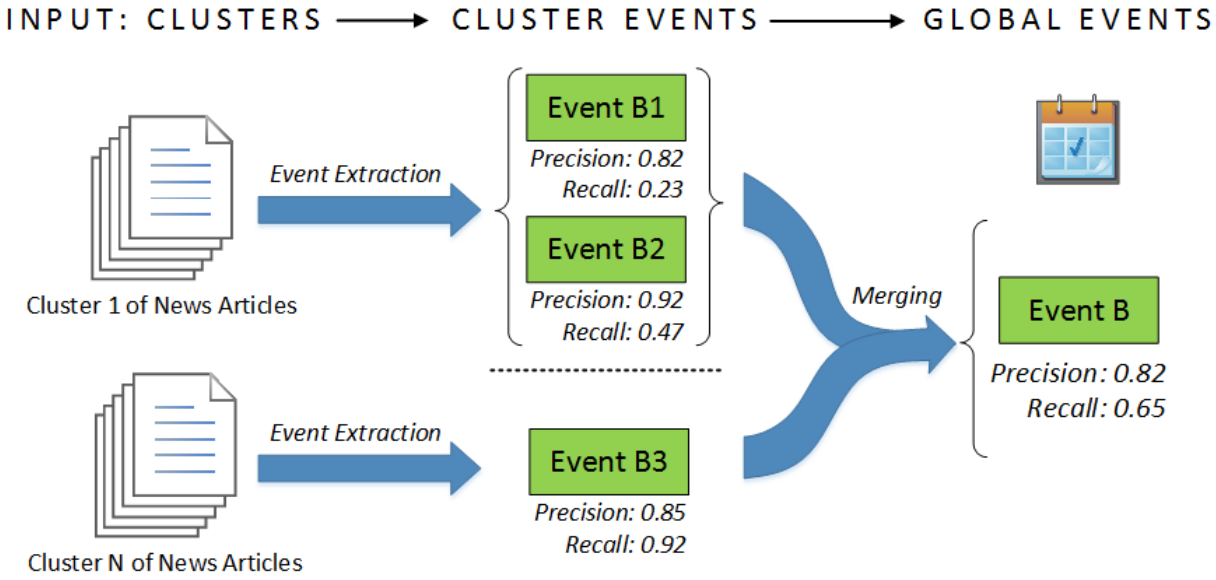

Figure 5-5. Merging of duplicated cluster events using (cosine-clustering, cosine-clustering) combination with (60%, 50%) minimum cosine similarity, respectively

In order to explain the fact that extraction of *global events* uses smaller minimum similarity than extraction of *cluster events*, it is necessary to review the extraction of *cluster events* as follows. Extraction of *cluster events* considers two tasks: (1) grouping *event mentions* speaking about the same actual event, and (2) not grouping *event mentions* speaking about different events. The first task is achieved by using synonyms in order to deal with different

textual representations of the same event. The second task can be achieved by setting high minimum grouping similarity. On the real-word evaluation corpus the best tradeoff between these two tasks is achieved by setting minimum similarity to 60%. This results in high precision values, but also in duplicated event instances. These duplicated event instances are represented by green-colored *cluster events B1* and *B2* on Figure 5-5.

On the next step of grouping *cluster events* into *global events*, the algorithm does not utilize synonyms. *Cluster events* already have much bigger bag of different words speaking about the same actual event and have increased weights for the repeated words. This allows grouping duplicated *cluster events*, having the same major participants and speaking about the same event, using smaller similarity score. This also relies on the fact that the same participants are mainly involved only in the one event during the same day.

5.4 Sources of Errors Analysis

This section describes sources of errors resulting in a loss of recall (Subsection 5.4.1) or precision (Subsection 5.4.2). All described errors were discovered after a manual investigation of extracted events and comparison with the annotated evaluation corpus.

5.4.1 Source of Errors for Recall

The loss of recall is mostly affected by different textual representations used to refer the same event. For some sentences, the event extraction failed to extract described events, but basing on observations, these events were captured from other sentences. All sources of errors are categorized into several classes basing on their nature. Table 5-10 lists sources of errors and relative amount of events affected by each error.

Table 5-10. Source of errors for recall

Source of errors	Amount of errors, %
Textual diversity	63.7
Coreference resolution required	21.4
Insufficient semantic roles	10.2
Passive voice	4.7

Textual Diversity

The main reason for loss of recall is low similarity textual representations of the same event. This class of errors represents more than a half of all detected errors (63.7%) making a significant influence on the evaluation results. Consider the following sentences that were

annotated as describing the same event of "*burning bank on Wednesday*", but were not grouped together by the event extraction algorithm:

- *Bank in Greece blew up from a fire-bomb attack on Wednesday killing at least three people as police fought pitched battles with striking protestors furious at brutal budget cuts designed to avert national bankruptcy.*
- *Athens bank went up in flames on Wednesday evening as tens of thousands of Greeks took to the streets to protest harsh spending cuts aimed at saving the country from bankruptcy.*

The above sentences are easy cases for the human annotator to recognize the same event. However, it is not true for the event extraction algorithm. In particular, in the given sentences such words as "*protestors*" and "*to protest*" do not provide any event grouping gain. That is because the "*protestors*" represents a noun, while "*to protest*" represents a verb. Therefore, these words, in fact, are reducing cosine similarity. Usage of synonyms sets for nouns and verbs also did not give any event grouping advantage.

Coreference Resolution Required

Coreference resolution is the task of finding all expressions that refer the same entity in a text. It attempts to create a chain that goes back and forth and provides what phrases refer to which others and which one is the representative one [55, 80]. Coreference may be named (*General Electric* and *GE*), pronominal ("*he*", "*they*", "*it*"), nominal ("*the company*"), demonstrative ("*this*", "*that*"), or more general terms for a specific entity.

Coreference resolution is not included as a part of event extraction algorithm. It is a separate topic in computer linguistics which may significantly increase the processing time of event extraction and potentially add errors to the result [89]. In the context of this work, there is no need to apply coreference resolution on the whole text of an article. While application of a coreference resolution on specific sentences and their neighborhood only, may not provide satisfactory results. Unhandled coreference resolution causes event extraction algorithm to lose a significant amount (21.4%) of recall value. To illustrate this source of errors, consider the following sentences:

- *John Jenkin, 23, of Newton Street, Millom was arrested Saturday morning after police found two women dead in his house.*
- *Police have this morning arrested him and he will appear before Furness District Magistrates.*

From the second sentence the algorithm extracts a short *event mention* "*Police have this morning arrested him*", while the *event mention* extracted from the first sentence is the same as the sentence itself. The extracted *event mentions* do not have a lot of common event details and, therefore, they are not grouped as representatives of the same event. The coreference resolution should replace the pronoun *him* in the second sentence with the person name *John Jenkin* and potentially identify his attributes such as age ("*23*") and location "*of Newton Street, Millom*". This aspect is considered as one of the limitations of the event extraction algorithm and proposed being improved in the future work.

Insufficient Semantic Roles

This source of errors causes 10.2% loss of recall. News articles are often written in syntactically rich way with overwhelming reliance on complex structures. Text of a single sentence may encompass many event details, statements and facts. Such sentences result in impossibility for the syntactic analyzer to construct corresponding dependency tree or to recognize presence of a temporal argument. To illustrate this source of errors, consider the following sentence:

- *In record time, Microsoft released today an out of band patch for a significant security flaw in Internet Explorer burst onto the scene 8 days ago.*

The syntactic analysis returns no dependency tree for an event "*security flaw in Internet Explorer burst onto the scene 8 days ago*". To illustrate the case with an empty temporal argument, consider the following sentence:

- *The bodies of Alice McMeekin, 58, and Kathryn Jenkin, 20, were found at a terraced house in Newton Street, Millom, on Saturday.*

The extracted *event mention* is "*The bodies of Alice McMeekin, 58, and Kathryn Jenkin, 20*" (*A1*), "*found*" (*P*), "*at a terraced house in Newton Street, Millom*" (*AM-LOC*). The syntactic analyzer does not recognize "*on Saturday*" as a temporal argument.

In general, insufficient semantic roles are caused by the sentence style and complexity, absence or excess of punctuation, and sentence segmentation errors. This source of errors represents a limitation of the used language-processing libraries. Such errors are considered as indirect errors because they are not caused by the event extraction algorithm.

Passive Voice

Event extraction relies on the predicate-argument structures from dependency trees. The passive voice sentences represent a more difficult case for syntactic analysis and more often result in incorrect dependency structures. Consider an example sentence:

- *"Doctor Who" has finally selected its 12th doctor: on Sunday Peter Capaldi is officially set to replace exiting star Matt Smith as the TARDIS leader.*

Syntactic analysis returns such dependency tree, among others, as "*Peter Capaldi",* "*set*" (*P*), "*to replace exiting star Matt Smith as the TARDIS leader*". However, the recognized verb predicate "*set"* does not represent an action performed by *Peter Capaldi.* Another kind or errors is an empty dependency tree as the result of syntactic analysis of a passive voice sentences. However, such situations are rare and contribute only to 4.7% of recall loss.

5.4.2 Source of Errors for Precision

Precision represents the correctness of extracted events [39]. Low precision indicates the presence of multiple different events within the group of *event mentions* intended to speak about the same actual event. All sources of errors affecting precision are categorized into several classes basing on their nature. Table 5-11 lists sources of errors and relative amount of events affected by each error. The sources of errors can be overlapping which means that several errors can be found at the same event.

Table 5-11. Source of errors for precision

Source of errors	Amount of errors, %
Coreference resolution required	46.9
Ambiguous temporal expressions	22.1
Word sense disambiguation required	13.3
Input data error	9.4
Conditional statements	8.3

Evaluation procedure penalizes extraction of events which are not annotated in the evaluation corpus. If such extracted not-annotated events are grouped to other annotated events, evaluation considers them as incorrect event members. However, if these non-annotated events act as separate events on specific dates, evaluation procedure penalizes them with 0.0 precision.

Coreference Resolution Required

The biggest loss of precision (46.9%) is caused by grouping *event mentions* into one event while actual described events are different. In most of the cases such situations happen due to the lack of such event details as person and organization names. Coreference resolution can significantly improve both precision and recall, as it was described in the previous subsection. To illustrate this source of errors, consider the following sentences:

- *Jones and his five children, ages 8, 7, 6, 2 and 1, disappeared two weeks ago, and he was arrested on Friday when he was stopped at a DUI checkpoint.*
- *An arrest of San Diego Chargers wide receiver Vincent Jackson will not affect his playing status for Sunday's divisional playoff game as yesterday he was arrested for suspicion of DUI.*

The two extracted *event mentions* are, respectively:

- *He was arrested on Friday when he was stopped at DUI checkpoint.*
- *Yesterday he was arrested for suspicion of DUI.*

The given *event mentions* describe two different events, but their respective event phrases look very similar to each other. Therefore, they are wrongly grouped as two instances of the same actual event. The coreference resolution should replace the pronoun "*he"* in the first sentence with person name "*Jones"*, and pronoun "*he*" in the second sentence with another person name "*Vincent Jackson"* or with the phrase "*San Diego Chargers wide receiver Vincent Jackson*". In general, having more event details is beneficial for grouping methods. The bigger amount of event details is expected to improve event precision.

Ambiguous Temporal Expressions

Extraction of ambiguous temporal expressions contributes to 22.1% of precision loss. Usage of ambiguous temporal expressions results in events that should not be extracted. Such events later are grouped to other events or appear as a separate event instances. The extracted ambiguous temporal expressions are mostly represented by names of holidays, festivities or other occasions that are celebrated in different cultures. Some examples of ambiguous temporal expressions that were not rejected by the temporal tagger are:

- "*Gary Neville discussed all the talking points from Manchester United's 4-2 derby victory over Manchester City on Super Sunday*". In this sentence, *Sunday* is detected as the temporal expression where *Super Sunday* is actually a name of a TV show;

- "*On Christmas holidays more than 250 volunteers spent their time helping families of children with life-threatening illnesses at Give Kids the World Village, in Kissimmee*". In this sentence, *Christmas* is recognized as the day-level temporal expression while the event was happening during a period of 7 days;
- "*The city will hold parades and concerts on Labor Day weekends*". Similar to the previous example, *Labor Day* is recognized as the day-level temporal expression while the events are happening over the weekends;

The possible solution for such erroneous cases is to disable recognition of holidays and other festivals. This will not result in recall loss as events represented by these cases are covered by other sentences using regular temporal expressions.

Word Sense Disambiguation Required

Word sense disambiguation is the ability to identify the meaning of words in the context of text [84]. Human language is ambiguous, so that many words can be interpreted in multiple ways depending on the context in which they occur. Event extraction algorithm utilizes synonyms when computing *cosine similarity* of *event mentions*. The resulting problem is that algorithm extracts all synonyms for each ever possible meaning for all nouns and verbs of each *event mention*. The resulting enormous collection represents greater chance for *event mentions* to have common synonyms. Presence of common synonyms increases the cosine similarity of *event mentions* and causes them to be grouped together. However, the actual meanings of words in original sentences can be different. Word sense disambiguation is useful not only to reduce the number of synonyms, but also to disambiguate the meaning of exactly the same word in different sentences. Consider the following examples:

- *The boy leapt from the bank into the cold river water.*
- *The boy jumped of the city bank into the river.*
- *My husband won the huge bank.*
- *He operated a bank of switches.*

The word "*bank*" has different meaning in each of the above sentences. In particular, in the first sentence it denotes side of the river, whereas in the second – the building. Nevertheless, without considering the word meaning, the two sentences have a very high cosine similarity. On the evaluation corpus this source of errors is rare. However, for systems running continuously on bigger number of inputted new clusters the amount of errors can be more significant. This aspect is proposed being improved in the future work.

Input Data Error

This error is not part of an event extraction algorithm, but comes from the news clustering system. The error describes the incorrect content retrieved by news crawler. The news article can be marked with an incorrect publication date or contain incorrect temporal expressions. This results in extraction of the same event for several days. Case of incorrect temporal expressions is illustrated by the following sentences (sentences are published on the same day and describe the same actual event):

- *Electronics manufacturer Hewlett-Packard announced **on Tuesday** that it has signed a definitive agreement to acquire EYP Mission Critical Facilities.*
- *Hewlett-Packard said **on Monday** 12 November that it has agreed to buy data center consulting company EYP Mission Critical Facilities.*

Another case assigned to this source of errors is incorrect usage of explicit week specifiers ("*last*", "*this*", "*next*"). Consider the sentence "*Volcano erupted on **last** Monday*", which was published on Wednesday. The temporal tagger recognizes temporal expression "*on **last** Monday*" and resolves it to the Monday on the previous week (9 days ago). However, the correct date when an event happened is Monday on this week (2 days ago). Wrong publication date can also be assigned to crawled news article, causing temporal tagger to use a wrong reference point when resolving relative temporal expressions. However, such input data errors are relatively rare and contribute only to 9.4% of precision loss.

Conditional Statements

This source of errors describes extraction of unauthentic events resulted from verbs in conditional statements (e.g. "*if ... then ...*", "*would*"). Some examples of such sentences are:

- *Third of us, polls show, would vote for Mr. Ford if elections were held today.*
- *Doug Ford smiled when asked if the city mayor would drop out on Thursday.*
- *If Apple on Tuesday announced price to be 250$ then it would avoid bad reviews.*

Event extraction from such sentences results in the following events, respectively:

- "*elections were held today*"
- "*city major drop out on Thursday*"
- "*Apple on Tuesday announced price to be 250$*"

Conditional statements are not handled by the event extraction algorithm. This aspect is considered as one of the limitations of the developed system. The conditional statements are relatively rare and represent the smallest precision loss of 8.3%.

5.5 Repository Presentation

This section describes the presentation of a created global repository of events. Extracted events are presented as calendar entries in the simple WEB interface (Figure 5-6). The page displays the top 3 most popular events for each date and for each news category separately. The full list of events for the given date and selected news category is shown on another page when clicking the link "*see all events*". Each entry in the lists of events displays the event representative as described in Section 3.5.

While there are several errors, the majority of event representatives are informative. One of the presented errors is "*arrest of Vincent Jackson*", having "*2008 GMC Sierra*" as an event location, which, in fact, represents the model of a car. Although some other event representatives, like "*Country held Parliamentary elections in Turkmenistan*", have an unusual way of presenting information.

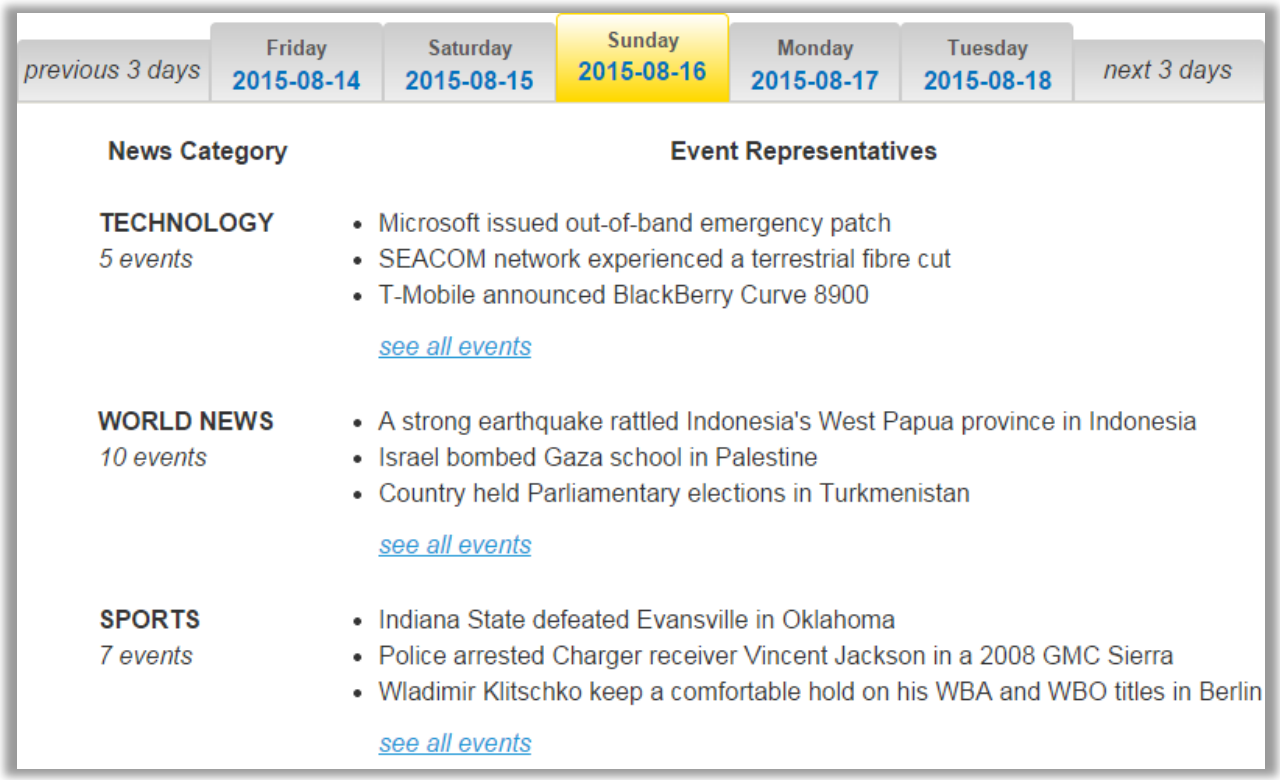

Figure 5-6. Repository presentation in the WEB interface

Summary

This chapter has presented an evaluation of the event extraction algorithm using four grouping methods and two annotated evaluation corpuses (ECB+ and real-world data). ECB+ corpus was adapted to represent the input required by the algorithm. Real-world corpus was manually annotated utilizing temporal tagging and clustering with high minimum cosine similarity to speed up the annotation process and reduce the amount of human errors. The evaluation corpuses were analyzed in order to determine their representativeness in relation to the intended language population.

Such metrics as *precision*, *recall* and *purity* were used to evaluate the extracted *cluster events* and *global events*. Accuracy of the temporal tagger was evaluated using the pre-annotated corpus of temporal expressions. Grouping method using *word alignment* is not practically applicable due to its long recessing time. Out of the remaining grouping methods, the group-average agglomerative clustering provided the best evaluation results when extracting *cluster events* as well as *global events*.

The proposed approach shows a strong potential for the construction of a global repository of events with 82.5% *precision*, 62.5% *recall* and 74.8% *purity* values. Reasons for loss of *recall* and *precision* were analyzed and illustrated. The next chapter gives an overall summarization of the thesis and lists the directions for future work.

Chapter 6

Conclusion and Future Work

This chapter makes an overview of the proposed algorithm for construction of a global repository of events and provides an overall conclusion. The basic idea is to indicate the main evaluation results, observations and limitations of the proposed approach and to list the directions for future work.

6.1 Conclusion

This thesis explores a way to build a global repository of events from exploiting topically related clusters of news articles. The proposed approach is novel in the sense that it utilizes the whole news clusters as single units instead of working with isolated articles alone. The algorithm operates on the sentence level and extracts an unrestricted set of events and all their possible event references using *OpenIE* techniques.

In the context of this work, time component of an event is a required component. Therefore, event extraction is limited to sentences that contain exact non-ambiguous day-level temporal expressions. The flow of the algorithm is to (1) detect and resolve temporal expressions, (2) extract *event mentions* from individual sentences by performing syntactic and semantic analysis, (3) group highly similar *event mentions* into *cluster events* by leveraging different representations of the same event within the same cluster, and (4) group highly similar *cluster events* into *global events*. The thesis describes four different grouping methods and evaluates the approach using two evaluation corpuses.

The proposed approach shows a strong potential for the construction of a global repository of events with 82.5% *precision*, 62.5% *recall* and 74.8% *purity*. It was observed that the same event can be referred by many different textual representations using different words and with different level of details. These low-similarity textual representations represent the biggest challenge discovered throughout the evaluation. This is the main reason for loss of recall, resulting in duplicated event instances. Absence of coreference resolution is another significant source of errors reducing both *precision* and *recall*.

In conclusion, the proposed approach shows promising results for event extraction. Hosting of extracted global repository of events can provide a valuable resource to gain a comprehensive overview of the world situation. The developed approach can serve as a resource for future event-linking efforts and can be further improved as it contains a lot of room for improvements.

6.2 Extraction of Event Mentions: Improved Parts

Extraction of *event mentions* in this thesis is based on the work "*Extracting a Repository of Events and Event References from News Clusters*" by Silvia Julinda [89] of the 2012-2014 IT4BI cohort. Although, the overall process of *event mentions* extraction looks very similar, it contains many improvements and differences in the information flow.

The new approach performs text preprocessing (dates normalization and text cleaning), improving recognition of temporal expressions and reducing the amount of insufficient semantic roles. This resulted in improved recall. The utilized definition of ambiguous temporal expressions is much stricter leading to improved precision.

The new approach calls temporal tagger only once per input sentence. However, previously it was called once per sentence to detect the presence of temporal expressions and, then, once per each temporal argument of the dependency tree. The new way of adjusting the resolved date does not use temporal tagger either, and allows more flexible adjustment of different temporal granularities. The text segmentation is currently done by the same pipeline as temporal tagging reducing the amount of used libraries. The new approach explicitly links detected predicate to the appropriate temporal expression, which removes the problematic case of multiple temporal arguments in the dependency tree.

Finally, the described approach was extended by grouping *event mentions* into *cluster events* and *cluster events* into *global events*. The two annotated corpuses allowed making more accurate evaluation of the approach, in comparison to the post-evaluation procedure performed previously.

6.3 Future Work

This section describes directions for future work. A number of reasons for precision and recall loss were identified and discussed. In order to improve the performance of the system these issues have to be addressed. One of the most promising future improvements is coreference resolution [55, 80], which can increase both precision and recall. Another increase of precision

can be obtained by improving the detection of ambiguous temporal expressions and rejection of conditional statements. In terms of recall, priority in future work includes exploring other resources for semantic role labelling to reduce the amount of insufficient semantic roles.

Research in direction of word sense disambiguation is another step of future work. There may be no need to disambiguate all words, but rather a substantial subset of them, that is, those conveying the real content of the sentence. This may be performed with respect to news categories or domain ontologies. The expected disambiguation style is, therefore, "disambiguate less, but disambiguate better" [84].

In this thesis, the group-average agglomerative clustering provided the best evaluation results. The best performing minimum similarity was identified by trying different similarity values. However, as different news clusters may differ in the number of described events and the level of details, it may be of interest to define and use the cluster-based minimum similarity. The future work includes research in cutting the dendrogram where the gap between two successive combination similarities is the largest (Figure 6-1). Expected result of such cutting is to reduce the overall amount of duplicated event instances.

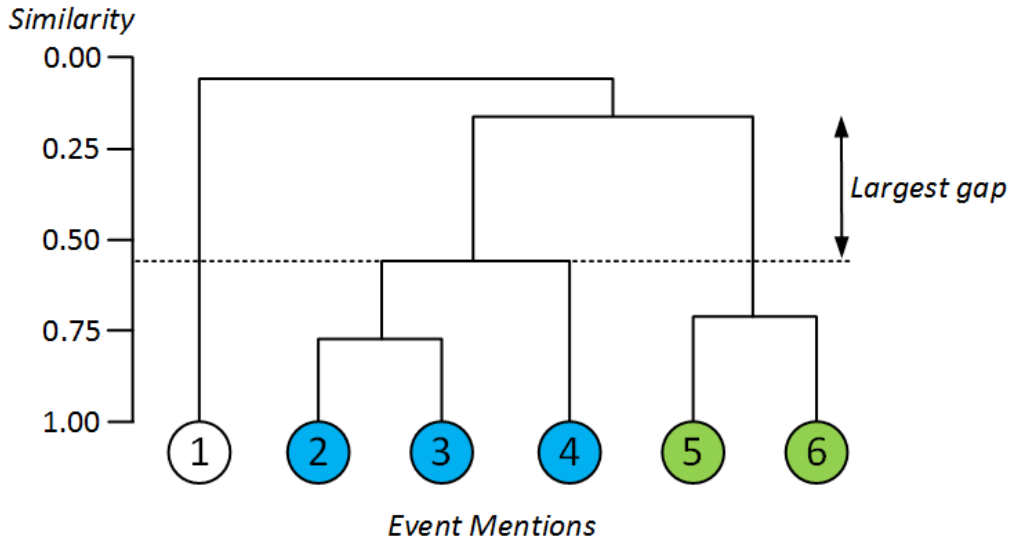

Figure 6-1. Cutting the dendrogram where the gap is the largest

The quality of text cleaning may be improved by utilizing *HTML markup* of news articles. Different grouping methods and event features can be added to improve the event grouping process. One potential idea is utilization of country where event happened [79]. Country can be identified on the basis of automatically recognized and disambiguated locations in the sentence or text. Other interesting event features [21, 32] can include event classes, head words, number and gender of words, classes of semantic arguments, coreferent predicates, and different feature combinations.

Usage of *hypernyms*, *hyponyms* and other *WordNet* relations in addition to *synonyms* can also be advantageous. One notable example is word "*police*" being not a synonym of word "*cops*". A simple way to measure the semantic similarity between two words is to treat *hyponym* or *hypernym* as undirected graph and measure the distance between them in *WordNet*. The shorter the path from one node to another, the more similar they are [77].

Finally, another important aspect for improvement is the performance. Because the natural language parsing task is CPU intensive, future work should improve the current approach by running the algorithm in parallel on cluster system.

Appendix A

Labels and Tags

This appendix provides descriptions of different labels and tags used in this thesis. *PropBank* semantic role labels are described in Section A.1. A table of all *CLEAR* dependency type labels and their descriptions are provided in Section A.2. Similarly, Section A.3 provides part-of-speech tags for English.

A.1 Semantic Role Labels

This section provides a table of used *PropBank* semantic role labels and their descriptions. For a detailed explanation of semantic role labels refer to [18].

Table A-1. List of PropBank semantic role labels

Label	Description
A0	Agent
A1	Patient, Theme
A2	Instrument, Benefactive, Attribute
AM-DIR	Direction
AM-LOC	Location
AM-MNR	Manner
AM-MOD	Modal
AM-PRD	Secondary predicate
AM-TMP	Temporal
C-V	Continuity of a verb

R-* and *C*-* labels represent referent and continuous arguments, respectively.

A.2 Dependency Type Labels

This section provides a table of all *CLEAR* dependency type labels and their descriptions. For a detailed explanation of dependency type labels refer to [51].

Table A-2. List of CLEAR dependency type labels

Label	Description	Label	Description
ACOMP	Adjectival complement	NEG	Negation modifier
ADVCL	Adverbial clause modifier	NMOD	Modifier of nominal
ADVMOD	Adverbial modifier	NN	Noun compound modifier
AGENT	Agent	NPADVMOD	Noun phrase as ADVMOD
AMOD	Adjectival modifier	NSUBJ	Nominal subject
APPOS	Appositional modifier	NSUBJPASS	Nominal subject (passive)
ATTR	Attribute	NUM	Numeric modifier
AUX	Auxiliary	NUMBER	Number compound modifier
AUXPASS	Auxiliary (passive)	OPRD	Object predicate
CC	Coordinating conjunction	PARATAXIS	Parataxis
CCOMP	Clausal complement	PARTMOD	Participial modifier
COMPLM	Complementizer	PCOMP	Complement of a preposition
CONJ	Conjunct	POBJ	Object of a preposition
CSUBJ	Clausal subject	POSS	Possession modifier
CSUBJPASS	Clausal subject (passive)	POSSESSIVE	Possessive modifier
DEP	Unclassified dependent	PRECONJ	Pre-correlative conjunction
DET	Determiner	PREDET	Predeterminer
DOBJ	Direct object	PREP	Prepositional modifier
EXPL	Expletive	PRT	Particle
INFMOD	Infinitival modifier	PUNCT	Punctuation
INTJ	Interjection	QUANTMOD	Quantifier phrase modifier
IOBJ	Indirect object	RCMOD	Relative clause modifier
MARK	Marker	ROOT	Root
META	Meta modifier	XCOMP	Open clausal complement

A.3 Part-of-speech Tags

This section provides a table of part-of-speech tags for English and their descriptions. For a detailed list of word-level and punctuation-like tags refer to [51].

Table A-3. List of part-of-speech tags for English

Tag	Description	Tag	Description
ADD	Email	POS	Possessive ending
AFX	Affix	PRP	Personal pronoun
CC	Coordinating conjunction	PRP$	Possessive pronoun
CD	Cardinal number	RB	Adverb
CODE	Code ID	RBR	Adverb, comparative
DT	Determiner	RBS	Adverb, superlative
EX	Existential there	RP	Particle
FW	Foreign word	TO	To
GW	Go with	UH	Interjection
IN	Preposition or subord. conjunction	VB	Verb, base form
JJ	Adjective	VBD	Verb, past tense
JJR	Adjective, comparative	VBG	Verb, gerund or present participle
JJS	Adjective, superlative	VBN	Verb, past participle
LS	List item marker	VBP	Verb, non-3rd person singl. present
MD	Modal	VBZ	Verb, 3rd person singular present
NN	Noun, singular or mass	WDT	*Wh*-determiner
NNS	Noun, plural	WP	*Wh*-pronoun
NNP	Proper noun, singular	WP$	Possessive *wh*-pronoun
NNPS	Proper noun, plural	WRB	*Wh*-adverb
PDT	Predeterminer	XX	Unknown

List of Figures

List of Tables

References

[1] Abdelati Hawwari, Jena D. Hwang, Aous Mansouri, and Martha Palmer. *Classification and Deterministic PropBank Annotation of Predicative Adjectives in Arabic*. In Workshop on Interoperable Semantic Annotation, page 44, 2011.

[2] Adam Kilgarriff. *Which words are particularly characteristic of a text? A survey of statistical approaches*. In Proceedings of the AISB Workshop on Language Engineering for Document Analysis and Recognition. Sussex, 04/1996, pages 33-40, 1996.

[3] Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. *Annotating Noun Argument Structure for NomBank*. In Proceedings of The International Conference on Language Resources and Evaluation (LREC), Volume 4, pages 803-806, 2004.

[4] Agata Cybulska and Piek Vossen. *Guidelines for ECB+ Annotation of Events and their Coreference*. Technical Report NWR-2014-1, Version Final, January 31, 2014.

[5] Agata Cybulska and Piek Vossen. *Using a sledgehammer to crack a nut? Lexical diversity and event coreference resolution.* In Proceedings of the 9th Language Resources and Evaluation Conference (LREC2014), Reykjavik, Iceland, 26-31 May, 2014.

[6] Akane Yakushiji, Yuka Tateisi, Yusuke Miyao, and Jun-ichi Tsujii. *Event extraction from biomedical papers using a full parser*. In Pacific Symposium on Biocomputing, Volume 6, pages 408-419, 2001.

[7] Alan Ritter, Oren Etzioni, Sam Clark, et al. *Open domain event extraction from Twitter*. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pages 1104-1112, 2012.

[8] Angel X. Chang and Christopher Manning. *SUTime: A library for recognizing and normalizing time expressions*. In Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC), pages 3735-3740, 2012.

[9] Anthony Fader, Stephen Soderland, and Oren Etzioni. *Identifying relations for open information extraction*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pages 1535-1545, USA, 2011.

[10] Balthasar Bickel. *Grammatical relations typology*. In The Oxford Handbook of Language Typology, Oxford: Oxford University Press, pages 399-444, 2010.

[11] Bretonnel Cohen, Karin Verspoor, Helen L. Johnson, Chris Roeder, Philip V. Ogren, William A. Baumgartner, Elizabeth White, Hannah Tipney, and Lawrence Hunter. *High-precision biological event extraction with a concept recognizer*. In Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task, Association for Computational Linguistics, pages 50-58, 2009.

[12] Chang-Shing Lee, Zhi-Wei Jian, and Lin-Kai Huang. *A fuzzy ontology and its application to news summarization*. In Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions, Volume 35, Issue 5, pages 859-880, 2005.

[13] Chen Bin, Su Jian, Pan Jialin Sinno, and Tan Chew-Lim. *A Unified Event Coreference Resolution by Integrating Multiple Resolvers*. In Proceedings of 5th International Joint Conference on Natural Language Processing, Chiang Mai, Thailand, November, 2011.

[14] Chinatsu Aone and Mila Ramos-Santacruz. *REES: A Large-Scale Relation And Event Extraction System*. In Proceedings of the 6th conference on Applied natural language processing, Association for Computational Linguistics, pages 76-83, 2000.

[15] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. *DBpedia - A crystallization point for the Web of Data*. In Web Semantics: Science, Services and Agents on the World Wide Web archive, Volume 7, Issue 3, pages 154-165, September, 2009.

[16] Christiane Fellbaum, *WordNet(s).* In Encyclopedia of Language and Linguistics, Second Edition, volume 13, Oxford: Elsevier, pages 665-670, 2006.

[17] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*, Volume 1. ISBN: 978-0521865715, Cambridge University Press, Cambridge, 2008.

[18] Claire Bonial, Jena Hwang, Julia Bonn, Kathryn Conger, Olga Babko-malaya, and Martha Palmer. *English PropBank Annotation Guidelines*. Center for Computational Language and Education Research, CU-Boulder, 2010.

[19] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. *The Berkeley FrameNet Project*. In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1, Association for Computational Linguistics, pages 86-90, 1998.

[20] Congle Zhang and Daniel S. Weld. *Harvesting Parallel News Streams to Generate Paraphrases of Event Relations*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2013.

[21] Cosmin Adrian Bejan and Marina del Rey. *Unsupervised event coreference resolution with rich linguistic features*. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1412-1422, Uppsala, Sweden, 2010.

[22] Cosmin Bejan and Sanda Harabagiu. *A Linguistic Resource for Discovering Event Structures and Resolving Event Coreference*. In Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC), Morocco, 2008.

[23] Daniel M. Cer, Marie-Catherine De Marneffe, Daniel Jurafsky, and Christopher D. Manning. *Parsing to Stanford dependencies: Trade-offs between speed and accuracy*. In Proceedings of the 7th conference on International Language Resources and Evaluation (LREC), European Languages Resources Association, 2010.

[24] Dragomir R. Radev, Sasha Blair-Goldensohn, Zhu Zhang, and Revathi Sundara Raghavan. *NewsInEssence: A System For Domain-Independent, Real-Time News Clustering and Multi-Document Summarization*. In Proceedings of the 1st International Conference on Human language technology research, Association for Computational Linguistics, pages 1-4, 2001.

[25] Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. *A comparison of extrinsic clustering evaluation metrics based on formal constraints*. In Information Retrieval Journal, Volume 12, Issue 4, pages 461-486, 2009.

[26] Fang Li, Huanye Sheng, and Dongmo Zhang. *Event Pattern Discovery from the Stock Market Bulletin*. In Proceeding of the 5th International Conference on Discovery Science (DS), Springer, pages 310-315, 2002.

[27] Fillmore C.J., Johnson C.R., and Petruck M.R.L. *Background to FrameNet*. In International Journal of Lexicography, Volume 16, Issue 3, Oxford University Press, pages 235-250, 2003.

[28] Fred Karlsson, Atro Voutilainen, Juha Heikkilae, and Arto Anttila. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Walter de Gruyter, Volume 4, ISBN: 978-3110141795, 328 pages, 1995.

[29] Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, and Franciska De Jong. *An overview of event extraction from text*. In Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011) at 10th International Semantic Web Conference (ISWC 2011), Volume 779, pages 48-57, 2011.

[30] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. *Introduction to WordNet: An On-line Lexical Database*. In 5 papers on WordNet, pages 1-9, 1993.

[31] Hamido Fujita, Ali Selamat, and Habibollah Haron, *New Trends in Software Methodologies, Tools and Techniques*. Section *Knowledge Based Engineering System for Structural Optical Design*. In Proceedings of 13th Social Media Tourism Symposium (SoMeT), 2014.

[32] Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. *Joint entity and event coreference resolution across documents*. In Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing and Natural Language Learning (EMNLP-CoNLL), pages 489-500, 2012.

[33] Heng Ji and Ralph Grishman. *Refining Event Extraction Through Cross-document Inference*. In Proceedings of Annual Meeting of the Association of Computational Linguistics, pages 254-262, 2008.

[34] Hilda Hardy, Vika Kanchakouskaya, and Tomek Strzalkowski. *Automatic event classification using surface text features*. In Proceedings of AAAI Workshop on Event Extraction and Synthesis, Boston, Massachusetts, 2006.

[35] Hong-Woo Chun, Young-Sook Hwang, and Hae-Chang Rim. *Unsupervised event extraction from biomedical literature using co-occurrence information and basic patterns.* In Proceedings of the 1st International Joint Conference on Natural Language (IJCNLP), Berlin, Heidelberg, 2005.

[36] Hristo Tanev, Jakub Piskorski, and Martin Atkinson. *Real-time news event extraction for global crisis monitoring*. In book Natural Language and Information Systems, Springer, pages 207-218, 2008.

[37] Inderjeet Mani and George Wilson. *Robust temporal processing of news*. In Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, pages 69-76, 2000.

[38] International Academic Bookstore. *LinguaLinks Library 5.0 Plus: SIL Language Freeware*. ISBN: 978-1556711237, International Academic Bookstore, 2004.

[39] Jakub Piskorski and Roman Yangarber. *Information Extraction: Past, Present and Future*. Book chapter in T. Poibeau, H. Saggion, J. Piskorski, R. Yangarber (editors) *Multi-source, Multilingual Information Extraction and Summarization*. Springer, Berlin and New York, 2013.

[40] Jakub Piskorski, Hristo Tanev, Martin Atkinson, and Erik van der Goot. *Cluster-Centric Approach to News Event Extraction*. In Proceeding of the Conference on New Trends in Multimedia and Network Information Systems, IOS Press, pages 276-290, 2008.

[41] James Allan, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. *Topic Detection and Tracking Pilot Study Final Report*. In Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, 1998.

[42] James Allan. *Topic detection and tracking: event-based information organization*. Volume 12, ISBN: 978-1461353119, Springer, 266 pages, 2002.

[43] James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer, and Graham Katz. *TimeML: Robust Specification of Event and Temporal Expressions in Text*. In Proceedings of Computational Semantics Workshop, 2003.

[44] Janara Christensen, Stephen Soderl, and Oren Etzioni. *Towards Coherent Multi-Document Summarization*. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1163-1173, June 2013.

[45] Jannik Strötgen and Michael Gertz. *HeidelTime: High Quality Rule-based Extraction and Normalization of Temporal Expressions*. In Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, pages 321-324, USA, 2010.

[46] Jari Björne and Tapio Salakoski. *Generalizing biomedical event extraction*. In Proceedings of the BioNLP Shared Task 2011 Workshop, Association for Computational Linguistics, pages 183-191, 2011.

[47] Jena D. Hwang, Archna Bhatia, Clare Bonial, Aous Mansouri, Ashwini Vaidya, Nianwen Xue, and Martha Palmer. *PropBank annotation of multilingual light verb constructions*. In Proceedings of the 4th Linguistic Annotation Workshop, Association for Computational Linguistics, pages 82-90, 2010.

[48] Jens Nilsson, Sebastian Riedel, and Deniz Yuret. *The CoNLL 2007 shared task on dependency parsing*. In Proceedings of the CoNLL shared task session of EMNLP-CoNLL, pages 915-932, 2007.

[49] Jethro Borsje, Frederik Hogenboom, and Flavius Frasincar. *Semi-automatic financial events discovery based on lexico-semantic patterns*. In International Journal of Web Engineering and Technology, Volume 6(2), pages 115-140, 2010.

[50] Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. *Overview of BioNLP'09 shared task on event extraction*. In Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task, Association for Computational Linguistics, pages 1-9, 2009.

[51] Jinho D. Choi and Martha Palmer. *Optimization of natural language processing components for robustness and scalability*. Doctoral Dissertation, 165 pages, 2012.

[52] Jinho D. Choi and Martha Palmer. *Retrieving correct semantic boundaries in dependency structure*. In Proceedings of ACL workshop on Linguistic Annotation (LAW‘10), pages 91-99, 2010.

[53] Joakim Nivre. *Dependency Grammar and Dependency Parsing*. Technical report, Växjö University, 32 pages, 2005.

[54] John Sinclair. *Corpus and Text: Basic Principles*. Book chapter in Martin Wynne (editor) *Developing linguistic corpora: a guide to good practice*. ISSN 14635194, 2004.

[55] Jonathan H. Clark and José P. González-Brenes. *Coreference Resolution: Current Trends and Future Directions*. In Language and Statistics II Literature Review, 2008.

[56] Julie A. Black and Nisheeth Ranjan. *Automated event extraction from email*. Final Report of CS224N/Ling237 course in Stanford, Spring, 2004. http://nlp.stanford.edu/courses/cs224n/2004/

[57] Karin Kipper Schuler. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. Dissertation, University of Pennsylvania, 2005.

[58] Keh-Yih Su, Tung-Hui Chiang, and Jing-Shin Chang. *An Overview of Corpus-Based Statistics-Oriented (CBSO) Techniques for Natural Language Processing*. International Journal of Computational Linguistics and Chinese Language Processing, Volume 1, Issue 1, August 1996.

[59] Linguistic Data Consortium. *ACE (Automatic Content Extraction) English Annotation Guidelines for Events.* ver. 5.4.3 2005.07.01, 2005.

[60] Lluís Màrquez, Xavier Carreras, Ken Litkowski, and Suzanne Stevenson. *Semantic Role Labeling: An Introduction to the Special Issue*. Computational Linguistics, Volume 34, Issue 2, pages 145-159, 2008.

[61] Lucien Tesnière. *Elements de syntaxe structurale*. Éditions Klinksieck, The University of Michigan, 670 pages, 1959.

[62] Maria T. Pazienza. *Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology*. ISBN-10: 354063438X, Springer, 222 pages, 1997.

[63] Martha Palmer, Daniel Gildea, and Nianwen Xue. *Semantic Role Labeling. Synthesis Lectures on Human Language Technologies*. ISBN-10: 1598298313, Morgan and Claypool Publishers, 104 pages, January 15, 2010.

[64] Martha Palmer, Daniel Gildea, and Paul Kingsbury. *The Proposition Bank: An Annotated Corpus of Semantic Roles*. In Journal Computational Linguistics, Volume 31, Issue 1, pages 71-106, March 2005.

[65] Marti A. Hearst. *Automatic acquisition of hyponyms from large text corpora*. In Proceedings of the 14th conference on Computational linguistics, Volume 2, Association for Computational Linguistics, pages 539-545, 1992.

[66] Marti A. Hearst. *To Appear in WordNet: An Electronic Lexical Database and Some of its Applications*, Christiane Fellbaum (Ed.), MIT Press, Automated Discovery of WordNet Relations, 1998.

[67] Martin Čmejrek, Jan Hajič, and Vladislav Kuboň. *Prague Czech-English Dependency Treebank: Any Hopes For A Common Annotation Scheme?* In Proceedings of HLT/NAACL 2004 Workshop: Frontiers in Corpus Annotation, pages 47-54, 2004.

[68] Michael A. Covington. *A Fundamental Algorithm for Dependency Parsing*. In Proceedings of the 39th Annual ACM Southeast Conference, pages 95-102, 2001.

[69] Michael C. McCord. *Slot Grammar: A System for Simpler Construction of Practical Natural Language Grammars*. In Rudi Studer, editor, Natural Language and Logic, volume 459 of Lecture Notes in Computer Science, Springer, pages 118-145, 1989.

[70] Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, et al. *Open language learning for information extraction*. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Association for Computational Linguistics, pages 523-534, 2012.

[71] Michele Banko, Michael Cararella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. *Open Information Extraction from the WEB*. In proceedings of the 7th International Joint Conferences on Artificial Intelligence (IJCAI), 2007.

[72] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. *Building a large annotated corpus of English: The PennTreeBank*. In Journal Computational Linguistics - Special issue on using large corpora, Volume 19, Issue 2, pages 313-330, June 1993.

[73] Nancy Chang. *Semantic role marking: A preliminary report*. Course paper, Linguistics 205, Fall 1997.

[74] Nathalie Friburger and Denis Maurel. *Textual Similarity Based on Proper Names*. In Proceedings of the workshop Mathematical/Formal Methods in Information Retrieval (MFIR), Tampere, Finland, pages 155-167, 2002.

[75] Noam Chomsky. *Aspects of the theory of syntax*. Cambridge, Massachusetts: MIT Press, ISBN: 978-0262530071, 261 pages, 1969.

[76] Peter Exner and Pierre Nugues. *Using Semantic Role Labeling to Extract Events from Wikipedia*. In Proceedings of the Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE). Workshop in conjunction with the 10th International Semantic Web Conference, pages 23-24, 2011.

[77] Philip Resnik. *WordNet and class-based probabilities*. In WordNet: An electronic lexical database, MIT Press, pages 239-263, 1998.

[78] Philipp Cimiano and Steffen Staab. *Learning by Googling*. In Association for Computing Machinery's Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD) Explorations Newsletter, Volume 6, Issue 2, pages 24-33, 2004.

[79] Pouliquen Bruno, Ralf Steinberger, Camelia Ignat, Emilia Käsper, and Irina Temnikova. *Multilingual and Cross-lingual News Topic Tracking*. In Proceedings of the 20th International Conference on Computational Linguistics (CoLing), Geneva, Switzerland, 23-27 August, 2004.

[80] Pradheep Elango. *Coreference Resolution: A Survey*. In Project report of the course Advanced Natural Language Processing, Computer Sciences Department, University of Wisconsin, Madison, 2006.

[81] Ralph Grishman, Silja Huttunen, and Roman Yangarber. *Real-time event extraction for infectious disease outbreaks*. In Proceedings of the 2nd international conference on Human Language Technology Research (HLT), Morgan Kaufmann Publishers Inc., pages 366-369, 2002.

[82] Rashmi S., Dr. M. Hanumanthappa, and Regina Lourdhu Suganthi. *Processing of Natural Languages Semantically - A Detailed Survey*. International Journal of Engineering Research & Technology, Volume 2, Issue 12, 2013.

[83] Regina Barzilay and Kathleen R. McKeown. *Sentence Fusion for Multidocument News Summarization*. In Journal Computational Linguistics, Volume 31, Issue 3, pages 297-328, September, 2005.

[84] Roberto Navigli. *Word sense disambiguation: A survey*. In ACM Computing Surveys (CSUR), Volume 41, Issue 2, Article No. 10, 69 pages, February 2009.

[85] Roser Saurí, Robert Knippen, Marc Verhagen, and James Pustejovsky. *Evita: A Robust Event Recognizer for QA Systems*. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pages 700-707, 2005.

[86] Ryan Benjamin Shaw. *Events and periods as concepts for organizing historical knowledge*. Ph.D. thesis, University of California, Berkeley, 2010.

[87] Sandra Kübler, Ryan McDonald, and Joakim Nivre. *Dependency parsing (Synthesis Lectures on Human Language Technologies)*. Morgan and Claypool Publishers, ISBN: 978-1598295962, 127 pages, 2009.

[88] Sebastian Riedel and Andrew McCallum. *Robust Biomedical Event Extraction with Dual Decomposition and Minimal. Domain Adaptation*. In Proceedings of the BioNLP Shared Task 2011 Workshop, Association for Computational Linguistics, pages 46-50, 2011.

[89] Silvia Julinda. *Extracting a Repository of Events and Event References from News Clusters*. Master Thesis, Technische Universität Berlin, Belrin, 2014.

[90] Sophia Ananiadou, Sampo Pyysalo, Jun'ichi Tsujii, and Douglas B. Kell. *Event extraction for systems biology by text mining the literature*. In Journal Trends in Biotechnology, Volume 28, Issue 7, pages 381-390, 2010.

[91] Soto Montalvo, R. Martínez, Arantza Casillas, and Víctor Fresno. *Multilingual news clustering: Feature translation vs. identification of cognate named entities*. In Journal Pattern Recognition Letters, Volume 28, Issue 16, pages 2305-2311, December, 2007.

[92] Soto Montalvo, Raquel Martínez, Arantza Casillas, and Víctor Fresno. *Bilingual News Clustering Using Named Entities and Fuzzy Similarity*. In book: Text, Speech and Dialogue, ISBN 978-3540746270, Springer, pages 107-114, 2007.

[93] Stefan Büttcher, Charles L. A. Clarke, and Gordon V. Cormack. *Information Retrieval: Implementing and Evaluating Search Engines*. ISBN: 978-0262026512, MIT Press, 632 pages, 2010.

[94] Takaharu Takeda and Atsuhiro Takasu. *Updatenews: a news clustering and summarization system using efficient text processing*. In Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries, ACM, pages 438-439, 2007.

[95] Thierry Poibeau, Horacio Saggion, and Jakub Pislorski. *Multi-source, Multilingual Information Extraction and Summarization*. ISBN 978-3-642-43090-9, 324 pages, 2013.

[96] Thomas E. Payne. *Describing Morphosyntax: A Guide for Field Linguists*., Cambridge, New York: Cambridge University Press, ISBN: 978-0521588058, 413 pages, 1997.

[97] TimeML Working Group. *Guidelines for Temporal Expression Annotation for English for TempEval 2010*. August 14, 2009.

[98] Vasin Punyakanok, Dan Roth, and Wen-tau Yih. *The importance of syntactic parsing and inference in semantic role labeling*. In Journal Computational Linguistics, Volume 34, Issue 2, pages 257-287, June 2008.

[99] Wei Wang, Dongyan Zhao, Lei Zou, Dong Wang, and Weiguo Zheng. *Extracting 5W1H Event Semantic Elements from Chinese Online News*. In Proceedings of 11th International Conference, WAIM 2010, China, pages 644-655, July 15-17, 2010.

[100] Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. *A lightweight and high performance monolingual word aligner*. In The 51st Annual Meeting of the Association for Computational Linguistics, Short Papers, Sofia, Bulgaria, pages 702-707, August 4-9, 2013.