Domain specific event topic information extraction on large text collections

Sun, Zhen

2007

Sun, Z. (2007). Domain specific event topic information extraction on large text collections.
Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/35738>

<https://doi.org/10.32657/10356/35738>

Domain Specific Event Topic Information Extraction on Large Text Collections

Sun Zhen

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Master of Engineering

2007

Abstract

In this research, we examine the problem of extracting information about domain specific event topics from a large document collection. Given a user query that specifies some seed information about an event topic, a conventional document retrieval system usually finds documents where the given query terms appear frequently but not necessarily the documents containing a complete set of information about the event topic. Documents related to the event topic but not containing the query terms will not be returned. A novel way of representing information about an event topic by its entity and relation instances has been introduced in this thesis. These entity and relation instances may be found in multiple documents. We define an extraction task that involves selecting documents relevant to a given event topic and extracting from them entity and relation instances. This task is known as domain specific event topic entity and relation extraction task. Information extraction patterns are used to extract domain relevant entity and relation instances from documents. We have proposed some document ranking strategies based on these information extraction patterns to address this event topic extraction task. These strategies are known as pattern-based document ranking strategies. Each ranking strategy attempts to assign a score to every document to estimate the gain of event topic related instances found in the document. We have conducted some experiments on three document collection datasets. The three datasets are constructed based on two terrorism events. Experiments have shown that our proposed pattern-based document ranking strategies performed well for document collections of different sizes. A personalized version of the extraction task has been studied. We propose to assign importance weights to entity and relation instances extracted and to personalize the weights based on user feedback. We further extend our document ranking strategies with the personalized entity and relation weighting scheme. We have also developed a working prototype system named INEXT to carry out the domain specific event topic extraction task.

Acknowledgments

I wish to express my deepest gratitude to my supervisor, Professor Lim Ee-Peng. I am thankful to him for providing me the opportunity for research. He has contributed endless hours of advice and guidance regarding my work. His guidance and encouragement have made my research a fulfilling and enriching experience. He spent much time discussing ideas of research with me and helping me on modifying papers and thesis, even during the most busy period. His tireless editorial effort vastly improved the quality of this thesis. I could have never completed it without his valuable ideas and suggestions.

Special thanks to my co-supervisor, Professor Chang Kui-Yu, for generating ideas and sharing his views of research. Discussion with him always gives me passion and confidence on my research work. My presentation skills have also been improved through weekly discussions with him and Professor Lim.

I would like to thanks the Director, Professor Ng Wee Keong, for making Center for Advanced Information Systems an efficient and coveted place to work in. I am especially grateful to researchers and students at the center, for their generous sharing of ideas, suggestions, and discussions, as well as the help in reviewing my writings.

Finally, I wish to thank my family to whom I owe the greatest debt. I could not have come this far without their love and support. Their sincerity, uprightness and sense of purpose have greatly influenced my life abroad and will continue to do so in the future.

Contents

Abstract	i
Acknowledgments	ii
1 Introduction	2
1.1 Motivation	2
1.2 Objectives and Contributions	5
1.3 Thesis Outline	6
2 Related Work	8
2.1 Named Entity Recognition	8
2.2 Multi-Slot Information Extraction	9
2.3 Information Extraction from Multiple Documents	11
2.4 New Event Detection	12
2.5 Coreference Resolution	13
2.6 Document Ranking and Retrieval	14
2.7 Terrorism Analysis and Visualization Tools	15
3 Domain Specific Event Topic Entity Relation Extraction Task with Document Ranking	16
3.1 Event Topic Representation using Entity and Relation Instances	16
3.2 Domain Specific Event Topic Entity Relation Extraction Task	18
3.3 Pattern-based Document Ranking	21
3.3.1 Initial Score Function	22
3.3.2 DiffCompare	22
3.3.3 CombineCompare	23

3.3.4	PartialMatch I	24
3.3.5	PartialMatch II	24
3.4	Content-based Document Ranking Strategies	25
4	Experiments	27
4.1	Datasets	27
4.2	Information Extraction System Components	31
4.2.1	Extraction of Entity Instances	32
4.2.2	Extraction of Relation Instances	33
4.3	Evaluation Settings	34
4.3.1	Ideal Document Ranking Strategy	34
4.3.2	Plus(+) Document Ranking Strategy	35
4.3.3	Performance Metrics	35
4.4	Experimental Results	36
4.4.1	Datasets with Manual Coreference Resolution	37
4.4.2	Datasets with Auto-Coreference Resolution	41
4.5	Comparisons between Patten-based vs Content-based Document Ranking Strategies	51
5	Personalized Event Topic Information Extraction	54
5.1	Personalized Weight-based Document Ranking	54
5.2	Personalized Event Topic Entity Relation Extraction	58
5.3	Experiment Setup	60
5.4	Experimental Results	61
5.4.1	KSB Case 1	61
5.4.2	KSB Case 2	61
5.4.3	AEB Case 1	66
5.4.4	AEB Case 2	66
5.4.5	Summary of Results	71

6	INEXT: A Tool for Event Topic Information Extraction	74
6.1	System Architecture	74
6.2	Document Ranking Module	76
6.3	Entity and Relation Annotation Module	78
7	Conclusion and Future Work	82
7.1	Summary of Research	82
7.2	Future Work	83
A	Named Entity Recognition and Coreference Resolution	85
A.1	Named Entity Recognition	85
A.1.1	Limitations of Named Entity Recognition	85
A.1.2	Named Entity Recognition Post-processing	86
A.2	Coreference Resolution	88
A.2.1	Overview of Coreference Resolution	88
A.2.2	Pronoun Coreference Resolution	89
A.2.3	Name Coreference Resolution	90
A.2.4	Experiments	91
B	Extraction Pattern Samples	93
C	List of Publications	95
	Bibliography	96

List of Tables

4.1	Ten relation classes with their source and target entity classes	31
4.2	Detailed Information of the three datasets	31
4.3	Summary of winning strategies on different datasets	46
5.1	Ideal weight of each entity class	57
5.2	Ideal weight of each relation class	57
5.3	Detailed Information of four sets of annotations on KSB-100 and AEB-100	60
5.4	Ideal weight of each entity class	68
5.5	Ideal weight of each relation class	68
5.6	Estimated weight of each entity class	72
5.7	Estimated weight of each relation class	73
A.1	Coreference Resolution Results	92

List of Figures

1.1	Entity Relation Graph about London Bombing	3
3.1	A relation instance example of relation class “Harm”	18
4.1	Entity Relation Graph of KSB event	29
4.2	Entity Relation Graph of AEB event	30
4.3	Performance on Extracted Instances (manual coreferenced AEB-100) . .	38
4.4	Document ranking recall (manual coreferenced AEB-100) respect to relevant document	38
4.5	Document ranking recall (manual coreferenced AEB-100) respect to optimal set	40
4.6	Performance on Extracted Instances (manual coreferenced KSB-100) . .	40
4.7	Performance on Extracted Instances (manual coreferenced AEB-1000) .	41
4.8	Performance on Extracted Instances (auto-coreferenced KSB-100)	42
4.9	Document ranking recall (auto-coreferenced KSB-100) respect to relevant document	43
4.10	Document ranking recall (auto-coreferenced KSB-100) respect to optimal set	43
4.11	Performance on Extracted Instances (auto-coreferenced AEB-100)	44
4.12	Document ranking recall (auto-coreferenced AEB-100) respect to relevant document	44
4.13	Document ranking recall (auto-coreferenced AEB-100) respect to optimal set	45
4.14	Performance on Extracted Instances (auto-coreferenced KSB-100 with MF2)	
	48	

4.15 Document ranking recall (auto-coreferenced KSB-100 with MF2) respect
to relevant document 48

4.16 Document ranking recall (auto-coreferenced KSB-100 with MF2) respect
to optimal set 49

4.17 Performance on Extracted Instances (auto-coreferenced KSB-100 with LF2) 49

4.18 Document ranking recall (auto-coreferenced KSB-100 with LF2) respect to
relevant document 50

4.19 Document ranking recall (auto-coreferenced KSB-100 with LF2) respect to
optimal set 50

4.20 Performance evaluation respect to extracted instances 52

4.21 Performance evaluation respect to relevant documents 52

4.22 Performance evaluation respect to optimal documents 53

5.1 Personalized Entity Relation Graph of KSB event 56

5.2 Entity and Relation Instances of KSB Case 1 62

5.3 Entity and Relation Instances of KSB Case 2 62

5.4 Entity and Relation Instances of AEB Case 1 63

5.5 Entity and Relation Instances of AEB Case 2 63

5.6 Performance on Extracted Instances (KSB Case 1) 64

5.7 Document selection recall (KSB Case 1) respect to optimal set 64

5.8 Document selection recall (KSB Case 1) respect to relevant document . . 65

5.9 Performance on Extracted Instances (KSB Case 2) 65

5.10 Document selection recall (KSB Case 2) respect to relevant document . . 67

5.11 Document selection recall (KSB Case 2) respect to optimal set 67

5.12 Performance on Extracted Instances (AEB Case 1) 68

5.13 Performance on Extracted Instances (AEB Case 2) 69

5.14 Document selection recall (AEB Case 2) respect to relevant document . . 69

5.15 Document selection recall (AEB Case 2) respect to optimal set 70

6.1 INEXT System Architecture 75

6.2 Query Interface 77

6.3 Document Ranking Strategies Configuration Interface 78

6.4	User Annotation Interface	79
6.5	Revised Document Ranking	80
6.6	Exported entity relation graph of the event topic	81
6.7	Content of “Graph.dot”	81
B.1	Extraction patterns of relation class “Org_Harm”	94

Chapter 1

Introduction

1.1 Motivation

Every day, thousands of news articles and reports are published and they cover a wide spectrum of events ranging from company acquisitions to terrorism attacks [2]. Information of such events is often hidden in multiple documents from possibly multiple collections. This event topic related information is an important piece of knowledge about organizations and individuals, and can be used for intelligence analysis.

For example in the terrorism domain, in order to study terrorism, terrorists and terrorist organizations, researchers have to read large number of articles to gather pieces of information and assemble them together. This effort is labor-intensive and not necessarily very effective.

We are interested to study a special event topic information extraction task where knowledge about an event topic is to be gathered from a large document collection. Consider the bombing event that took place in London on July 7, 2005. As shown in Figure 1.1, knowledge about this event topic can be represented using entity and relation instances in an *entity relation graph*. People, organizations, etc., which are involved in the event topic can be represented by entity instances. Each relation instance then represents the relationship between a pair of entity instances. To gather the above knowledge, one will have find from the large document collection those documents reporting the event topic, also known as the *relevant* documents. Nevertheless, this relevant subset may still consist of many documents and reading through all of them will require significant amount of efforts and time.

1.1. MOTIVATION

Figure 1.1: Entity Relation Graph about London Bombing

As illustrated by the above example, the relevant documents are those that contain different parts of the entity relation graph (see Figure 1.1) to be constructed. Suppose there is a document retrieval system and the large document collection has been indexed by the system. In order to construct the whole entity relation graph, a user will typically use some initial query keywords, e.g. “London bombing July 2005”, to search through the system for some relevant documents. However, it may turn out that only a few of relevant documents can be found using the initial query keywords. The user has to read through these documents to extract instances related to the event topic and also to formulate new query words using those instances. In this process, multiple queries will have to be constructed for the document retrieval system so as to extract the complete set of entity and relation instances.

The above task clearly has not been well supported by the existing document retrieval systems, which usually assume that each user query (keyword based or attribute based) is independent of other queries as they are designed to find documents that contain query terms only [9]. The way a user analyzes the returned documents is however completely beyond their scope. Query reformation systems use some full-text database to expand

1.1. MOTIVATION

a user query or to change the query structure in order to improve the search results [17]. However, these query reformation systems can only generate new queries that are semantically similar to the given user query. They therefore may return large number of documents relevant to the query but not the underlying event topic. Multi-document summarization systems are able to create a summary from multiple documents [18]. However, the extraction of the sentences forming a summary is not driven by an event topic.

In this research, we therefore study information extraction from an analyst’s standpoint. We treat information extraction as an integral part of a larger task. The task entails finding documents and extracting from them entity and relation instances. We call this the *domain specific event topic entity and relation extraction task*. A document retrieval system that supports this task will therefore have to return documents that contain the required instances and to extract them.

We aim to address the domain specific event topic entity and relation extraction task in this research. One research focus is to keep the subset of documents to be read by the user as small as possible to minimize the user’s effort. This improves the productivity of analyzing events. However, there are some technical challenges in identifying a subset of documents describing an event topic and extracting event-related information. They are:

- Information about an event topic often exists across multiple documents. In order not to miss any entity/relation instances about the event topic, it is necessary to find all documents which contain some instances about the event topic. However, the same entity/relation instances may be found in multiple documents. It is important to select documents that reduce the amount of overlap with the already found instances. Ideally, we aim to find the smallest subset of documents that can cover all the required entity/relation instances.
- Document ranking will have to be different from the existing document retrieval systems, which simply apply query term frequencies and document frequencies to assign scores to each document [9]. The document ranking for this task must be able to find all relevant documents and always give the documents containing more

1.2. OBJECTIVES AND CONTRIBUTIONS

related and novel instances higher scores. It means the relevance score of each document will change as new entity/relation instances are found.

- The extraction of entity and relation instances related to an event topic from documents requires extraction patterns. The extraction patterns must be able to identify entity instances and relationships among entity instances and they are domain specific. In our work, we aim to create extraction patterns specifically for terrorism domain.

1.2 Objectives and Contributions

We formulate a domain specific entity and relation extraction task in our work and study several document ranking strategies based on information extraction patterns to address this task. These are known as pattern-based document ranking strategies.

The main objectives and contributions arising from this thesis are summarized as follows:

- Definition of domain specific event topic entity and relation extraction task

We formally define the extraction task which incorporates a document ranking strategy to find event topic related entity and relation instances. The extraction task aims to incrementally select a set of documents relevant to an event topic, using some document ranking strategies. A set of patterns for extracting domain specific event topic entity and relation instances from a document are assumed to be given. We also assume that some seed entity instances are given to bootstrap the extraction process. This task has not been studied before and our research therefore establishes the necessary foundation.

- Design of pattern-based document ranking strategies

We propose a few document ranking strategies to identify the smallest possible subset of documents for covering event topic related instances. Each strategy aims to maximize the novelty of the set of entity and relation instances that can be found in the next document to be extracted. In this way, one can hopefully reduce the number of documents to be examined by the experts.

1.3. THESIS OUTLINE

- Construction of datasets and evaluations of proposed document ranking strategies

Since the extraction task defined in our research is a new direction to this area, there is no suitable dataset available to evaluate our proposed solutions. Therefore, we have created three datasets with two different terrorism events. We have also proposed a set of evaluation metrics to measure the ability of different strategies to find documents and instances. The experimental results showed that our strategies performed well on the propose extraction task.

- Personalized version of event topic extraction task

We propose a personalized version of the domain specific event topic information extraction task, which extends the earlier extraction task by assuming that a user is interested in only a subset of all instances relevant to the event topic. The extraction task should actually tune to user interested instances as the user provides his/her annotation feedback. An extended pattern-based document ranking strategy which is based on an entity and relation weighting scheme is proposed to address this problem. The new strategy scores documents using the estimated weights of entity and relation classes assigned by some weighting scheme based on user feedback and also the amount of related and novel instances in the documents.

- INEXT: A working prototype system

We have implemented a a working prototype system to support our proposed domain specific event topic entity and relation extraction task. It incorporates a few of the document ranking strategies proposed in this thesis. Several natural language processing components like named entity recognition and coreference resolution have also been developed and incorporated into this system.

1.3 Thesis Outline

The rest of the report is organized as follows:

- In Chapter 2, we survey the topics related to this research.

1.3. THESIS OUTLINE

- In Chapter 3, we present a formal definitions of our proposed domain specific entity relation extraction task. We also provide detailed descriptions of our proposed pattern-based document ranking strategies in this chapter.
- In Chapter 4, we give a detailed explanation on the experimental setup and results of different ranking strategies in our proposed extraction task on several datasets.
- In Chapter 5, we present a personalized version of the proposed extraction task and extend the pattern-based document ranking strategy to address this problem
- In Chapter 6, we present the working prototype system built for supporting the extraction task proposed in this research.
- In Chapter 7, we conclude this research with a summary and an overview of future research direction.

Chapter 2

Related Work

Information extraction [42] involves extracting useful text segments from a collection of documents. The focus has been traditionally on building accurate and robust information extraction systems. However, our work is to examine the IE problem from a different angle.

2.1 Named Entity Recognition

Finding entity and relation instances of a certain event topic is our research focus and this is related to named entity recognition. Named Entity Recognition (NER) involves extracting objects of general classes such as person names, locations, and organizations from plain text. NER problem has been well studied and evaluated in the Sixth Message Understanding Conference in 1995 [5] and it is an essential component of many language processing tasks.

Much of the named entity recognition research was based on English text. Palmer and Day developed statistical methods for finding named entities from newswire articles in multiple languages: Chinese, English, French, Japanese, Portuguese and Spanish [32]. They showed that the level of difficulty of the NER task was different for the six languages but a large part of the task can be performed with simple methods. In their paper, they presented a statistical profile of a Named Entity task for the different languages.

Maynard et al. [28] introduced a Muse system to extract a general class of named entities from text of different types, such as Email and spoken texts, scientific texts and

2.2. MULTI-SLOT INFORMATION EXTRACTION

religious texts. Muse was able to recognize the entity types in MUC, and two additional types: address and identifier.

Li et al. [46] presented a model for focused named entity recognition. Focused named entities represent topics in a document. It was shown in [46] that the focused named entities can be used in many natural language processing applications, such as topic detection, search engine ranking, and text summarization. A statistical model was introduced to convert the named entity recognition problem into a classification problem. A number of classification algorithms were compared and the impact of selecting different linguistic features was analyzed.

Named entity recognition can be viewed as a kind of single-slot extraction. Single-slot extraction systems such as AutoSlog [35] and its extensions [38, 36, 37, 43] have been developed and they have shown high extraction accuracy. With a set of labelled training documents, AutoSlog was able to build a set of single-slot extraction patterns for extracting entities of interest from documents.

Subsequent extensions of AutoSlog [38, 36] focused on reducing the effort in labelling training corpus. Riloff [36] presented a system called AutoSlog-TS that used only pre-classified training corpus with relevant and irrelevant documents to create dictionaries of extraction patterns. The AutoSlog-TS system used AutoSlog to find all extraction patterns first from all training documents and then used classified information on the corpus to filter out incorrectly learned patterns.

Riloff and Jones also presented a multi-level bootstrapping algorithm to generate both the semantic lexicon and extraction patterns simultaneously [37]. Ellen further presented another bootstrapping method that was able to learn semantic lexicons for multiple categories simultaneously [43].

2.2 Multi-Slot Information Extraction

The goal of multi-slot extraction is to extract several interested entities simultaneously using one extraction pattern. Since these entities are extracted with one pattern, a certain relation among the entities is observed. Therefore, multi-slot information extraction is useful for identifying relations among entities.

2.2. MULTI-SLOT INFORMATION EXTRACTION

There have been some research on multi-slot extraction [40, 39, 23, 20, 22, 31], which developed multi-slot extraction patterns for extraction purpose. All these works on multi-slot extraction were targeted at unstructured text (plain text).

LIEP [20] addressed the issue of increasing requirements of hand-crafted dictionaries of extraction patterns for most information extraction system to work. LIEP presented a system that was able to build dictionaries of multi-slot extraction patterns directly from training texts provided by users. LIEP first generated new patterns from training examples, and then generalized learned patterns that had same relationship in syntactic structure but were with different constituents, such as head words and properties. The patterns learnt were shown to perform well compared to a manual approach.

Soderland, in the Crystal project, introduced *Concept Node* (CN) for multi-slot extraction patterns and applied it to any component phrase of a sentence in a document [40]. CN represents an extraction pattern in Crystal. The generation of CN by Crystal requires an induction procedure. Firstly, Crystal creates initial CNs from a training corpus. Crystal then generalizes each pair of CNs according to their syntactic structure. It was shown that CN could be generalized as far as possible without producing errors and it always worked better than human intuitions in creating the rules for extraction. Soderland further modified Crystal to conduct Web information extraction [39].

Agichtein and Gravano, in the *Snowball* project [6], defined a relation of interest as a tuple, and used a set of pre-defined tuples for training and generating a set of multi-slot extraction patterns. Snowball is a bootstrapping-based pattern generation system and it requires very little human-interactions in pattern generation.

Named entity recognition and multi-slot information extraction are related to our research, but there are significant differences:

1. Named entity recognition focuses on identifying specific entities from text, while our research aims to find entities and relations and use them to describe a certain event topic.
2. The goal of multi-slot IE is to generate multi-slot extraction patterns to extract instances of interest from a single document, while our work uses multi-slot extraction patterns to find entities and relations from multiple documents.

2.3 Information Extraction from Multiple Documents

Complete information about an event topic is always hidden in multiple documents. The following are some research efforts on multiple document extraction [18, 27, 34, 7].

Goldstein et al. [18] presented an approach to create a summary from multiple documents. The summary was built by extracting sentences from different documents and processing the sentences in a few steps: a certain compression, passage selection and redundancy removal. The targets of extraction were sentences, while our work aims to extract entity and relation instances.

Masterson and Kushmerick addressed the problem of extracting entities from multiple documents [27]. In their work, they extended IE techniques to a multi-document extraction task and improved the IE performance by using structural and temporal relationships among documents. They created a template with fixed slots to present the information required, then applied information extraction on multiple documents to extract entities for the slots. Their research however is still a kind of single slot extraction since it does not concern identifying relationships among entities.

Reidsma et al. [34] applied information extraction on multiple documents of multimedia archives. In their work, information extracted from multiple documents were merged and integrated to obtain a more complete summary. The proposed merging algorithm performed alignment, unification and reordering on the extracted information. The merged summary then could be used to enhance the performance of multimedia archive retrieval. However, the IE method proposed was still based on single documents and only the merging algorithm considered multi-documents information.

QXtract also dealt with a similar problem [7]. In QXtract, Agichtein and Gravano developed an automatic query-based technique to retrieve documents useful for the extraction of user-defined relations from large text databases. In their work, they focused on finding a document that was useful to generate queries for wanted relation. The generated query is then used to find more documents and this process repeats itself. This work is quite similar to our work, however there are some key differences:

1. QXtract focuses on learning queries to retrieve relevant documents for extracting a target relation and was not driven by event. Our work is to find all information about a certain event topic and it may involve multiple relations.

2.4. NEW EVENT DETECTION

2. QXtract retrieves documents by a search engine and performed IE to provide tuples for generating queries. Our work on the other hand focuses on document selection and uses IE to score documents.
3. Extraction patterns in our work may change adaptively after a document is selected and annotated. The IE system of QXtract is independent of the document retrieval process.

2.4 New Event Detection

There is much research on detecting events and extracting information about events and they are related to our work.

Harsha et al. [26] presented a method to use *Link Grammar* to discover instances of events and extract them from unstructured documents. Link Grammar represented a certain relation between two constitutions using the syntactic structure in a sentence. Rules of Link Grammar had been used to find specific instances about an event. This is similar to our work as we are also interested on finding instances of entities and relations about an event topic from text. However, the event defined in [26] was restricted to a specific action only, such as kidnaping, killing. A set of synonymous verbs were used to describe a certain action and it was defined as one event. Moreover, the extraction was limited by a certain syntactic structures supported by Link Grammar and the event was assumed to exist in a single document only.

Link Detection addresses the problem on detecting stories of the same event, or stories that are linked. A story reporting an airline crash event and another story reporting the subsequent compensation to victims are considered as two linked stories. New Event Detection (NED)[8, 11, 24] is a document selection task to identify the first story of an event from an ordered collection of news articles. In NED, the only focus is to find the first story by scoring all articles. In [12], it was shown that link detection and new event detection are asymmetric and they are both related to our work as all these tasks involve finding documents about a certain event. However, the criteria of finding documents in these tasks are different.

2.5. COREFERENCE RESOLUTION

The work in [8] and [44] studied several approaches to use TF-IDF similarity metrics to resolve the NED problems. A more recent work [11] presented an enhanced version on TF-IDF method and integrated normalization techniques, source-specific models to give an improved analysis on NED. A detailed investigation by Kumaran and Allan on NED problem and two modifications were presented in [24]. Text classification models were used to classify articles into specific categories and named-entity techniques were used to create three different document representations for NED task.

NED and our work are similar in finding documents describing a certain event. However, as entity and relation instances of a certain event are usually distributed among multiple documents, the first story does not necessarily contain all the entity and relation instances of an event. The focus of our work is to find as small as possible the subset of documents that contain complete event topic related information instead of the first story document.

2.5 Coreference Resolution

Coreference resolution is a task to determine whether two text expressions appearing in an article referring to the same entity. It is an important information extraction task and it is a well-studied problem since the Sixth Message Understanding Conference in 1995 [5].

Mitkov gave a detailed survey on major coreference resolution approaches and existing applications [29]. In [29], anaphora resolution (or better known as pronoun coreference resolution) is defined as the task of finding reference between pronouns and their antecedents. Wee et al. proposed a machine learning approach to solve noun phrase coreference resolution problem in unstructured text [41]. The approach learns a classifier from a small annotated training corpus and uses the trained classifier to determine the coreference of every pair of noun phrases identified in the test corpus. A pipeline of natural language processing (NLP) tools was implemented to identify all possible noun phrases in a text document. This proposed approach was widely used as a baseline for comparisons.

Yang et al. [45] proposed two approaches to enhance the machine learning approach presented in [41]. To improve recall, a larger feature set derived by string matching was

2.6. DOCUMENT RANKING AND RETRIEVAL

used. A new selection strategy for negative training samples was also proposed so as to improve precision. An evaluation conducted based on MUC theoretic evaluation metric [30] showed that the proposed approach yields a better performance.

Zhou and Su proposed a constraint-based approach to coreference resolution problem for general noun phrases in unstructured text [19]. They defined a set of agents each holding a set of constraints to be applied on the noun phrases. The outcomes of applying constraints are then used to determine whether a pair of noun phrases could be coreferenced. It was shown that this method gave a good performance.

Coreference resolution is an important component in our work. We conduct coreference resolution on our datasets before the domain specific event topic entity relation extraction. In our work, we consider both pronoun coreference resolution and name coreference resolution. We have implemented a auto-coreference resolution module and evaluated it against the method proposed in [19] using the MUC-6 metric. Readers can refer to Appendix A for a detailed discussion.

2.6 Document Ranking and Retrieval

The term frequency and inverse document frequency (TF-IDF) weighting strategy is often used in information retrieval and text mining for ranking documents [9]. This weighting strategy presents a statistical measure used to evaluate how important a word is to a document. The importance increases proportionally to the number of times the word appears in the document but is offset by how common the word is in all of the documents in the whole document collection. It is often used by search engines to find the most relevant documents to a user query. However, TF-IDF focuses only on the document containing query terms but not event topic relevant entity and relation instances that are not covered by any query term.

Finn and Kushmerick proposed various active learning selection strategies to incrementally select documents from a large collection for user labelling so as to derive good extraction patterns [16]. In their work, they proposed several document ranking strategies to find documents from a large collection for an expert user to derive extraction patterns from them. The focus was to select documents containing as little redundant

2.7. TERRORISM ANALYSIS AND VISUALIZATION TOOLS

information among them as possible, i.e., documents selected that are textually most dissimilar to each other so as to minimize the effort of human labelling. In contrast, our work focuses on finding documents containing information related to an event topic of interest and the set of extraction patterns is assumed to be given beforehand to assist in the extraction task.

2.7 Terrorism Analysis and Visualization Tools

Zhou et al. [47] proposed an intelligent knowledge management system to support the discovery and analysis of multilingual terrorist-created Web data. In their work, they gave a definition of the "Dark Web" and stated the technical challenges in obtaining and analyzing information from the Dark Web. They developed a systematic approach to identify, extract, and collect up-to-date multilingual terrorist web data. An intelligent web-based knowledge portal integrated with advanced text and web mining techniques was also implemented. In our work, we could use this knowledge portal as our data source.

Brown [49] built a logistic regression model to predict suicide bombings in counterterrorism applications. Brown identified in his work that suicide bombing is one of the most commonly reported acts of terrorism that presents enormous difficulties in understanding and prevention. Brown proposed a spatial choice model to forecast criminal behavior, especially for suicide bombings. The fusion of logistic regression and spatial preference techniques for modelling the spatial choice of suicide bombers into a unified forecast was proposed. In our work, we extract event topic related information from a large collection of documents. Our work can help to provide inputs to [49] in the form of entity and relation instances instead of raw text documents.

Goldberg et al. [48] developed a network analysis tool named Netmap to automatically draw a graphical representation of a criminal network. They proposed several methods to create and consolidate links in order to build a social network for analysis. In our work, the information about an event topic is essentially a network, in which nodes represent entity instances and links represent relation instances. Therefore, Netmap can be a useful tool in our work to further analyze the event topic.

Chapter 3

Domain Specific Event Topic Entity Relation Extraction Task with Document Ranking

In this chapter, we first give the definitions of domain entity and relation classes. Then we define the domain specific event topic entity and relation extraction task and some notations used for the rest of this thesis. Lastly, we propose several pattern-based document ranking strategies for the extraction task.

3.1 Event Topic Representation using Entity and Relation Instances

In our extraction task, we represent an event topic by a set of entity and relation instances. The entity instances represent the people, organizations, locations, dates/times and other information involved in the event topic. The relation instances provide the links between these entity instances so as to represent their inter-relationships. In the extraction task, we assume that an expert user wants to derive all entity and relationship instances for only one event topic belonging to some domain (e.g., biomedical, terrorism, etc.). To ensure that only relevant instances are extracted, we assume that these entity and relation instances can be grouped under a few entity and relation classes, respectively, appropriate for the domain.

Let E be a set of *entity classes*, i.e. $E = \{E_1, E_2, \dots, E_n\}$, and R be a set of *relation classes*, $R = \{R_1, R_2, \dots, R_m\}$, and they are known to be relevant to the application

3.1. EVENT TOPIC REPRESENTATION USING ENTITY AND RELATION INSTANCES

domain where the extraction task is to be performed. E and R together define the type of instances to be extracted for a target event topic. An entity class E_i denotes a set of entity instances of the same type, and each entity instance is usually a noun or noun phrase appearing in the documents. Each relation class R_i represents a semantic relationship from source entity class $SourceEnt(R_i)$ to target entity class $TargetEnt(R_i)$ and is associated with an *action class* A_i . A_i refers to a set of verbs or verb phrases that relate source entity instances in $SourceEnt(R_i)$ to target entity instances in $TargetEnt(R_i)$. Each relation instance is comprised of a source entity instance from $SourceEnt(R_i)$, a target entity instance from $TargetEnt(R_i)$, and an action instance from A_i , i.e., $R_i \subseteq SourceEnt(R_i) \times A_i \times TargetEnt(R_i)$, where $SourceEnt(R_i), TargetEnt(R_i) \in E$.

We use $\mathcal{E}$ to denote the set of all entity instances contained in E , i.e. $\mathcal{E} = \cup_{i=1}^n E_i$, and $\mathcal{R}$ to denote the set of all relation instances in R i.e., $\mathcal{R} = \cup_{i=1}^m R_i$. The event topic denoted as TE is defined below:

$$TE = \mathcal{E} \cup \mathcal{R}$$

We have assumed that a set of extraction patterns EP is given, and we will use this set of extraction patterns to obtain all entity and relation instances of an event topic of interest, i.e., TE . Each extraction pattern EP_j in EP is a template that aims to extract instances of a specific entity or relation class, denoted by $Class(EP_j)$, from a natural language text document. Let $F_{extract}()$ be an extraction program using an extraction pattern EP_j . $F_{extract}()$ is therefore defined by:

$$F_{extract}(EP_j; d) = \text{extracted entity (or relation) instances}$$

Ideally, $F_{extract}(EP_j; d)$ returns exactly the set of instances in $Class(EP_j)$ that can be found in a document d . In practice, however, this is not always possible due to the imprecision in patterns learnt.

Figure 3.1 illustrates a relation instance in which an instance of the “Harm” relation class is extracted from the following sentence, “*Al-Qaeda Cell had abducted Kim Sun on Jun 17*”. In this example, the source entity instance “*Al-Qaeda Cell*” is from the “*Terrorist Organization*” entity class, the action instance “*abducted*” belongs to the “*Harm*” action class, and the target entity instance “*Kim Sun*” is from the “*Victim*” entity class.

3.2. DOMAIN SPECIFIC EVENT TOPIC ENTITY RELATION EXTRACTION TASK

Figure 3.1: A relation instance example of relation class “Harm”

3.2 Domain Specific Event Topic Entity Relation Extraction Task

The domain specific nature of this extraction task allows us to confine the instances to be from a set of entity and relation classes. We assume that these entity and relation classes are known beforehand and a set of extraction patterns are available to extract instances of these classes. Suppose we are given a set of extraction patterns EP , a collection of documents D , and a set of seed entity instances W belonging to an event topic of interest to a user. Let E and R represent the entity classes and relation classes relevant to the event topic respectively. We use $\mathcal{E}$ to denote the set of all entity instances contained in E , i.e. $\mathcal{E} = \cup_{i=1}^n E_i$, and $\mathcal{R}$ to denote the set of all relation instances in R i.e., $\mathcal{R} = \cup_{i=1}^m R_i$. W is a small subset of $\mathcal{E}$ to bootstrap the extraction of other instances. To ensure that all instances will be extracted given the seed entity instances W , we require all instances in $\mathcal{E}$ to be directly or indirectly linked to W through same relation instances in $\mathcal{R}$.

Let r_s denote a relation instance in $\mathcal{R}$, of which e_s^s and e_s^t are the corresponding source and target entity instances, respectively. The relation instance is a relevant relation instance only if it has at least one entity instance inside the seeds or there is a path linking from this relation instance to a certain relation instance which contains at least one entity instance inside the seeds. Two relation instances are linked as long as they share one entity instance between them. Therefore, $r_s = (e_s^s, a_s, e_s^t)$ is a relation instance in $\mathcal{R}$ if it satisfies one of the conditions below:

1. $(e_s^s \in W)$ or $(e_s^t \in W)$
2. $\exists \text{ path } \Pi(r_s, r_{s+1}, \dots, r_{s+m}), r_{s+m} = (e_{s+m}^s, a_{s+m}, e_{s+m}^t)$ and $[(e_{s+m}^s \in W) \text{ or } (e_{s+m}^t \in W)]$

3.2. DOMAIN SPECIFIC EVENT TOPIC ENTITY RELATION EXTRACTION TASK

In the **domain specific event topic entity relation extraction task**, documents for extracting event topic related instances are selected one at a time. At the beginning, the seed entity instances W are given to identify the relevant documents. Each time a document is selected, it is given to the expert user for manual inspection so as to annotate the event topic entity and relation instances. In manual annotation, the expert user selects a document based on the score. The user will read through the document together with auto-extracted entity/relation instances highlighted inside the document (refer to Figure 6.4). The user will manually determine the entity and relation instances and extract them from the document. Note that manual annotation is conducted to ensure that no instances are missed. This process repeats until all event topic entity and relation instances are found.

Algorithm 1 Domain Specific Event Topic Entity and Relation Extraction

```

1: input:  $EP, D, W$ 
2: for each document  $d_j$  in  $D$  do
3:   apply  $EP$  on  $d_j$  to obtain  $\mathcal{E}'_j, \mathcal{R}'_j$  (Extraction Step)
4:   compute  $score(d_j)$  using initial score function (see Section 3.3.1)
5: end for
6: repeat
7:   //User selects a document  $d_s$ 
8:   Move  $d_s$  from  $D$  to  $S$ 
9:   //User annotates  $\mathcal{E}_s$  and  $\mathcal{R}_s$  in  $d_s$ 
10:  Add  $\mathcal{E}_s$  and  $\mathcal{R}_s$  to  $\mathcal{E}$  and  $\mathcal{R}$  respectively
11:  for each document  $d_j$  in  $D$  do
12:    compute  $score(d_j)$  based on  $\mathcal{E}'_j, \mathcal{R}'_j, \mathcal{E}, \mathcal{R}$  (using one of the document ranking
      strategies in Section 3.3.2-3.3.5)
13:  end for
14: until termination condition is satisfied
15: output:  $S, \mathcal{E}, \mathcal{R}$ 

```

The detailed description of the task is depicted in Algorithm 1. In the extraction task, the extraction patterns EP are used to find the existence of entity and relation instances that could be relevant to the event topic. The extraction patterns can be for single-slot, or multi-slot extraction. The former is designed for extracting entity instances while the latter can be used for extracting both entity and relation instances. The entity and relation instances extracted from a document d_j using EP are stored in $\mathcal{E}'_j$ and $\mathcal{R}'_j$ respectively.

3.2. DOMAIN SPECIFIC EVENT TOPIC ENTITY RELATION EXTRACTION TASK

In each iteration, a document ranking strategy is used to score every document from D . The expert user then selects a document, presumably from the top scored ones, annotates entity and relation instances and adds them to $\mathcal{E}$ and $\mathcal{R}$ respectively. The process repeats until all instances are found, or a termination condition is met. For example, the termination condition can be some upper bound on the number of documents extracted, or the number of instances found.

Assuming that the expert user has in mind a set of entity and relation instances to be extracted for an event topic. We can then define a set of documents containing relevant instances as *relevant set* denoted by L . The objective of the domain specific event topic entity and relation extraction task is to select the smallest subset O of L that cover all relevant instances. We call O the **optimal set**. Recall that $\mathcal{E}_j$ and $\mathcal{R}_j$ denote the set of entity and relation instances in document d_j respectively. A set of documents is an optimal set only if documents inside it cover all entity and relation instances and there is no other set of documents that contains all entity and relation instances but has fewer number of documents. Then O is an optimal set if and only if it satisfies the following two conditions:

1. $(\cup_{d_j \in O} \mathcal{R}_j = \mathcal{R})$ and $(\cup_{d_j \in O} \mathcal{E}_j = \mathcal{E})$
2. $\nexists O'$ s.t. $(\cup_{d_j \in O'} \mathcal{R}_j = \mathcal{R})$ and $(\cup_{d_j \in O'} \mathcal{E}_j = \mathcal{E})$ and $(|O'| < |O|)$

Document ranking strategies using different score functions are required in the above extraction task. In general, documents containing more novel and related information about the event topic of interest should be given higher scores. In Section 3.3, we propose five document ranking strategies, each based on a different score function. Since all these strategies rely on extraction patterns, we call them **pattern-based document ranking strategies**.

The manner in which extraction patterns are applied in the above entity relation extraction task is another focus of our work. Well-designed extraction patterns can extract entity and relation instances more accurately, which leads to more accurate document ranking. Furthermore, there are some other research issues for the above extraction task:

3.3. PATTERN-BASED DOCUMENT RANKING

1. The extraction task is driven by events. User always annotates the document with highest score returned by our proposed score functions of document ranking strategies. It is important for the score function to give higher scores for documents containing information related to the event topic of interest. At the same time, we do not want the user to annotate documents containing information already been seen. Therefore, it is challenging to design a good score function.
2. The proposed extraction task has not been studied before. It is necessary to design a set of evaluation metrics to measure the performance of different document ranking strategies on the extraction task. The evaluation metrics must be able to measure how well the extraction task on finding the event-related instances and how accurate the ranking strategies on scoring each document.

3.3 Pattern-based Document Ranking

A document ranking strategy is required in our proposed event-driven extraction task to select the best document for a user to annotate. The TF-IDF weight (term frequency - inverse document frequency) is a weight often used in information retrieval and text mining for ranking documents [9]. This weight is a statistical measure used to evaluate how important a word is to a document. The importance increases proportionally to the number of times a word appears in the document but is offset by how common the word is in all of the documents in the collection. It is often used by search engines to find the most relevant documents to a user query. However, TF-IDF weight focused on the document containing as many as possible of words (term) inside a user query and it cannot give high weights to documents containing novel information (entity/relation instances) related to an event topic but not containing any term of a given query.

We have proposed several document ranking strategies using different score functions for the event topic driven extraction task. In the following, we present the score functions of these document ranking strategies. We also describe two existing document ranking strategies based on document content [16], i.e., not using the extraction patterns.

3.3. PATTERN-BASED DOCUMENT RANKING

3.3.1 Initial Score Function

This is the default score function that is used to rank documents based on the given seed entity instances W and is used in the first iteration only when there is no other entity or relation instance already known by the user. The initial score function is designed to assign higher scores to documents that have more extraction patterns fired. The first term of the score formula 3.1 considers the proportion of W that is extracted. This is to give higher scores to documents containing more seeds. The second term considers the number of extraction patterns fired on the document, and the third term takes account of the number of relation instances that can be extracted from the document. The more extraction patterns that are fired and the more relation instances that are extracted, the more likely the document contains domain relevant entity and relation instances.

$$score(d_j) = \frac{|\mathcal{E}'_j \cap W| + \frac{|W|}{\gamma}}{|W|} \cdot \log_2(|EP_j|) \cdot \sum_{k=1}^{|EP_j|} f_k \quad (3.1)$$

where $\gamma \gg |W|$ is a smoothing factor that prevents the numerator from becoming zero if $\mathcal{E}'_j \cap W$ is empty. EP_j is a subset of EP that fired on document d_j , and f_k is the number of relation instances extracted by extraction pattern $ep_{j,k}$.

Compared to TF-IDF weight, initial score function considers both the importance of seed terms to a document and the potential novel and related event topic information inside a document. The objective is not only to find a document with the most occurrences of those seed terms, but also new entity/relation instances related to the event topic but not in the seeds.

Initial score strategy is the only one that does not depend on a pre-selected set of documents or annotated instances. It depends solely on the entity/relation instances extracted by the extraction patterns on the candidate documents. On the contrary, the other proposed strategies shown below require a pre-existing set of selected documents or annotated instances in order to give a score on a candidate document.

3.3.2 DiffCompare

This strategy examines the amount of overlap between relation instances extracted from the current document d_j with the accumulated relation instance set $\mathcal{R}$. We only consider

3.3. PATTERN-BASED DOCUMENT RANKING

relation instances here. This is because our assumption that entity instances annotated are usually linked to previously found entity instances via relation instances in each iteration. The score will be determined by the overlap between relation instances as well as the intersection between the extracted entity instances $\mathcal{E}'_j$ and W . This is to assign higher score for documents having direct links to the seed set. Contribution from the two factors are linearly weighted by $\alpha \in [0, 1]$. Equation 3.2 shows the score function:

$$\text{score}(d_j) = \alpha \cdot \frac{|\mathcal{R}'_j - \mathcal{R}|}{\max_{d_l \in D} |\mathcal{R}'_l|} + (1 - \alpha) \cdot \frac{|\mathcal{E}'_j \cap W|}{|W|} \quad (3.2)$$

Compared to initial score function, DiffCompare takes into account of the annotated entity and relation instances accumulated from previous selections. DiffCompare only needs to compute the difference between extracted relation instances and annotated instances as well as the intersection between seeds and extracted entity instances. Therefore, it is the most efficient strategy among all the following strategies. It works well for information distributed in different documents with few overlaps.

3.3.3 CombineCompare

This strategy combines the amount of intersection and dissimilarity between relation instances extracted from d_i with instances in $\mathcal{R}$. A modifier $\beta \in [0, 1]$ is used to adjust the relative importance of overlapping relation instances compared with novel relation instances (i.e., relevant relation instances that have not been extracted so far). When the former is more important, $\beta > 0.5$. When $\beta = 0.5$, both are treated equally important. Equation 3.3 gives the score function of this strategy. Note that when $\beta = 0$, this is equivalent to DiffCompare.

$$\text{score}(d_j) = \alpha \cdot \frac{\beta \cdot |\mathcal{R}'_j \cap \mathcal{R}| + (1 - \beta) \cdot (|\mathcal{R}'_j - \mathcal{R}|)}{\max_{d_l \in D} |\mathcal{R}'_l|} + (1 - \alpha) \cdot \frac{|\mathcal{E}'_j \cap W|}{|W|} \quad (3.3)$$

CombineCompare is fairly efficient, as it only needs to calculate the intersection and disjunction of two relation instance sets. Compared to DiffCompare, it has one additional user-adjustable parameter β , which makes it more robust and flexible.

3.3. PATTERN-BASED DOCUMENT RANKING

3.3.4 PartialMatch I

In this document ranking strategy, we want to select documents with relation instances linked to those entity instances that have already been found. This requires a partial match between the former and latter. Note that all entity instances in the event topic are connected with others using relation instances. This applies even in the midst of extraction task. Hence, we need to conduct partial match between a relation instance extracted using *EP* and the relation instances found so far.

Given two relation instances $r_s = (e_s^s, a_s, e_s^t)$ and $r_t = (e_t^s, a_t, e_t^t)$, the partial match of r_s and r_t denoted by $PartialMatch(r_s, r_t)$ is defined by:

$$PartialMatch = \begin{cases} 0 & \text{if } e_s^s \neq e_t^s \wedge e_s^t \neq e_t^t \\ 0 & \text{if } e_s^s = e_t^s \wedge e_s^t = e_t^t \wedge (a_s \in A_p, a_t \in A_q, p \neq q) \\ 1 & \text{otherwise} \end{cases}$$

The *PartialMatch* comparison is conducted between an auto-extracted relation instance and an annotated relation instance. It will always return a successful match (i.e., 1) unless there is no entity instance found equally between the two relation instances or these two relation instances are identical. The idea of this partial match is to judge the potentiality of an auto-extracted instance containing novel but related event topic information compared to those already found relation instances.

With *PartialMatch* measuring the novelty of instances, we now define its score function in Equation 3.4:

$$score(d_j) = \alpha \cdot \frac{\sum_{k=1}^{M_j} \sum_{h=1}^{|\mathcal{R}|} PartialMatch(r'_{j,k}, r_h)}{|\mathcal{R}'_j| \cdot |\mathcal{R}| + 1} + (1 - \alpha) \cdot \frac{|\mathcal{E}'_j \cap W|}{|W|} \quad (3.4)$$

where M_j is the number of relation instances extracted from d_j using *EP*; $r'_{j,k}$ is the kth relation instance from $\mathcal{R}'_j$; and r_h is the hth instance in $\mathcal{R}$.

3.3.5 PartialMatch II

In *PartialMatch I*, we consider the relation instances of a document linked to those instances found and also the amount of entity instances intersected with the seeds. In

3.4. CONTENT-BASED DOCUMENT RANKING STRATEGIES

PartialMatch II, we also want to consider the importance of those auto-extracted entity instance to those which have been annotated by the user. Therefore, we extend the PartialMatch I document ranking strategy to also consider the amount of entity instances observed in $\mathcal{E}$, and we call this strategy PartialMatch II

With the *PartialMatch* function defined in Section 3.3.4, we now derive the score function for the PartialMatch II strategy in Equation 3.5:

$$score(d_j) = \alpha \cdot \frac{\sum_{k=1}^{|\mathcal{R}'_j|} \sum_{h=1}^{|\mathcal{R}|} PartialMatch(r'_{j,k}, r_h)}{|\mathcal{R}'_j| \cdot |\mathcal{R}| + 1} + (1-\alpha) \cdot \left(\lambda \cdot \frac{|\mathcal{E}'_j \cap \mathcal{E}|}{|\mathcal{E}|} + (1-\lambda) \cdot \frac{|\mathcal{E}'_j \cap W'|}{|W'|} \right) \quad (3.5)$$

where λ is a factor to control the importance between the seeds and those annotated entity instances, a higher λ denotes that a document containing more seed terms will be given a higher score, in the contrary a lower λ gives a document containing more annotated entity instances a higher score; $W' = W - \mathcal{E}$ presents the seeds which are not extracted by user and not included into the extracted entity instance set.

PartialMatch (I and II) requires more computations as it compares every pair of relation instances. If there are M relation instances in the resultant set, and N relation instances are extracted from document d_j , the computational complexity of $score(d_j)$ is $O(MN)$. Moreover, it is good for selecting documents containing both related and novel information.

3.4 Content-based Document Ranking Strategies

Finn and Kushmerick developed content-based document ranking strategies for identifying the documents to be manually labelled so as to learn the extraction patterns. Although the nature of the problem is different, the document ranking strategies presented in [16] are related to our work. Among the strategies are *COMPARE* and *EXTRACT-COMPARE* which have been shown to give good performance[16].

COMPARE favors documents that are content-wise least similar to the pool of already selected documents. The score is measured using Equation 3.6.

$$score(d_j) = \sum_{d_k \in S} \frac{|T_j \cup T_k|}{|T_j \cap T_k| + 1} \quad (3.6)$$

3.4. CONTENT-BASED DOCUMENT RANKING STRATEGIES

where S is the set of already selected documents; T_j and T_k are sets of terms from d_j and d_k respectively.

EXTRACTCOMPARE gives preference to the document with content most dissimilar to the current set of extracted entities from the already selected document pool. This strategy applies extraction rules only to those already selected documents in S , but evaluates a candidate document purely based on its content similarity. Therefore, we consider it a content-based strategy. Equation 3.7 gives the score function:

$$score(d_j) = \sum_{d_k \in S} \frac{|\mathcal{E}'_k \cup T_j|}{|\mathcal{E}'_k \cap T_j| + 1} \quad (3.7)$$

where $\mathcal{E}'_k$ is the set of entity instances extracted from document k .

In our experiments (see Chapter 4), we have implemented COMPARE and EXTRACTCOMPARE strategies with one minor modification. Instead of randomly selecting a document as the first document as in [16], we select the highest scored document according to the initial score function. We feel that this modification gives a fairer comparison between the two content-based strategies and our proposed pattern-based strategies.

Chapter 4

Experiments

We now present the experiments evaluating the performance of different document ranking strategies on the domain specific event topic entity and relation extraction task. We first give a detailed description of the experiment setup, including the construction of datasets, choice of information extraction system components, construction of extraction patterns and performance metrics. This is followed by experimental results and discussions.

4.1 Datasets

We used three datasets, one covering Korean Hostage Kim-Sun's beheading event (KSB) in Iraq, June 2004 and two covering the Australian Embassy bombing event (AEB) in Jakarta, September 2004. They are known as the KSB-100, AEB-100 and AEB-1000 datasets respectively. Documents of these datasets were chosen based on each event topic from a collection of web documents. The collection of documents were downloaded from an online news website and the documents have been converted to plain text. More than 10 thousand documents downloaded during the week after each of the two terrorism events occurred.

KSB-100 dataset has 100 documents. There are 34 relevant documents (see Chapter 3) and 66 irrelevant documents in the dataset. The 34 relevant documents were selected by an expert familiar with the beheading event. The 66 irrelevant documents were also selected from the 10 thousand documents downloaded during the week after the event occurred. These irrelevant documents were intentionally selected among those describing

4.1. DATASETS

some other criminal events, such as murders and kidnaps. WordNet[15] is a lexical database for English language. We use the word “kill”, “murder”, “kidnap”, and the synonyms obtained using WordNet to select these irrelevant documents with the help of some search engines. In other words, both relevant and irrelevant documents describe some terrorism and crime related events. This increases the level of difficulty in document ranking. The entity and relation instances about the event topic were then determined by the expert from the relevant documents as shown in Figure 4.1. In Figure 4.1, the number assigned to each entity/relation instance denotes the number of documents where the entity/relation instance appears. To simulate a user conducting domain specific event topic entity and relation extraction task, query seed words are then chosen. The query seed words used in the extraction task for this dataset are “Kim Sun”, “Al-Qaeda Cell” and “Beheading of Kim Sun”. The seeds were chosen randomly from the entity relation graph of KSB-100.

AEB-100 dataset has 100 documents consisting of 34 relevant documents and 66 irrelevant documents. The 34 relevant documents were selected in a same manner as for KSB-100 dataset by an expert familiar with the event topic. The 66 irrelevant documents are the same as those used in KSB-100 dataset. Figure 4.2 shows the entity relation graph of AEB-100, in which the entity and relation instances were determined by the expert by reading through the relevant documents. The query seed words used for the extraction task in AEB-100 are: “Australian Embassy”, “Australian Embassy Bombing”, “Suicide Bombers” and “Elisabeth Musu”. Among the seeds of AEB-100, Elisabeth Musu was chosen to ensure the seeds are connected to all other instances in the entity relation graph.

AEB-1000 dataset has 1000 documents. Both the relevant documents and the seed words in AEB-1000 are identical to that of AEB-100. In addition to the 66 irrelevant documents, we randomly selected 900 irrelevant documents for AEB-1000, in which most of irrelevant documents are not from the terrorism and crime related events. With AEB-1000, we can evaluate the performance of our document ranking strategies for a larger dataset.

In the terrorism domain, we have adopted seven *entity classes* and ten *relation classes*. The entity classes include *Victim*, *Terrorist*, *Terrorist Org*, *Event*, *Location*, *Employer*,

4.1. DATASETS

Figure 4.1: Entity Relation Graph of KSB event

4.1. DATASETS

Figure 4.2: Entity Relation Graph of AEB event

4.2. INFORMATION EXTRACTION SYSTEM COMPONENTS

Table 4.1: Ten relation classes with their source and target entity classes

$\langle \text{Relation} \rangle$	$\langle \langle \text{SrcEnt} \rangle, \langle \text{TgtEnt} \rangle \rangle$	$\langle \text{Relation} \rangle$	$\langle \langle \text{SrcEnt} \rangle, \langle \text{TgtEnt} \rangle \rangle$
<i>Terrorist_Harm</i>	(Terrorist, Victim)	<i>Org_Harm</i>	(Terrorist Org, Victim)
<i>Connect_to</i>	(Terrorist Org, Terrorist Org)	<i>Terrorist_Resp_for</i>	(Terrorist, Event)
<i>Org_Resp_for</i>	(Terrorist Org, Event)	<i>Member_of</i>	(Terrorist, Terrorist Org)
<i>Leader_of</i>	(Terrorist, Terrorist Org)	<i>Happen</i>	(Event, Location)
<i>Work_for</i>	(Victim, Employer)	<i>Relative_of</i>	(Victim, Relative)

and *Relative*. The relation classes are *Terrorist_Harm*¹, *Org_Harm*², *Connect_to*, *Terrorist_Resp_for*³, *Org_Resp_for*⁴, *Member_of*, *Leader_of*, *Happen*, *Work_for*, and *Relative_of*. Each relation class is defined for a pair of *source* and *target entity classes* as shown in Table 4.1. Table 4.2 shows more detailed information about the three datasets.

Table 4.2: Detailed Information of the three datasets

	$ \mathcal{E} $	$ \mathcal{R} $	# of relevant docs	# of optimal docs	Total # of docs
KSB-100	18	22	34	9	100
AEB-100	19	20	34	9	100
AEB-1000	19	20	34	9	1000

4.2 Information Extraction System Components

As shown in Algorithm 1, our experiments requires a set of extraction patterns to be applied by some IE system components in the extraction step. In general, our proposed document ranking strategies are independent of the IE system components used, and the latter can be treated as a “black box”. The only requirement of the IE system components is that it is able to perform extraction of entity and relation instances from text documents.

In our work, the IE system components chosen are Crystal [40] and BBN Identifinder [14]. Crystal is a system that is able to automatically induce a dictionary of extraction patterns to identify specific information of interest from a training corpus. The extraction pattern generated is called concept node in Crystal, and it can support both single-slot and multi-slot extraction patterns. The implementation of Crystal was called Badger

¹This represents the terrorist harming a victim.

²This represents the terrorist organization harming a victim.

³This represents the terrorist responsible for an event topic.

⁴This represents the terrorist organization responsible for an event topic.

4.2. INFORMATION EXTRACTION SYSTEM COMPONENTS

information extraction system by the Center for Intelligent Information Retrieval at the University of Massachusetts [1]. The Badger IE system is an open source application and free to be used for research purpose. BBN Identifinder is a well-known named entity extraction system and we choose BBN Identifinder to extract entity instances in the extraction step.

The objective of domain entity and relation extraction system components is to determine the domain specific entity instances and inter-relationships among them as opposed to general named entity recognition. Moreover, we would like to determine the semantic classes for these entity and relation instances.

4.2.1 Extraction of Entity Instances

In our work, we have implemented three entity extraction system components to extract the entity instances from each document. They are:

1. BBN Named Entity Identifier

BBN Identifinder is a well-known named entity extraction system and it performs well on named entity extraction task. We choose BBN Identifinder to extract person names, organization names and locations from each documents and include the identified entities into the extracted entity instances set. This is helpful to find those individual entities in the document that are not involved in any relations. The domain specific semantic classes cannot be assigned to the entity instances extracted using BBN Identifinder. This is because BBN Identifinder only extracts general types of named entities. However, since our proposed document ranking strategies do not incorporate entity classes into the score function, it is fine to apply BBN Identifinder to extract entity instances without assigning their corresponding domain specific semantic classes in our experiments.

2. Known Directory of Terrorists and Organizations

In our work, we have gathered 21 terrorist names from the FBI website⁵ and 54 terrorist organization names from the ICT website⁶. These well-known terrorists

⁵<http://www.fbi.gov/mostwant/terrorists/fugitives.htm>

⁶<http://www.ict.org.il>

4.2. INFORMATION EXTRACTION SYSTEM COMPONENTS

and terrorist organizations added to our extracted entity instances set if they are found in a document.

3. Crystal Single-Slot Extraction Patterns

A set of single-slot extraction patterns based on some common linguistic structures is used to extract the domain specific entity instances of the defined entity classes. The linguistic structures include $\langle \text{Subject} \rangle \langle \text{Verb} \rangle \langle \text{Object} \rangle$, $\langle \text{Subject} \rangle \langle \text{Verb} \rangle \langle \text{Prepositional Phrase} \rangle$, $\langle \text{Verb} \rangle \langle \text{Object} \rangle \langle \text{Prepositional Phrase} \rangle$ and $\langle \text{Subject} \rangle \langle \text{Prepositional Phrase} \rangle$.

Each part of the linguistic structure is treated as a slot in the extraction pattern. By leaving one slot as a variable without constraining it to some entity instances or action instances, we obtain multiple single-slot extraction patterns. For example, $\langle \text{Subject}(\text{Terrorist}) \rangle \langle \text{Verb}(\text{"belong"}) \rangle \langle \text{Object}(\text{"Al-Qaeda Cell"}) \rangle$ is a single-slot extraction pattern that is used to extract instances of entity class *Terrorist*, in which the slot $\langle \text{Subject}(\text{Terrorist}) \rangle$ is a variable and the other two slots are constrained by the action instance "belong" and the entity instance "Al-Qaeda Cell" respectively.

The list of terrorist names and terrorist organization names are used to obtain some single-slot extraction patterns in our experiments.

4.2.2 Extraction of Relation Instances

In order to extract relation instances, we need some multi-slot extraction patterns. By constraining only one slot of some linguistic structures and leaving at least 2 slots as variables, we can obtain some multi-slot extraction patterns. For example, $\langle \text{Subject}(\text{Terrorist}) \rangle \langle \text{Verb}(\text{"belong"}) \rangle \langle \text{Object}(\text{Terrorist Org}) \rangle$ is a multi-slot extraction pattern that can extract an instance of the *Member_Of* relation. The entity class variables (i.e., *Terrorist* and *Terrorist Org*) in parentheses represent entities to be extracted to form a relation. To generalize the multi-slot extraction pattern, we use WordNet[15] to obtain other action instances with similar meaning. For example, "belong" can be replaced by "is member of", "part of", etc., to obtain other extraction patterns.

We have manually created the set of multi-slot extraction patterns based on Crystal Concept Node format. There are 41 different multi-slot extraction patterns created in this

4.3. EVALUATION SETTINGS

experiment. Readers can refer to Appendix B for some examples of extraction patterns used in our work.

4.3 Evaluation Settings

Our experiments evaluated all five document ranking strategies, namely: DiffCompare, CombineCompare, PartialMatch I, PartialMatch II and PartialMatch+ (see Section 4.3.2). We assumed the user will always select the document with the highest score assigned by each strategy during the extraction task. It is possible that multiple documents will be assigned the same highest score by some document ranking strategies. We therefore propose two steps to resolve this. The first step is to sort the documents with the same score by their initial scores and the one with highest initial score will be selected. If identical scores are still observed after the first step, random selection will be performed among the documents.

In all the experiments, we set $\alpha = 0.6$ to give it more weight to the relation instances instead of entity instances as some entity instances are already part of relation instances. For CombineCompare, we set $\beta = 0.8$ as it gave the best performance among different β values we have experimented. For PartialMatch II, we set $\lambda = 0.5$ to give equal importance to both seeds and user extracted entity instances. We have run through the extraction task over 100 iterations. To have a more detailed analysis of the performance of different strategies, we zoom into the first 45 iterations of the extraction task in the experiment.

4.3.1 Ideal Document Ranking Strategy

We introduce an ideal document ranking strategy here to compare our proposed document ranking strategies. The ideal strategy selects for each iteration a document that gives the largest gain in the performance measurement and the selected document must be a relevant document. Here, the ideal strategy assumes that instances in the entity relation graph are known. Therefore, the score formula for ideal document ranking strategy is defined as follows:

$$score(d_i) = M(d_i)$$

4.3. EVALUATION SETTINGS

where M refers to the improvement of a chosen performance metric brought by the document d_i . Refer to Section 4.3.3 for the set of performance metrics to be measured.

Note that the ideal document ranking strategy is not achievable in practice as we do not know the entity/relation instances to be extracted from each document beforehand.

4.3.2 Plus(+) Document Ranking Strategy

Plus(+) is an enhancement that can be applied to any of the proposed pattern-based strategies (i.e., DiffCompare, CombineCompare, PartialMatch I and II). The idea is to incrementally increase the number of single-slot extraction patterns by instantiating some slots of some extraction patterns with new entity instances. For example, $\langle \text{Subject}(\text{Terrorist}) \rangle \langle \text{Verb}(\text{"belong"}) \rangle \langle \text{Object}(\text{"Al-Qaeda Cell"}) \rangle$ is a single-slot extraction pattern that can extract an instance of the *Terrorist* entity class. Suppose a new entity instance "Tawhid Wa Al-Jihad" of entity class *Terrorist Org* is annotated by a user. By instantiating the "Object" slot with this newly found entity instance, we obtain a new single-slot extraction pattern, which is $\langle \text{Subject}(\text{Terrorist}) \rangle \langle \text{Verb}(\text{"belong"}) \rangle \langle \text{Object}(\text{"Tawhid Wa Al-Jihad"}) \rangle$. In addition, the new single-slot extraction pattern may extract more relation instances as well. Once some entity instances of the *Terrorist* class are extracted by the pattern, some new relation instances representing the connections between those entity instances and "Tawhid Wa Al-Jihad" will also be extracted.

The Plus-based methods requires new extraction patterns to be considered after every iteration. Thus, this strategy requires additional computational efforts. So far, we have enhanced PartialMatch I with Plus and named it PartialMatch plus(+).

4.3.3 Performance Metrics

We also propose a set of performance metrics defined below and they were measured after every document was selected. These performance metrics focus on how much relevant instances the selected documents contain, how accurate the relevant documents or optimal documents are selected.

1. Evaluation on Extracted Entity and Relation Instances

4.4. EXPERIMENTAL RESULTS

Suppose we have all relevant entity instances in set $\mathcal{E}_r$ and all relevant relation instances in set $\mathcal{R}_r$. To evaluate on the resultant instance sets obtained in extraction task i.e., $\mathcal{E}$ and $\mathcal{R}$, the recall measure is defined as follows:

- $Recall_{average} = \frac{1}{2} (Recall_{entity} + Recall_{relation})$
 where $Recall_{entity} = \frac{|\mathcal{E}_r \cap \mathcal{E}|}{|\mathcal{E}_r|}$ and $Recall_{relation} = \frac{|\mathcal{R}_r \cap \mathcal{R}|}{|\mathcal{R}_r|}$

2. Evaluation on Document Ranking

Let L be the set of all relevant documents and S denote the set of selected documents. The precision and recall measures with respect to the relevant documents are defined as follows:

- $Recall_{rel.doc} = \frac{|S \cap L|}{|L|}$
- $Precision_{rel.doc} = \frac{|S \cap L|}{|S|}$

Suppose there are v different optimal sets among all relevant documents as the optimal set is not always unique. Let $\mathcal{O}$ denote the set of all optimal sets, i.e., $\mathcal{O} = \{O_1, O_2, \dots, O_v\}$. We have $|O_1| = |O_2| = \dots = |O_v|$. Therefore, the recall and precision measures respect to optimal set are defined as follows:

- $Recall_{opt.doc} = \frac{\max_{O_i \in \mathcal{O}} |O_i \cap S|}{|O_1|}$
- $Precision_{opt.doc} = \frac{\max_{O_i \in \mathcal{O}} |O_i \cap S|}{|S|}$

Since $Precision_{rel.doc}$ and $Precision_{opt.doc}$ can be directly derived from $Recall_{rel.doc}$ and $Recall_{opt.doc}$ respectively for each iteration, we therefore decided not to use them in the subsequent evaluations.

4.4 Experimental Results

In this section, we first show the experimental results of six document ranking strategies on the domain specific event topic entity and relation extraction task using the datasets with manual coreference resolution, including AEB-100, KSB-100 and AEB-1000. Coreference resolution refers to determining the identity relationships among objects within

4.4. EXPERIMENTAL RESULTS

documents. An expert was asked to manually resolve the identity relationship among objects with documents, and we name it manual coreference resolution. Secondly, we show the experimental results of different functions on the datasets with auto-coreference resolution, including AEB-100 and KSB-100. Auto-coreference resolution refers to resolving the identity relationships among objects in documents automatically by some programs. Readers can refer to Appendix A for a detailed description of the auto-coreference resolution components implemented in our work. Thirdly, we present some experimental results using different choices of seeds on the auto-coreferenced datasets in both KSB-100 and AEB-100.

4.4.1 Datasets with Manual Coreference Resolution

AEB-100: Figure 4.3 shows the $Recall_{average}$ of the six document ranking strategies for the AEB-100 dataset. PartialMatch I gave the best performance as it achieved perfect recall with the smallest number of iterations (documents selected). PartialMatch+ and PartialMatch II were the runners-up, which achieved perfect recall a few more iterations after PartialMatch I, followed by DiffCompare and CombineCompare. PartialMatch I selected less than half the number of documents than CombineCompare and DiffCompare to obtain all entity and relation instances, although in the first few iterations, DiffCompare and CombineCompare had higher $Recall_{average}$. This is because PartialMatch was more likely to select documents strongly related through relation instances using partial match function and expand slowly on the set of instances.

We expected PartialMatch+ to perform better than PartialMatch I. However, PartialMatch+ achieved perfect recall a few iterations later than PartialMatch I. This is because PartialMatch+ used extracted instances to enlarge the set of extraction patterns, thus extracting more instances than PartialMatch I. However, these instances did not contribute much to the $Recall_{average}$ for this particular experiment.

Figure 4.4 shows the performance of the six document ranking strategies on selecting the relevant documents ($Recall_{rel.doc}$). We found that PartialMatch+ consistently performed well in selecting the relevant documents for the AEB-100 dataset. It obtained the same recall value as Ideal in the first 15 iterations. While $recall_{rel.doc}$ of other strategies failed to increase for a few of iterations indicating that they were not able to select relevant

4.4. EXPERIMENTAL RESULTS

Figure 4.3: Performance on Extracted Instances (manual coreferenced AEB-100)

Figure 4.4: Document ranking recall (manual coreferenced AEB-100) respect to relevant document

4.4. EXPERIMENTAL RESULTS

documents all the time. This is because some entity/relation instances of other criminal events were also extracted by the extraction patterns. Those instances were extracted because there were some domain specific action words contained in the documents, however those instances were not relevant to the beheading event. For example, PartialMatch I selected an irrelevant document in the 9th iteration. “*Terrorist_Harm(“gunman”, “rival”)*” was one of the relation instances extracted from the irrelevant document based on a sentence, “*an influential political power broker, to 25 years in jail on charges of hiring a **gunman** to kill a **rival***”. Because of the existence of documents describing other criminal events, it became more challenging for the document ranking strategies to select a relevant document.

As shown in Figure 4.5, PartialMatch I performed better than other document ranking strategies on selecting the optimal documents ($Recall_{opt.doc}$). It selected all optimal documents in the first 14 iterations, which was slightly better than PartialMatch+ (18th iteration) and PartialMatch II (18th iteration), and was much more superior than DiffCompare and CombineCompare since the latter could not find all optimal documents within 45 iterations. This also illustrated that PartialMatch+ was likely to select relevant documents than optimal documents and this was the reason PartialMatch+ reached perfect $Recall_{opt.doc}$ a bit later than PartialMatch I.

KSB-100: Figure 4.6 shows the $Recall_{average}$ for the KSB-100 dataset. PartialMatch I still outperformed other document ranking strategies and achieved perfect recall in the first 19 iterations for KSB-100 dataset. PartialMatch+ performed slightly worse than PartialMatch I and achieved perfect recall in the 22th iteration, followed by PartialMatch II (23th iteration). DiffCompare and CombineCompare came next and they could not achieve perfect recall within the first 45 iterations. Since the other performance results ($Recall_{rel.doc}$ and $Recall_{opt.doc}$) of the five document ranking strategies on KSB-100 were similar to those on AEB-100, we do not show the corresponding figures here.

AEB-1000: For the AEB-1000 dataset, PartialMatch+ achieved perfect $Recall_{average}$ in the 39th iteration as shown in Figure 4.7. This was followed by PartialMatch I and II as they achieved almost 90% $Recall_{average}$ in the first 45 iterations. DiffCompare and CombineCompare gave the worse $Recall_{average}$. In fact, PartialMatch+ outperformed all the other document ranking strategies in $Recall_{rel.doc}$ and $Recall_{opt.doc}$. Nevertheless, due to space constraints, we leave out these figures. On the whole, for AEB-1000

4.4. EXPERIMENTAL RESULTS

Figure 4.5: Document ranking recall (manual coreferenced AEB-100) respect to optimal set

Figure 4.6: Performance on Extracted Instances (manual coreferenced KSB-100)

4.4. EXPERIMENTAL RESULTS

Figure 4.7: Performance on Extracted Instances (manual coreferenced AEB-1000)

dataset, PartialMatch+ selected only 5% of the total number of documents in order to obtain all entity and relation instances. In contrast, CombineCompare could find only 70% instances by selecting 5% of all documents. This indicates that PartialMatch+ was able to scale up with the size of document collection better than the other strategies.

4.4.2 Datasets with Auto-Coreference Resolution

In this section, we present the experimental results of different document ranking strategies on the KSB-100 and AEB-100 datasets with auto-coreference resolution. We conducted experiments with the same seeds used in previous experiments to observe the performance impact of manual coreference resolution and auto-coreference resolution. We also conducted experiments with different choices of seeds in order to investigate the effects of difference seeds on the datasets with auto-coreference resolution.

KSB-100: Figure 4.8 shows the $Recall_{average}$ of KSB-100 dataset. PartialMatch I still achieved perfect recall with the smallest number of iterations (20th) for this dataset. PartialMatch+ was next and achieved perfect recall at the 23th iteration. PartialMatch

4.4. EXPERIMENTAL RESULTS

II performed slightly worse than PartialMatch+ and achieved perfect recall at the 24th iteration. DiffCompare was better than CombineCompare, but none of them could achieve perfect recall in the first 45 iterations. Figure 4.9 shows PartialMatch+ outperformed other strategies on selecting the relevant documents, and Figure 4.10 shows PartialMatch I found all optimal document earlier than other strategies.

Figure 4.8: Performance on Extracted Instances (auto-coreferenced KSB-100)

AEB-100: Figure 4.11 shows the $Recall_{average}$ for AEB-100 dataset. We observed that PartialMatch I, PartialMatch II and PartialMatch+ performed consistently better than CombineCompare and DiffCompare in the first 25 iterations. Only at the 43th iteration and beyond, CombineCompare and DiffCompare performed better than the rest and achieved perfect $Recall_{average}$. Figures 4.12 and 4.13 show that PartialMatch I, PartialMatch II and PartialMatch+ outperformed DiffCompare and CombineCompare in the first 25 iterations on both $Recall_{rel.doc}$ and $Recall_{opt.doc}$, which were similar to our previous observations.

4.4. EXPERIMENTAL RESULTS

Figure 4.9: Document ranking recall (auto-coreferenced KSB-100) respect to relevant document

Figure 4.10: Document ranking recall (auto-coreferenced KSB-100) respect to optimal set

4.4. EXPERIMENTAL RESULTS

Figure 4.11: Performance on Extracted Instances (auto-coreferenced AEB-100)

Figure 4.12: Document ranking recall (auto-coreferenced AEB-100) respect to relevant document

4.4. EXPERIMENTAL RESULTS

Figure 4.13: Document ranking recall (auto-coreferenced AEB-100) respect to optimal set

We conclude from results on datasets with auto-coreference resolution that the auto-coreference resolution affected the performance of different document ranking strategies adversely due to some wrongly coreferenced objects in the documents with auto-coreference resolution. However, the relative performances of different document ranking strategies remained the same. For KSB-100 dataset, PartialMatch I was able to achieve perfect $Recall_{average}$ at the 19th iteration with manual coreference resolution, which was 1 iteration earlier than the result of PartialMatch I on KSB-100 with auto-coreference resolution. However, the AEB-100 with auto-coreference resolution dataset gave a much worse result, in which DiffCompare (the strategy gave the best performance) achieved perfect $Recall_{average}$ at the 44th iteration. In contrast, PartialMatch I achieved perfect $Recall$ at the 14th iteration on the dataset with manual coreference resolution. Table 4.3 shows a summary on different datasets with the best performing strategies in different evaluations. The number shown in each bracket indicates the number of iterations required for the document ranking strategy to achieve perfect recall with the particular evaluation metric. Since all document ranking strategies could not achieve perfect

4.4. EXPERIMENTAL RESULTS

$Recall_{rel_doc}$ for all datasets in the first 45 iterations, the percentage of $Recall_{rel_doc}$ obtained by individual strategy in the first 45 iterations is then shown in the bracket.

Table 4.3: Summary of winning strategies on different datasets

Metrics	Manual Coreference Resolution		Auto Coreference Resolution	
	KSB-100	AEB-100	KSB-100	AEB-100
$Recall_{average}$	PartialMatch I (19)	PartialMatch I (14)	PartialMatch I (20)	DiffCompare (44)
$Recall_{rel_doc}$	PartialMatch II (68%)	PartialMatch+ (61%)	PartialMatch+ (68%)	DiffCompare (62%)
$Recall_{opt_doc}$	PartialMatch I (19)	PartialMatch I (14)	PartialMatch I (20)	DiffCompare (44)

As shown in Table 4.3, PartialMatch series ranking strategies (PartialMatch I, II and Plus) outperformed other ranking strategies in all datasets except auto-coreferenced AEB-100. In Section 3.2, we have defined that relevant relation instances either contain the seeds or there must be a path linking the relation instance to some relation instance which contains the seed(s). We have also defined that the optimal set containing all relevant entity and relation instances. PartialMatch based ranking strategies are good at finding the optimal set in most cases. The only exception happened with auto-coreference AEB-100 dataset, in which DiffCompare outperformed all PartialMatch based strategies. This was because of the incompleteness and inaccuracy of the auto-coreference process. As a result, some relation instances are unavailable to PartialMatch based strategies.

Choice of seeds: In the following, we show the experimental results of five document ranking strategies with different choices of seeds. We chose the following seeds in the experiments, in which the :

- Randomly picked up 2 most frequent entity instances as seeds (MF2)
- Randomly picked up 3 most frequent entity instances as seeds (MF3)
- Randomly picked up 4 most frequent entity instances as seeds (MF4)
- Randomly picked up 2 least frequent entity instances as seeds (LF2)
- Randomly picked up 3 least frequent entity instances as seeds (LF3)
- Randomly picked up 4 least frequent entity instances as seeds (LF4)

4.4. EXPERIMENTAL RESULTS

We expected that with the most frequent entity instances as seeds, the different strategies shall be able to take fewer iterations to extract all the instances and find all the optimal documents than using the least frequent entity instances as seeds.

Figures 4.14 to 4.16 show the performance of different ranking strategies on the KSB-100 dataset with MF2. PartialMatch I was the best among other strategies, as it achieved perfect $Recall_{average}$ at the 20th iteration and also found all the optimal documents. PartialMatch+ and PartialMatch II outperformed the other strategies on selecting the relevant documents. DiffCompare and CombineCompare gave the worst performance in all evaluations. The results of the five document ranking strategies with MF3 and MF4 on KSB-100 were similar to the results with MF2. Therefore, we leave out the figures here.

Figures 4.17 to 4.19 show the performance with LF2 as seeds. PartialMatch II achieved perfect $Recall_{average}$ at the 25th iteration, while other document ranking strategies could not achieve perfect recall within the first 45 iterations. However, we noticed that the performance of DiffCompare, CombineCompare, PartialMatch I and PartialMatch+ were quite close to perfect recall (above 95%) at the 45th iteration. PartialMatch II outperformed the other strategies on selecting the relevant documents, followed by PartialMatch+. It also outperformed the other strategies on finding the optimal documents. PartialMatch II considers both accumulated user annotated entity instances and seeds, therefore it is more likely to pick up those documents containing the entity instances but not seeds. The results of the five document ranking strategies with LF3 and LF4 on KSB-100 were similar to the results with MF2. Therefore, we have left out the figures here.

The performance of the five document ranking strategies on AEB-100 with different choices of seeds were found to be similar to those on KSB-100. We conclude that using different choices of seeds with the similar frequency will not affect the performance of the document ranking strategies very much. However, using the least frequent entity instances as seeds affected the performance of different strategies adversely compared to those using most frequent entity instances as seeds.

4.4. EXPERIMENTAL RESULTS

Figure 4.14: Performance on Extracted Instances (auto-coreferenced KSB-100 with MF2)

Figure 4.15: Document ranking recall (auto-coreferenced KSB-100 with MF2) respect to relevant document

4.4. EXPERIMENTAL RESULTS

Figure 4.16: Document ranking recall (auto-coreferenced KSB-100 with MF2) respect to optimal set

Figure 4.17: Performance on Extracted Instances (auto-coreferenced KSB-100 with LF2)

4.4. EXPERIMENTAL RESULTS

Figure 4.18: Document ranking recall (auto-coreferenced KSB-100 with LF2) respect to relevant document

Figure 4.19: Document ranking recall (auto-coreferenced KSB-100 with LF2) respect to optimal set

4.5. COMPARISONS BETWEEN PATTEN-BASED VS CONTENT-BASED DOCUMENT RANKING STRATEGIES

4.5 Comparisons between Patten-based vs Content-based Document Ranking Strategies

COMPARE and EXTRACTCOMPARE are two document ranking strategies based on document content described in [16] (see Chapter 3). Originally they were developed for identifying documents to be manually labelled by an expert so as to learn some extraction patterns from them. Since PartialMatch I usually performs better than the other four document ranking strategies, we now try to use these two content-based document ranking strategies for the extraction task and compare them with PartialMatch I.

As shown in Figure 4.20, PartialMatch I outperformed COMPARE and EXTRACTCOMPARE in $Recall_{average}$. PartialMatch I achieved perfect recall on the 14th iteration. In the contrast, EXTRACTCOMPARE could only achieve 57% $Recall_{average}$ in the first 45 iterations, followed by COMPARE ($Recall_{average} = 48\%$).

Figures 4.21 and 4.22 show that COMPARE and EXTRACTCOMPARE could find only half the number of relevant and optimal documents compared to PartialMatch I in the first 45 iterations. We therefore conclude that PartialMatch I outperforms the two content-based strategies on the proposed domain specific event topic entity and relation extraction task. We obtained the same finding for the KSB-100 dataset.

4.5. COMPARISONS BETWEEN PATTEN-BASED VS CONTENT-BASED DOCUMENT RANKING STRATEGIES

Figure 4.20: Performance evaluation respect to extracted instances

Figure 4.21: Performance evaluation respect to relevant documents

4.5. COMPARISONS BETWEEN PATTEN-BASED VS CONTENT-BASED DOCUMENT RANKING STRATEGIES

Figure 4.22: Performance evaluation respect to optimal documents

Chapter 5

Personalized Event Topic Information Extraction

In the discussions covered by Chapters 3 and 4, we have so far assumed a fixed set of entity and relation classes and a set of extraction patterns for extracting instances. In the extraction task, we have also assumed that a user will annotate each document selected all possible instances of the set of entity/relation classes related to an event topic.

In this chapter, we relax the above assumptions and propose a personalized version of domain specific event topic entity relation extraction task. We develop a weighting scheme for entity and relation classes and incorporate their weights into the PartialMatch II document ranking strategy. Some experiments on the personalized document ranking strategy will be described and its comparison with the other document ranking strategies on this personalized extraction task will also be given.

5.1 Personalized Weight-based Document Ranking

We have earlier assumed that the user is interested in all entity and relation instances of a fixed set of classes of a specific event topic. However, in practice, the user may only be interested in a subset of entity and relation instances of the event topic. Since the user is only interested in a subset of entity and relation instances, it is likely that some of the given entity or relation classes have no instances the user wants and they are known as the *irrelevant classes*. For a relevant class, the user may require fewer instances than all that can be found for the class related to the event topic. The proportion of

5.1. PERSONALIZED WEIGHT-BASED DOCUMENT RANKING

required instances over all instances of the class that can be found somewhat indicates the importance of the class. The user required entity and relation instances and their corresponding relevant entity and relation classes are user specific. Different users may have therefore different requirements on the subset of entity and relation instances for the same event topic. To cope with different user requirements, we introduce a *domain specific personalized event topic entity relation extraction task*.

Consider the entity relation graph example shown in Figure 5.1, which is reproduced from Figure 4.1. For a user who would like to know the terrorists and terrorist organizations who killed “Kim Sun”, as well as those victims, a subset of these instances can be included in the required entity relation graph for the user as shown in a blob in Figure 5.1. The extraction of this subset of instances is known as the domain specific personalized event topic entity relation extraction task.

In the personalized event topic entity relation extraction task, a user will annotate in the way similar to the non-personalized task a selected document for some entity and relation instances. The objective of the extraction task is to find the smallest subset of documents which contain all instances interested to the user from the relevant classes.

The number of user required instances of a certain class indicates the importance of the class in the personalized extraction task. If we know exactly how many instances of each class the user is interested, we will be able to determine the importance of all classes. We choose to use some weight to represent the importance of each class. Generally, each entity class and relation class is assigned a weight to represent their importance of user interest.

Let AE_i denote the number of user interested entity instances of entity class E_i for an event topic, and AR_i denote the number of user interested relation instances of relation class R_i . The total number of entity instances of entity class E_i which can be found related to the event topic is denoted as TE_i . TR_i is denoted in the same manner as TE_i . Let $w(E_i)$ and $w(R_i)$ denote the weights of entity class E_i and R_i respectively. Assuming the subset of instances shown in Figure 5.1 is given, the ideal weight of each class involved in the subset can be calculated as following:

5.1. PERSONALIZED WEIGHT-BASED DOCUMENT RANKING

Figure 5.1: Personalized Entity Relation Graph of KSB event

5.1. PERSONALIZED WEIGHT-BASED DOCUMENT RANKING

1. $w(E_i) = \frac{AE_i}{TE_i}$ and $w(R_i) = \frac{AR_i}{TR_i}$
2. $w(E_i) = \frac{w(E_i)}{\sum_{j=1}^{|E|} w(E_j)}$ and $w(R_i) = \frac{w(R_i)}{\sum_{j=1}^{|R|} w(R_j)}$

Tables 5.1 and 5.2 show the ideal weights of the entity and relation classes of interest to the user respectively. For those irrelevant entity/relation classes to the user, their weights are simply 0.

Table 5.1: Ideal weight of each entity class

	Terrorist	Terrorist Org	Victim
Weights	0.13	0.22	0.65

Table 5.2: Ideal weight of each relation class

	Terrorist_Harm	Org_Harm
Weights	0.5	0.5

Suppose the weight of each class is already known, we propose an extended document ranking strategy by incorporating the weight of each class with any of the previously proposed document ranking strategies. In the experiment shown in Chapter 4, Partial-Match II gave a comparative good performance among the other strategies. In addition, we would like to consider both the importance of entity and relation classes in document ranking. Thus, we choose PartialMatch II to be incorporated with the weight of each class and this new strategy is known as *personalized partial match (PPM) document ranking strategy*.

Given two relations $r_s = (e_s^s, a_s, e_s^t)$ and $r_t = (e_t^s, a_t, e_t^t)$, the personalized partial match of r_s and r_t denoted by $PPM(r_s, r_t)$ is defined by:

$$PPM = \begin{cases} 0 & \text{if } e_s^s \neq e_t^s \wedge e_s^t \neq e_t^t \\ 0 & \text{if } e_s^s = e_t^s \wedge e_s^t = e_t^t \wedge (r_s \in R_p, r_t \in R_q, p \neq q) \\ w(R_p) & \text{otherwise} \end{cases}$$

PPM actually measures the novelty of relations respect to their weights compared to partial match function shown in Section 3.3.4 in which the weight of each class is equal and is simply assigned as 1 upon a successful match. The weights of different classes are

5.2. PERSONALIZED EVENT TOPIC ENTITY RELATION EXTRACTION

used in defining the personalized partial match ranking strategy as shown in following Equation:

$$score(d_j) = \alpha \cdot \frac{\sum_{k=1}^{|\mathcal{R}'_j|} \sum_{h=1}^{|\mathcal{R}|} PPM(r'_{j,k}, r_h)}{\sum_{k=1}^{|\mathcal{R}'_j|} w(R_{j,k}) \cdot |\mathcal{R}| + 1} + (1 - \alpha) \cdot (\lambda \cdot \frac{\sum_{k=1}^{|\mathcal{E}'_j \cap \mathcal{E}|} w(E_k)}{\sum_{h=1}^{|\mathcal{E}|} w(E_h)} + (1 - \lambda) \cdot \frac{\sum_{k=1}^{|\mathcal{E}'_j \cap \mathcal{W}'|} w(E_k)}{\sum_{h=1}^{|\mathcal{W}'|} w(E_h)}) \quad (5.1)$$

where $\mathcal{R}'_j$ represents the set of relation instances extracted from document d_j ; $r'_{j,k}$ is the k th relation instance from $\mathcal{R}'_j$; r_h is the h th relation instance in $\mathcal{R}$; And $R_{j,k}$ is the relation class of the extracted relation instance $r'_{j,k}$.

5.2 Personalized Event Topic Entity Relation Extraction

In practice, it is impossible to know the importance of each class beforehand. Therefore, we have to estimate the importance of each class in the extraction task. We propose a *weighting scheme* to present the estimated importance of each class in the extraction task. Briefly, each class will be assigned with a weight. The weight of each class will be adjusted according to user feedback to represent estimated importance of the class. This weighting scheme is incorporated in the personalized event topic extraction task.

Algorithm 2 gives a detailed explanation on the personalized event topic entity relation extraction task. There are three differences between the personalized extraction algorithm and the previous extraction algorithm. Before the personalized extraction starts, the weight of each class is initialized. Then, after each iteration, the weight of every class has to be adjusted according to user's annotation before the remaining documents are re-scored using some document ranking strategy. Lastly, in order to utilize the weighting scheme into the personalized extraction task, the PPM strategy is chosen to score on each document with the estimated weight of each class.

In the weighting scheme, every entity class is assigned the same weight initially and the summation of weights is equal to 1, i.e., $w(E_1) = w(E_2) \dots = w(E_n) = \frac{1}{|E|}$. Similarly, every relation class has the same weight initially and the summation of weights is equal to 1, i.e., $w(R_1) = w(R_2) \dots = w(R_m) = \frac{1}{|R|}$. The weights of all classes will be adjusted according to user's annotation on a selected document after each iteration.

5.2. PERSONALIZED EVENT TOPIC ENTITY RELATION EXTRACTION

Algorithm 2 Personalized Event Topic Entity Relation Extraction

```

1: input:  $EP, D, W$ 
2: Initialize the weight of each class
3: for each document  $d_j$  in  $D$  do
4:   apply  $EP$  on  $d_j$  to obtain  $\mathcal{E}'_j, \mathcal{R}'_j$ 
5:   compute  $\text{score}(d_j)$  using initial score function (see Section 3.3.1)
6: end for
7: repeat
8:   //User selects a document  $d_s$ 
9:   Move  $d_s$  from  $D$  to  $S$ 
10:  //User annotates  $\mathcal{E}_s$  and  $\mathcal{R}_s$  in  $d_s$ 
11:  Add  $\mathcal{E}_s$  and  $\mathcal{R}_s$  to  $\mathcal{E}$  and  $\mathcal{R}$  respectively
12:  Adjust the weight of each class based on  $\mathcal{E}'_j, \mathcal{R}'_j, \mathcal{E}, \mathcal{R}$ 
13:  for each document  $d_j$  in  $D$  do
14:    compute  $\text{score}(d_j)$  based on  $\mathcal{E}'_j, \mathcal{R}'_j, \mathcal{E}, \mathcal{R}$  (using personalized partial match shown in Section 5.1)
15:  end for
16: until termination condition is satisfied
17: output:  $S, \mathcal{E}, \mathcal{R}$ 

```

Let $\mathbb{E}_i$ denote the set of extracted and user annotated instances of the entity class E_i as some iteration. Similarly, Let $\mathbb{R}_i$ denote the number of extracted and user annotated instances of the relation class R_i at some iteration. Let NE_i and NR_i denote the number of entity and relation instances annotated by user through all previous iterations separately.

For each entity class E_i in E , the adjustment of $w(E_i)$ is performed as Equation 5.2:

$$w(E_i)' = \gamma \cdot w(E_i) + (1 - \gamma) \cdot w(E_i) \cdot \frac{NE_i}{|\mathbb{E}_i|} \quad (5.2)$$

where γ is used as a parameter to control the amount of weight to be retained for each relation class after the adjustment. The weight is always adjusted by adding the retained weights with the importance of the certain classes determined in the previous iteration. The proportion of annotated instances over all instances presents the importance of a certain class in the iteration. After all $w(E_i)'$ s have been calculated for this iteration, they will be normalized by the summation of new weights, i.e., $w(E_i) = \frac{w(E_i)'}{\sum_{j=1}^{|E|} w(E_j)'}$. The same adjustment will be applied to the weights of relation classes as well.

5.3. EXPERIMENT SETUP

It is possible that some entity instances extracted by the entity extraction system components cannot be assigned some pre-defined entity classes. We therefore introduce a special *Exception* entity class for these entity instances. The weight of the *Exception* entity class is derived in a way similar to other entity classes.

5.3 Experiment Setup

KSB-100 and AEB-100 datasets are chosen for this experiment. To simulate an user only interested in a subset of entity and relation instances for a particular event topic, we modified the annotations of two datasets as follows:

For each dataset, we constructed two different sets of annotations, each consisting of around 50% of all instances randomly chosen from the original annotations. We named them KSB Case 1, KSB Case 2, AEB Case 1 and AEB Case 2. Each set of annotations involves only a subset of entity and relation classes compared to the original annotations. The detailed information about each set of annotations is shown in Table 5.3.

Table 5.3: Detailed Information of four sets of annotations on KSB-100 and AEB-100

	$ \mathcal{E} $	$ \mathcal{R} $	# of Rel docs	# of Opt docs	Entity classes involved	Relation classes involved
KSB Case 1	9	8	31	6	Victim Terrorist Terrorist Org Employer	Work_for Leader_of Member_of Connect_to Org_Harm
KSB Case 2	10	11	10	6	Victim Terrorist Terrorist Org Event Relative	Leader_of Member_of Terrorist.Resp_for Terrorist.Harm Org_Harm Relative_of
AEB Case 1	11	9	24	6	Terrorist Terrorist Org Relative Victim	Connect_to Member_of Leader_of Relative_of
AEB Case 2	10	11	17	5	Terrorist Terrorist Org Victim Location Event	Terrorist.Harm Org.Resp_for Terrorist.Resp_for Happen

Figures 5.2 to 5.5 show the entity relation graph of each set of annotations. We chose the two most frequent entity instances in each set of annotations as the seeds. As mentioned in Section 3.2, it is necessary for all entity instances in the entity relation graph to be directly or indirectly linked to one of the entity instances in the seeds through

5.4. EXPERIMENTAL RESULTS

the relation instances. Therefore, we also chose “Elisabeth” as the seeds in AEB Case 1 to fulfill this requirement.

We have applied PartialMatch I, PartialMatch II and PPM on the four datasets. Their performance was measured by $Recall_{average}$, $Recall_{rel.doc}$ and $Recall_{opt.doc}$ (see Section 4.3.3). KSB-100 and AEB-100 datasets with auto-coreference resolution were used in this experiment. In our experiments, we set $\alpha = 0.6$ and $\lambda = 0.5$ to make them consistency with the experiments shown in Chapter 4. For Personalized PM, we set $\gamma = 0.1$ to give a smaller amount of weight retained for each class. This is because we want to favor weight changes of once there are some instances annotated by the user.

5.4 Experimental Results

5.4.1 KSB Case 1

Figure 5.6 shows the $Recall_{average}$ of KSB Case 1. PPM outperformed the other strategies as it achieved perfect recall with the smallest number of iterations (20th iteration). PartialMatch II was the runner-up (22th iteration), followed by PartialMatch I (26th iteration). Ideal strategy (6th iteration) showed the ideal performance in that cannot be achieved in normal circumstances. Figure 5.7 shows PPM found all optimal documents 2 iterations earlier than PartialMatch II and 6 iterations earlier than PartialMatch I. PPM did not match PartialMatch I and PartialMatch II on selecting relevant documents as shown in Figure 5.8. This is because the relevant documents also contain irrelevant entity and relation instances which are partially linked to the relevant instances annotated by the user. As PartialMatch I and II treat the instances of all classes equally, those relevant documents will be given higher scores as they contain more related and novel instances. However, PPM only focuses on instances from the relevant classes. Therefore, once all entity and relation instances were found by PPM, it was less likely to select the remaining relevant documents compared to PartialMatch I and II.

5.4.2 KSB Case 2

Figures 5.9 to 5.11 show that PPM outperformed PartialMatch I and PartialMatch II on KSB Case 2 dataset. It achieved perfect recall of $Recall_{average}$ and $Recall_{opt.doc}$ in

5.4. EXPERIMENTAL RESULTS

Figure 5.2: Entity and Relation Instances of KSB Case 1

Figure 5.3: Entity and Relation Instances of KSB Case 2

5.4. EXPERIMENTAL RESULTS

Figure 5.4: Entity and Relation Instances of AEB Case 1

Figure 5.5: Entity and Relation Instances of AEB Case 2

5.4. EXPERIMENTAL RESULTS

Figure 5.6: Performance on Extracted Instances (KSB Case 1)

Figure 5.7: Document selection recall (KSB Case 1) respect to optimal set

5.4. EXPERIMENTAL RESULTS

Figure 5.8: Document selection recall (KSB Case 1) respect to relevant document

Figure 5.9: Performance on Extracted Instances (KSB Case 2)

5.4. EXPERIMENTAL RESULTS

the 24th iteration, while PartialMatch I and PartialMatch II required 36 iterations to achieve perfect recall. In KSB Case 2, PPM also performed better than PartialMatch I and II on selecting relevant documents as shown in Figure 5.10. This was because of the small amount of relevant documents in KSB Case 2. When PPM selected the last optimal document, it selected all relevant documents as well.

5.4.3 AEB Case 1

Figure 5.12 shows PPM gave perfect recall on extracted instances much earlier than PartialMatch I and PartialMatch II on the AEB case 1 dataset. PPM achieved perfect $Recall_{average}$ at 28th iteration, while PartialMatch I and PartialMatch II could not achieve perfect $Recall_{average}$ within the first 45 iterations. Since the other results on AEB case 1 dataset of all document ranking strategies were similar as those on KSB case 2, we leave out the figures here. PPM outperformed both PartialMatch I and PartialMatch II on all evaluations, followed by PartialMatch II and PartialMatch I.

5.4.4 AEB Case 2

Figure 5.13 shows PPM achieved perfect $Recall_{average}$ in the first 14 iterations on AEB case 2. This was also the best performance of PPM among all four different sets of annotations. Figure 5.14 shows PPM found more relevant documents than PartialMatch I and PartialMatch II in the first 45 iterations. It was able to find at least 80% of the relevant documents, while PartialMatch I and PartialMatch II only found 70% of the relevant documents. PPM outperformed the other two strategies on selecting the optimal documents shown in Figure 5.15. It found all optimal documents at the 14th iteration, while the other two strategies were only able to find 80% of all optimal documents.

Although PPM obtained best performance on AEB Case 2 among all four sets of annotations, it still performed worse than the Ideal strategy. Ideal strategy found all instances and optimal documents at the 6th iteration. We later conducted a detailed investigation on the ideal weights and estimated weights using PPM on AEB Case 2. The Ideal weight of each class was calculated based on the annotation, which was to assume we knew everything beforehand. Table 5.4 shows the ideal weight of each entity

5.4. EXPERIMENTAL RESULTS

Figure 5.10: Document selection recall (KSB Case 2) respect to relevant document

Figure 5.11: Document selection recall (KSB Case 2) respect to optimal set

5.4. EXPERIMENTAL RESULTS

Figure 5.12: Performance on Extracted Instances (AEB Case 1)

class involved in AEB Case 2. Table 5.5 shows the ideal weight of each relation class involved in AEB Case 2. For classes not involved in AEB Case 2, their weights are 0.

We recorded the weight of each class (including relevant and irrelevant classes) after each iteration in the experiment. Tables 5.6 to 5.7 show the estimated weights of each entity and relation classes in the experiment on AEB Case 2. Initially, each entity class has a weight of 0.125 and the weight of each relation class is 0.1.

Table 5.4: Ideal weight of each entity class

	Terrorist	Terrorist Org	Victim	Location	Event
Weight	0.143	0.143	0.143	0.256	0.256

Table 5.5: Ideal weight of each relation class

	Terrorist_Harm	Org_Resp_for	Terrorist_Resp_for	Happen
Weight	0.25	0.25	0.25	0.25

We observe from Table 5.6 that the estimated weight of entity class *Event* (0.4266) and *Location* (0.4266) are quite correct comparing to the ideal weights in Table 5.4. The

5.4. EXPERIMENTAL RESULTS

Figure 5.13: Performance on Extracted Instances (AEB Case 2)

Figure 5.14: Document selection recall (AEB Case 2) respect to relevant document

5.4. EXPERIMENTAL RESULTS

Figure 5.15: Document selection recall (AEB Case 2) respect to optimal set

remaining three entity classes which are involved in AEB Case 2 are supposed to be given higher weights compared to the irrelevant classes. However, they were given much lower weights compared to the ideal weights in the experiment. This was because the number of wrong extracted entity instances of these three classes was large and the number of annotated entity instances was relatively small, i.e., $|\mathcal{E}_i| \gg NE_i$. Therefore, the weights of these classes reduced quickly with more iterations. However, the weights of these three classes could still increase once there were some annotated instances found by the user. For example, the weight of *Terrorist* class increased from 0.1497 to 0.1509 from the 3th iteration to the 4th iteration.

The weight of *Employer* class did not change throughout the first 14 iterations. This was because there was no entity instance extracted using the extraction patterns, nor instances annotated from the selected documents. Therefore, there was no evidence on the class whether it was relevant to the user or not.

Table 5.7 shows the estimated weights of the relevant relation classes. The weight of *Happen* and *Org_Resp_for* classes were almost the same compared to the corresponding

5.4. EXPERIMENTAL RESULTS

ideal weights. The other two relevant relation classes *Terrorist_Harm* (0.133) and *Terrorist_Resp_for* (0.143) were assigned weights slight smaller compared to their ideal weights (0.25). The irrelevant classes (except *Work* and *Member_of*) decreased with the number of iterations increasing.

5.4.5 Summary of Results

We conclude that the proposed personalized partial match document ranking strategy based on weighting scheme performs well on this personalized event topic extraction task. PPM is shown to find all user interested entity and relation instances, and optimal documents sooner than PartialMatch I and PartialMatch II. The weighting scheme is also able to give fairly accurate estimated weights to the user interested entity and relation classes.

5.4. EXPERIMENTAL RESULTS

Table 5.6: Estimated weight of each entity class

Iterations	Victim	Terrorist	Terrorist Org	Location
1	0.0247525	0.1250	0.080445545	0.2475
2	0.002867	0.1448	0.02608945	0.2867
3	0.0002965	0.1497	0.006744738	0.2965
4	7.47E-05	0.1509	0.001291752	0.2988
5	1.87E-05	0.1511	0.000245803	0.2993
6	4.90E-06	0.1227	4.89E-05	0.3136
7	1.16E-06	0.0988	9.24E-06	0.3256
8	2.75E-07	0.0905	1.91E-06	0.3849
9	5.72E-08	0.0729	3.48E-07	0.4003
10	1.02E-08	0.0577	6.21E-08	0.4086
11	1.81E-09	0.0454	1.10E-08	0.4148
12	3.21E-10	0.0356	1.95E-09	0.4197
13	5.68E-11	0.0278	3.45E-10	0.4236
14	1.00E-11	0.0217	5.71E-11	0.4266
Iterations	Relative	Employer	Event	Exception
1	0.125	0.125	0.2475	0.024752475
2	0.125	0.125	0.2867	0.002866972
3	0.125	0.125	0.2965	0.000296472
4	0.125	0.125	0.2988	2.99E-05
5	0.125	0.125	0.2993	2.99E-06
6	0.125	0.125	0.3136	3.14E-07
7	0.125	0.125	0.3256	3.26E-08
8	0.014775	0.125	0.3849	3.85E-09
9	0.0015366	0.125	0.4003	4.00E-10
10	0.0001568	0.125	0.4086	4.09E-11
11	1.59E-05	0.125	0.4148	4.15E-12
12	1.61E-06	0.125	0.4197	4.20E-13
13	1.63E-07	0.125	0.4236	4.24E-14
14	1.64E-08	0.125	0.4266	4.27E-15

5.4. EXPERIMENTAL RESULTS

Table 5.7: Estimated weight of each relation class

Iterations	Connect	Org_Harm	Terrorist_Harm	Org_Resp_for	Terrorist_Resp_for
1	0.018181818	0.01818182	0.1	0.181818182	0.1
2	0.001945525	0.00194553	0.1	0.194552529	0.107003891
3	0.000195925	0.00019592	0.1	0.195924765	0.107758621
4	1.96E-05	1.96E-05	0.1000588	0.196039992	0.107821996
5	1.96E-06	1.96E-06	0.1000647	0.196051522	0.107828337
6	2.25E-07	2.25E-07	0.114829	0.224978458	0.123738152
7	2.28E-08	2.28E-08	0.1165487	0.228347674	0.125591221
8	2.58E-09	2.58E-09	0.1315166	0.257673604	0.141720482
9	2.61E-10	2.61E-10	0.1332276	0.261025872	0.14356423
10	2.61E-11	2.61E-11	0.1334012	0.261365902	0.143751246
11	2.61E-12	2.61E-12	0.1334185	0.261399954	0.143769975
12	2.61E-13	2.61E-13	0.1334203	0.26140336	0.143771848
13	2.61E-14	2.61E-14	0.1334204	0.2614037	0.143772035
14	2.61E-15	2.61E-15	0.1334205	0.261403734	0.143772054
Iterations	Work	Member_of	Leader_of	Relative_of	Happen
1	0.1	0.1	0.1	0.1	0.181818182
2	0.1	0.1	0.1	0.1	0.194552529
3	0.1	0.1	0.1	0.1	0.195924765
4	0.1	0.1	0.1	0.1	0.196039992
5	0.1	0.1	0.1	0.1	0.196051522
6	0.1	0.1	0.0114755	0.1	0.224978458
7	0.1	0.1	0.0011647	0.1	0.228347674
8	0.1	0.1	0.0001314	0.011284267	0.257673604
9	0.1	0.1	1.33E-05	0.001143107	0.261025872
10	0.1	0.1	1.33E-06	0.00011446	0.261365902
11	0.1	0.1	1.33E-07	1.14E-05	0.261399954
12	0.1	0.1	1.33E-08	1.14E-06	0.26140336
13	0.1	0.1	1.33E-09	1.14E-07	0.2614037
14	0.1	0.1	1.33E-10	1.14E-08	0.261403734

Chapter 6

INEXT: A Tool for Event Topic Information Extraction

In this chapter, we introduce a working prototype system called tool for event topic INformation EXtraction (INEXT). INEXT was developed to carry out domain specific event topic entity and relation extraction task and it implemented our proposed document ranking strategies. We briefly describe the system architecture of INEXT and the main user interface design.

6.1 System Architecture

Figure 6.1 depicts the overall system architecture of INEXT. It assumes the existence of a collection of documents on which event topic information extraction is to be conducted. At present, only documents containing plain text can be handled.

INEXT consists of a *named entity recognition (NER) module* that extracts named entities from text documents. These named entities are domain independent and they include person names, organisation names, place names and others that can appear in documents. The extracted named entities are stored in a database which are specially designed for storing extracted objects. Readers can refer to Appendix A.1 for our implementation of the NER module.

The *coreference resolution (CR) module* is responsible for matching semantically equivalent pronouns and named entities found within the documents. This is necessary as the same named entity (e.g., person name) may be referenced multiple times

6.1. SYSTEM ARCHITECTURE

Figure 6.1: INEXT System Architecture

either as pronouns or other named entities. Given a document, once a named entity is identified to be the antecedent of a given pronoun or a given named entity, the given pronoun or named entity is then replaced by the identified named entity. We propose our own name coreference resolution method in CR module. Readers can refer to Appendix A.2 for a detailed description.

The extraction of domain specific entity and relation instances is performed by the *domain entity and relation extraction (DERE)* module. It relies on multi-slot extraction patterns to assign semantic classes to entities and to determine relationships among the extracted entities respectively. We have incorporated the Badger IE system component (see Section 4.1.2) into INEXT to perform the extraction of domain specific entity and relation instances.

The *document ranking (DR)* module provides the core document ranking strategies required by event topic information extraction tasks. The idea here is to use the entity seeds as well as the extracted domain specific entity and relation instances to compute document relevance scores. The highly ranked documents are expected to contain many domain specific named entities and relations that have not yet been examined by the user yet. PartialMatch I gave the best performance on selecting optimal documents and finding instances in our earlier experiments (refer to Section 4.4). PartialMatch+, our another proposed document ranking strategy, did well on a large scale documents

6.2. DOCUMENT RANKING MODULE

collection. Therefore, we have implemented these two document ranking strategies in INEXT.

The user can manually annotate one or more highly ranked documents using the *entity and relation annotation (ERA) module*. The annotation module allows relevant entity and relation instances in the document to be marked up by the user and added to the database for storing annotated objects. These annotated objects can be further used for learning new extraction patterns by domain entity and relation extraction module (see Section 4.3.2).

As illustrated in Figure 6.1, both the named entity recognition and coreference resolution modules process documents only once. On the other hand, the domain entity and relation extraction, document ranking, and entity and relation annotation modules are invoked whenever a query (entity seeds) is evaluated. Furthermore, the three-step process involving these three modules may iterate until the user decides to end the extraction process, or no other relevant documents can be found, or no new entities and relations can be extracted.

Visual C#.NET is used for the implementation of INEXT. The .NET framework [4] provides useful libraries and components to facilitate the user interface design and the software development. In the following two sections, we will elaborate the user interface design of DR and ERA modules. The other modules run in the back end and do not involve user interface.

6.2 Document Ranking Module

To begin with an event topic information extraction task on a document collection, a user is required to provide an initial set of seed entities related to an event topic of interest.

Figure 6.2 illustrates the query window of INEXT. There are four panels in the document ranking module. The initial setting panel shows the file directory of the given document collection and allows the user to key in some seed entities as query. The document ranking panel lists the result documents, their relevance scores determined by some ranking strategies and some document snippets. The two panels on the right of Figure 6.2 show the sets of entities and relations already annotated by the user respectively.

6.2. DOCUMENT RANKING MODULE

Suppose that INEXT has been given a document collection consisting of news articles, and a user provides some seed entities to the initial setting panel. Figure 6.2 shows that the given seeds are “Al-Qaeda Cell”, “Kim Sun” and “Beheading of Kim Sun”. INEXT will then rank the documents according to the *initial score function* (see Section 3.3.1). Figure 6.2 shows the documents in decreasing scores together with their snippets. In the document snippets, seeds found are marked up by ‘<’s and ‘>’s for easy identification.

Figure 6.2: Query Interface

Documents can be selected from the document ranking panel and be annotated by the user. A detailed description of the annotation module can be found in Section 6.3.

After some new entity and relation instances from a selected resultant document are annotated, our document ranking module will revise the document scores by applying either PartialMatch I or PartialMatch+ according to the user’s choice. In fact, Partial-Match I and PartialMatch+ will be used for the second and subsequent iterations. The user can choose either PartialMatch I or PartialMatch+ by clicking the “Tool” button in the menu bar of this query interface. Figure 6.3 shows the document ranking strategies selection interface. The user has to configure the document ranking strategy before he/she performs document ranking in the subsequent iterations. By default, Partial-Match I is chosen and α is set to 0.6. The personalized partial match strategy has not been implemented in INEXT due to the time constraints.

6.3. ENTITY AND RELATION ANNOTATION MODULE

Figure 6.3: Document Ranking Strategies Configuration Interface

6.3 Entity and Relation Annotation Module

When one of the highly ranked documents is selected, the entity and relation annotation module will be activated allowing the user to annotate the selected document. Figure 6.4 depicts the annotation user interface where the selected document is displayed in the left panel. As shown in Figure 6.4, document 011.txt which was given the highest score by the initial score function has been selected for annotation.

Within the document panel, the domain entities extracted by INEXT are highlighted in yellow allowing the user to easily recognize potentially relevant entity instances. The domain relations extracted by INEXT are shown in a list box on the right hand side and they are highlighted in yellow to indicate that these relation instances are extracted but not annotated by the user yet. The user can also select any names from the document by highlighting and annotating them as relevant entity instances. Although we expect most of these names to be among those extracted by INEXT, they are not necessary to be so since it is possible that some relevant entity instances may not be extracted by our extraction patterns.

Once annotated, the entity instances will be highlighted in pink in the document panel to distinguish them from the ones extracted by INEXT. The user can further assign (and also update) entity classes to the annotated entity instances within the annotation interface (see the entity/relation annotation panel in Figure 6.4). Within the annotation interface, the relation instances extracted are shown in the list box on the right. Those extracted but not annotated relation instances are highlighted in yellow while those annotated relation instances are highlighted in pink. The user can annotate a relation

6.3. ENTITY AND RELATION ANNOTATION MODULE

Figure 6.4: User Annotation Interface

instance by choosing the relation class, source entity instance and target entity instance from the entity/relation annotation panel.

Note that all user annotated entity and relation instances for the event topic information extraction task are maintained throughout the extraction session and are also displayed in both the query interface and user annotation interface for reference.

Once the user completes the annotation of the selected document, he or she can return to the document ranking interface giving the annotated entity and relation instances as feedback to the document ranking module. In this example, we suppose PartialMatch I has been chosen for document ranking.

As shown in Figure 6.5, the previously annotated document, i.e., 011.txt, is now given a smaller score as it does not contain many entity or relation instances which are not already seen (or annotated) by the user. Instead, some other documents are now given

6.3. ENTITY AND RELATION ANNOTATION MODULE

higher ranks. To easily distinguish annotated documents from the rest, all annotated documents are highlighted in yellow. The user can now select another document for viewing and annotation, repeating the process until a complete set of required instances related to the event topic is constructed.

Figure 6.5: Revised Document Ranking

Finally, INEXT can export a entity relation graph containing the annotated entity and relation instances. In the entity relation graph (see Figure 6.6), nodes represent entity instances and edges represent relation instances. We have adopted Graphviz [3] to draw the entity relation graph. Graphviz is an open source graph visualization software. It has several main graph layout programs, which take graph descriptions in a simple text language, and create diagrams in several useful formats such as images, SVG, etc.. A file containing the annotated entity and relation instances in the format required by Graphviz is generated and can be stored for future reference and updates. The file is named as "Graph.dot" and stored in the same directory as the document collection. Figure 6.7 shows the content of the graph file for drawing the entity relation graph in Figure 6.6. The entity instances are listed before relation instances in the file. Entity instances are connected using "->" symbols so as to represent relation instances and the labels indicate the corresponding relation classes.

6.3. ENTITY AND RELATION ANNOTATION MODULE

Figure 6.6: Exported entity relation graph of the event topic

```
digraph G {
    "KIM SUN";
    "BEHEADING OF KIM SUN";
    "ZARQAWI";
    "TAWHID WA AL-JIHAD";
    "KIM CHOON-HO";
    "KIM CHONG-KYU";
    "AL-QAEDA CELL";
    "ZARQAWI" -> "KIM SUN"[label="HARM T"];
    "ZARQAWI" -> "TAWHID WA AL-JIHAD"[label="LEADER"];
    "KIM SUN" -> "KIM CHOON-HO"[label="WORK"];
    "KIM SUN" -> "KIM CHONG-KYU"[label="RELATIVE"];
    "ZARQAWI" -> "BEHEADING OF KIM SUN"[label="RESPONSE T"];
    "TAWHID WA AL-JIHAD" -> "AL-QAEDA CELL"[label="CONNECT"];
    "AL-QAEDA CELL" -> "KIM SUN"[label="HARM O"];
    "AL-QAEDA CELL" -> "BEHEADING OF KIM SUN"[label="RESPONSE_O"];
}
```

Figure 6.7: Content of “Graph.dot”

Chapter 7

Conclusion and Future Work

7.1 Summary of Research

We have proposed a new domain specific event topic entity and relation extraction task and developed five pattern-based document ranking strategies in this thesis. Our objective is to select as few documents as possible to construct the entity and relation instances related to some event topics of interest to the user.

We have defined performance metrics to measure the performance of different document ranking strategies. We also constructed 3 datasets on 2 different recent terrorism events for experiment evaluations. The results conclude that PartialMatch I was likely to perform better than the other pattern-based document ranking strategies and was better than some existing content-based document ranking strategies on both KSB-100 and AEB-100 datasets. For a large scale dataset (AEB-1000), PartialMatch+ found all the required event topic related information by selecting only 5% of the documents.

In addition to the event topic extraction task, we have also proposed a personalized event topic entity relation extraction task. The objective of this task is to find the smallest subset of documents which contain all instances to the user only from the relevant classes. We have therefore designed a personalized partial match document ranking strategy incorporated with a weighting scheme for this task. Experimental results show that the personalized partial match strategy performed better than other pattern-based document ranking strategies on the personalized extraction task and the weighting scheme was able to give a fairly accurate weight estimation for each class.

7.2. FUTURE WORK

A working prototype system named INEXT was developed to support the proposed event topic entity and relation extraction task. We also implemented a named entity recognition module and a coreference resolution module which are presented in Appendix A. The evaluation and enhancement of this INEXT system will be part of our future work.

Although the proposed document ranking strategies have shown fairly good results in the experiments, there are still some limitations in them. Firstly, some domain knowledge is assumed to be given in order to carry out the extraction task, and this includes a set of domain specific classes, and a set of domain specific extraction patterns. Secondly, our work also assumes that an expert user will manually extract entity and relation instances from the selected documents.

7.2 Future Work

In the following, we will list a few important future works that should be investigated.

1. Enhancement on the proposed document ranking strategies

We have shown in Chapter 4 that our proposed strategies work well on the event topic entity and relation extraction task and maintain good performance on large document collection. Nevertheless, further investigations are still required to further improve their performance. Currently, PartialMatch I gave the best performance on selecting the optimal documents and finding instances from small document collection. PartialMatch+ delivered a better performance than the other strategies on selecting the relevant documents and also performed well on the large dataset. It will be therefore interesting to explore a combination of strategies to give a better performance in document ranking.

2. Enhancement of Extraction Techniques

We also plan to investigate the impact of IE system components on the extraction task performance. Now the performance of our proposed strategies are highly dependent on the performance of IE. Although we have adapted several entity extraction system components, such as BBN IdentiFinder, crystal single-slot extraction patterns, and applying known directory of Terrorists and Organizations,

7.2. FUTURE WORK

quite a few useful instances in the documents were still unable to be extracted. Therefore a better IE system component will be essential to contribute to more accurate event topic entity relation extraction.

3. Improvement on the Personalized Event Topic Extraction Task

As shown in Chapter 5, the proposed personalized partial match document ranking strategy performed better than the other strategies on the personalized event topic extraction task. The weighting scheme for entity and relation classes did help PPM by giving good estimations of class importance. However, when compared with the Ideal strategy, we feel that there are rooms for improvement.

4. Relaxation on domain knowledge and manual annotation assumptions

We plan to relax the assumption that a set of domain specific classes and a set of extraction patterns are given in order to carry out the extraction task in our future work. Only a basic set of classes and extraction patterns will be required in the future, novel and relevant classes and extraction patterns should be identified and included into the basic set during execution of the extraction task. Furthermore, we also plan to reduce the expert's effort from manually annotating the whole document for entity and relation instances in our future work.

Appendix A

Named Entity Recognition and Coreference Resolution

In this chapter, we first describe the Named Entity Recognition (NER) methods and tools used in our research. We present some limitations observed for most NER packages and propose a post-processing procedure to address them. Secondly, we explain in detail the Coreference Resolution (CR) module developed in our work and present some experimental results on our CR module.

A.1 Named Entity Recognition

Named Entity Recognition (NER) refers to identifying and extracting different types of named entities from a text document. NER is also important to our coreference resolution module as the latter uses the NER results to resolve coreferences.

A.1.1 Limitations of Named Entity Recognition

In NER [28, 46], different types of entities can be recognized, such as person names, organization names, locations and etc.. We have adopted BBN for extracting the entities of following types: person names and organization names [14]. The named entities extracted by BBN will be used by later coreference resolution module.

Although NER is a well studied research topic, there are some limitations in the existing NER packages (e.g., CATE and BBN). In general, we observe that there are three limitations. They are:

A.1. NAMED ENTITY RECOGNITION

1. Incomplete names

Incomplete name refers to a partial name extracted by some NER package while missing the full name. We consider those missing parts of a complete name as a missing name component. For example, “George Bush” may be recognized as “George”, leaving “Bush” as a missing name component. Incomplete names have been found in the results of existing NER packages such as GATE and BBN.

2. Incorrect name splitting

Incorrect name splitting refers to names splitted into multiple parts and each part is recognized as an individual named entity. For example, “George Washington” may be recognized as “George” and “Washington” as two different person names.

3. Salutations and designations

This refers to names recognized with some designations (e.g., “President Lee”, “Professor Zhang”), or salutations (e.g., Mr, Miss, Madam and etc). Some NER packages may extract names together with their salutations or designations. In our work, we focus on finding the actual name of each entity. Therefore, we would like to remove those salutations and designations from a named entity extracted.

A.1.2 Named Entity Recognition Post-processing

To address the above limitations, we introduce a post-processing procedure on the NER results by BBN which is used in our experiments and development of INEXT.

Algorithm 3 shows the detailed post-processing procedure. The first step in the procedure is to apply a part-of-speech (POS) tagger to tag all documents. The tagged document set is denoted by D_p . The tagger parses the documents into phrases and assigns them syntactic roles, e.g., noun phrase, verb phrase, etc.. A POS tagger called MXPOST [33] has been used in INEXT.

We have observed that missing name components are nouns or noun phrases, and missing name components should be next to the incomplete name extracted, either on the left or right of the latter. In Algorithm 3, the NER results by BBN is stored in the list L_i for each document d_i . A noun phrase p that is found right before and after a

A.1. NAMED ENTITY RECOGNITION

Algorithm 3 Named Entity Recognition Post-processing

```

1: Input: Named entity list  $L_i$  for each  $d_i \in D^p$ , indexed title list  $T$ 
2: apply POS on all documents and obtain  $D^p$ 
3: //For addressing limitations 1 and 2
4: for each  $d_i^p \in D^p$  do
5:    $old = null$ 
6:   while  $L_i \neq \phi$  do
7:      $ent = \text{dequeue}(L_i)$ 
8:     if  $ent$  is not found in  $old$  then
9:       for each noun phrase  $p$  that appears right before  $ent$  do
10:         $ent = p \oplus ent$ 
11:      end for
12:      for each noun phrase  $p$  that appears right after  $ent$  do
13:         $ent = ent \oplus p$ 
14:      end for
15:       $old = ent$ ; add  $ent$  to  $L_i^R$ 
16:    end if
17:  end while
18: end for
19: //For addressing limitation 3
20: for each named entity  $ent \in L_i^R$  do
21:   if  $ent$  is a person name then
22:     for each token  $t$  of  $ent$  from left to right do
23:       if  $t \in T$  then
24:         $ent = ent \ominus t$ 
25:       end if
26:     end for
27:     update  $ent$  in  $L_i^R$ 
28:   end if
29: end for
30: Output:  $L_i^R$ 

```

A.2. COREFERENCE RESOLUTION

named entity *ent* is concatenated with *ent* using the concatenation operation ($\oplus$). Once a named entity is post-processed, the recovered named entity is inserted into a resultant list L_i^R , which stores all post-processed named entities. Every time before conducting post-processing on a new named entity, a checking will be performed for the existence of the named entity in *old*, partially or completely. Therefore, if a full name expression has been recovered and stored in L_i^R , the subsequent parts of recovered names will not be processed.

To address the third limitation, we extracted 27,695 job titles from the U.S. Department of Labor website¹. In addition, we also added some titles in the terrorism domain, such as “Terrorist”, “Militant”, etc.. All the titles are stored in an indexed list T . As shown in Algorithm 3, for each named entity in L_i^R , we check whether it contains any title from T as a prefix. If so, we remove the prefix from the name using the delete operation $\ominus$. If multiple titles appear as prefixes in the name, they are iteratively removed.

A.2 Coreference Resolution

Firstly, our coreference resolution module has two aspects: pronoun and name coreference resolutions. Secondly, we describe the pronoun coreference resolution component deployed in our work. Lastly, we explain the proposed name coreference resolution component, by giving some detailed explanations on the proposed method.

A.2.1 Overview of Coreference Resolution

Coreference resolution refers to determining the identity relationships among objects within documents. In our work, the objects involved are named entities and pronouns. For example, in the following text²:

Sajida Al Rishawi, 35, and *her* husband were on a suicide mission at a hotel in **Jordan** where a wedding reception was being held. But *their* plan failed, *she* said, when only *her* husband’s bomb detonated. “*My* husband organised everything,” said **Sajida** in the Jordanian TV broadcast, translated and aired on **CNN** yesterday.

¹<http://www.bls.gov/ncs/ocs/ocsm/commmain.htm>

²Named entities are in boldface while pronouns are in italic.

A.2. COREFERENCE RESOLUTION

“Sajida Al Rishawi” and “Sajida” refer to the same person and resolving the identity between them is known as *name coreference resolution*. The “her”, “she” and “my” refer to “Sajida Al Rishawi” and resolving them is known as *pronoun coreference resolution*. Both the above two types of coreferences are most common and are addressed by our coreference resolution module.

With coreference resolution, equivalence classes of name entities and pronouns can be created. Each equivalence class consists of names and pronouns referring to the same real world person, organisation or object. This establishes the basis for assembling and integrating knowledge from different parts of a document, as well as from different documents. For example, in the above text, one can determine after coreference resolution that the entity, Sajida Al Rishawi, is involved in a mission entity, i.e., suicide bombing. The same person entity appeared in the media entity, CNN. In other words, this allows us to construct a network of inter-relationships among relevant entities similar to that shown in Figure 1.1.

In the following, we describe our approach to resolve both pronoun and name coreferences in the context of person and organisation names. These are currently the two types of entities handled by our coreference resolution module.

A.2.2 Pronoun Coreference Resolution

Pronoun coreference resolution is a well studied problem. Given a pronoun appearing in a sentence, Mitkov has developed some score function to measure the antecedent indicators of each candidate named entity (within two-sentence distance from the pronoun)[30]. Modern coreference resolution system, such as the multi-agent strategy system proposed by Zhou and Su [19], and the coreference resolution system by string matching proposed by Yang et al., solve both pronoun and name coreference resolution concurrently. In their methods, all possible anaphora (pronoun/name) and antecedents (names) are first identified by some pre-processing tools. Each pair of anaphora and antecedent is then determined by some constraints or classifiers for a successful reference matching.

In our work, we adopt the GATE’s pronoun coreference resolution implementation. GATE can determine for pronouns such as “it”, “its”, “he”, “she”, etc., the appropriate named entities found in the same document[13]. Bontcheva *et al.* presented some shallow

A.2. COREFERENCE RESOLUTION

methods for pronoun coreference resolution in GATE and claimed that it can achieve up to 79% precision and 78% recall on the ACE corpus using their methods [10]. We have applied this pronoun coreference component on two datasets we developed. A comparison using our system between Zhou’s coreference system was conducted and presented in later section.

A.2.3 Name Coreference Resolution

Name coreference resolution is conducted after the pronouns are resolved. A given person name or organisation name may appear in different forms and determining their equivalence is not always trivial.

In our work, we adopt a simple approach that resolves a name with a more complete name that appears before the former. This assumes that the names are processed according to order they appear. Algorithm 4 shows the detail of our proposed approach.

Algorithm 4 Name Coreference Resolution

```

1: input: the post-processed named entity Set  $L_i^R$  of document  $d_i$ 
2: for each name  $p_j$  in  $L_i^R$  do
3:   for each name  $p_k$  which is precedent of  $p_j$ , i.e.,  $k < j$  do
4:     if ( $\exists p_k$  s.t.  $p_j$  is a subset token strings of  $p_k$ ) and ( $p_k$  is the nearest name to  $p_j$ )
       then
5:        $p_j$  is coreferenced with  $p_k$ 
6:       Replace  $p_j$  with  $p_k$  in document  $d'_i$ 
7:     end if
8:   end for
9: end for
10: output: Name coreferenced document  $d_i^{new}$ 

```

Suppose we have a list of names to be resolved of each document. For each name to be resolved (e.g., p_j is “Al Rishawi”) also known as the target name, we first tokenize it into a string a tokens (e.g., {“Al”, “Rishawi”}). A search is then performed on the other names that appear before the target name. These names are tokenized and compared with the target name token string using string comparison. The nearest name with token string containing the target name token string (e.g., p_k is {“Sajida Al Rishawi”}) will be chosen to coreference with the target name.

A.2. COREFERENCE RESOLUTION

A.2.4 Experiments

In our experiments, we have evaluated our proposed coreference resolution module on KSB and AEB datasets. We also applied the coreference resolution system presented in [19] on the two datasets.

Evaluation Settings

We applied the standard scoring scheme for coreference task used in MUC6 [25]. This scoring scheme produces a recall and precision scores and also does not require explicit computation of the transitive closure of coreference.

Let S denote a set of equivalence classes in a document generated by an expert, i.e., $S = \{S_1, S_2, S_3, \dots\}$, each class S_i represents a set of entities which are equivalent and co-referred among themselves in the document. Each entity in a document refers to a pronoun or a noun phrase. Similarly, let H denote a set of equivalence classes in a document generated by our proposed coreference resolver system, i.e., $H = \{H_1, H_2, H_3, \dots\}$.

$p(S_i, H)$ is a partition of S_i relative to H . Each subset of S_i in the partition $p(S_i, H)$ is formed by intersecting S_i and each response set H_j in H that overlap with S_i . For each H_j in H , if there is an intersection between S_i and H_j (i.e., $I = S_i \cap H_j$ and $I \neq \phi$), I is a partition of S_i and $I \subseteq p(S_i, H)$. Every element in S_i which does not belong to H_j will be treated as a single element partition of S_i . In other words, all intersections between H_j and S_i together with all remaining elements in S_i but not in H form the partition set $p(S_i)$, i.e.,

$$p(S_i, H) = \bigcup_{j=1}^{|H|} \{S_i \cap H_j\} \cup \{\{s\} | s \in (S_i - \bigcup_{j=1}^{|H|} H_j)\} - \{\phi\}$$

Similarly, $p(H_i, S)$ is defined as a partition of H_i relative to S . Therefore, the recall and precision can be defined as follows:

1. Recall

$$R_{coref} = \frac{\sum_{i=1}^{|S|} (|S_i| - |p(S_i, H)|)}{\sum_{i=1}^{|S|} (|S_i| - 1)} \quad (\text{A.1})$$

2. Precision

$$P_{coref} = \frac{\sum_{j=1}^{|H|} (|H_j| - |p(H_j, S)|)}{\sum_{j=1}^{|H|} (|H_j| - 1)} \quad (\text{A.2})$$

A.2. COREFERENCE RESOLUTION

From P_{coref} and R_{coref} , we define F-measure F_{coref} , which combines both P_{coref} and R_{coref} as follows:

$$F_{coref} = \frac{2 \cdot P_{coref} \cdot R_{coref}}{P_{coref} + R_{coref}} \quad (\text{A.3})$$

Experimental Results

We applied our proposed coreference resolution module on the relevant documents in both KSB-100 and AEB-100 datasets. An expert annotated all necessary coreference linkages including both pronoun and name coreference linkages in all the documents. An evaluation was conducted on the overall performance of the results after pronoun and name coreference resolution steps. Furthermore, we applied the multi-agent coreference resolution approach proposed in [19] on the same document set. An overall performance evaluation was conducted on the results of the multi-agent approach.

Table A.1 shows the performance of our proposed coreference resolution module and the multi-agent approach. Our proposed coreference resolution module outperformed the multi-agent approach on the overall performance of coreference resolution on both KSB and AEB datasets.

	Our coreference resolution module	Multi-Agent Approach[19])
KSB	0.47	0.33
AEB	0.55	0.31

Table A.1: Coreference Resolution Results

Appendix B

Extraction Pattern Samples

We have shown the extraction patterns used for extracting instances of relation class “Org_Harm” in this appendix. There are six different extraction patterns manually created for this class. Each extraction pattern is derived from one of the linguistic structures shown in Section 4.2.1 by constraining one slot to some entity or action instances.

Figure B.1 shows the six different extraction patterns manually created for extracting instances of relation class “Org_Harm”. For example, consider the first pattern (ID=100) shown in Figure B.1. It is derived from the linguistic structure: ⟨Subject⟩ ⟨Verb⟩ ⟨Prepositional Phrase⟩. The ⟨Verb⟩ slot has been constrained by the action class “ws_HarmVerb” and the other two slots are variables. This multi-slot extraction pattern is able to extract relation instances of class “Org_Harm”, in which the subject extracted shows the victim’s name and the prepositional phrases indicates the terrorist organization’s name that harms the victim.

A list of verbs were gathered from WordNet and used as constraints for the “ws_HarmVerb” class. Those verbs are: “KILL”, “BEHEAD”, “ABDUCT”, “HURT”, “WOUND”, “INJURE”, “BRUISE”, “OFFEND”, “SPITE”, “DAMAGE”, “SHOOT”, “ELIMINATE”, “ANNIHILATE”, “EXTINGUISH”, “DECIMATE”, “DISPATCH”, “SACRIFICE”, “DECAPITATE”, “MURDER”, “SLAY”, “HIT”, “ASSASSINATE”, “EXECUTE”. In addition, a list of terrorist organization names was pre-collected from the Web (see Section 4.2.1). These names are used as constraints for the “ws_TerroristOrg” class in the patterns shown in Figure B.1.

APPENDIX B. EXTRACTION PATTERN SAMPLES

<p>CN-type Org_Harm ID: 100 Status: GENERALIZED</p> <p>Constraints:</p> <p>SUBJ:: ==> Victim classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>VERB:: ==> Verb mode: passive classes: ws_Root_Class mod class: ws_Root_Class head class: ws_HarmVerb</p> <p>PP:: ==> TerroristORG prep: BY classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p>	<p>CN-type Org_Harm ID: 103 Status: GENERALIZED</p> <p>Constraints:</p> <p>SUBJ:: ==> TerroristORG classes: ws_Root_Class mod class: ws_Root_Class head class: ws_TerroristOrg</p> <p>VERB:: ==> Verb mode: active classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>OBJ1:: ==> Victim classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p>
<p>CN-type Org_Harm ID: 101 Status: GENERALIZED</p> <p>Constraints:</p> <p>SUBJ:: ==> TerroristORG classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>VERB:: ==> Verb mode: active classes: ws_Root_Class mod class: ws_Root_Class head class: ws_HarmVerb</p> <p>OBJ1:: ==> Victim classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p>	<p>CN-type Org_Harm ID: 104 Status: GENERALIZED</p> <p>Constraints:</p> <p>SUBJ:: ==> Victim classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Victim</p> <p>VERB:: ==> Verb mode: passive classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>PP:: ==> TerroristORG prep: BY classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p>
<p>CN-type Org_Harm ID: 102 Status: GENERALIZED</p> <p>Constraints:</p> <p>SUBJ:: ==> Victim classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>VERB:: ==> Verb mode: passive classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>PP:: ==> TerroristORG prep: BY classes: ws_Root_Class mod class: ws_Root_Class head class: ws_TerroristOrg</p>	<p>CN-type Org_Harm ID: 105 Status: GENERALIZED</p> <p>Constraints:</p> <p>SUBJ:: ==> TerroristORG classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>VERB:: ==> Verb mode: active classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Root_Class</p> <p>OBJ1:: ==> Victim classes: ws_Root_Class mod class: ws_Root_Class head class: ws_Victim</p>

Figure B.1: Extraction patterns of relation class “Org_Harm”

Appendix C

List of Publications

During the two-year Master candidature, the author has contributed a total of 2 international publications. The list is given below.

- Zhen Sun, Ee-Peng Lim, Kuiyu Chang, Teng-Kwee Ong and Rohan Kumar Gunaratna. Event-Driven Document Selection for Terrorism Information Extraction. In *Proc. of 3rd IEEE International Conference on Intelligence and Security Informatics*, USA, May 2005 (Best Paper Award)
- Zhen Sun and Ee-Peng Lim. INEXT: An Investigative Search Tool for Knowledge Extraction. In *Proc. of Workshop on Intelligence and Security Informatics (WISI-2006)*, Singapore, April 2006

Bibliography

- [1] Badger information extraction (ie) software.
<http://www-nlp.cs.umass.edu/software/badger.html>.
- [2] Extremists online. <http://www.bombsecurity.com/extremists.html>.
- [3] Graphviz - graph visualization software. <http://www.graphviz.org/>.
- [4] Visual c#.net. <http://msdn.microsoft.com/vcsharp/>.
- [5] The sixth message understanding conference.
<http://www.cs.nyu.edu/cs/faculty/grishman/muc6.html>, 1995.
- [6] Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*, 2000, San Antonio, TX, USA.
- [7] Eugene Agichtein and Luis Gravano. Querying text database for efficient information extraction. In *Proceedings of the 19th IEEE International Conference on Data Engineering*, 2003, Bangalore, India.
- [8] James Allan, Ron Papka, and Victor Lavrenko. On-line new event detection and tracking. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 37-45. ACM Press, 1998, Melbourne, Australia.
- [9] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

BIBLIOGRAPHY

- [10] Kalina Bontcheva, Marin Dimitrov, Diana Maynard, Valentin Tablan, and Hamish Cunningham. Shallow methods for named entity coreference resolution. In *Proceedings of the Traitement Automatique des Langues Naturelles*, 2002, Nancy, France.
- [11] Thorsten Brants and Francine Chen. A system for new event detection. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pages 330–337. ACM Press, 2003, Toronto, Canada.
- [12] Francine Chen, Ayman Farahat, and Thorsten Brants. Story link detection and new event detection are asymmetric. In *Proceedings of Human Language Technology Conference*, 2003, Edmonton, Alberta.
- [13] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002, Philadelphia, PA, USA.
- [14] Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. An algorithm that learns whats in a name. *Machine Learning*, 34(1-3):211–231, 1999.
- [15] Christiane Fellbaum. Wordnet: An electronic lexical database. MIT Press, May 1998.
- [16] Aidan Finn and Nicholas Kushmerick. Active learning selection strategies for information extraction. In *Proceedings of Workshop on Adaptive Text Extraction and Mining*, 2003, Cavtat-Dubrovnik, Croatia.
- [17] Susan Gauch and John B. Smith. An expert system for automatic query reformation. *Journal of the American Society of Information Science*, 44(3):124–136, 1993.
- [18] Jade Goldstein, Vibhu O. Mittal, Jaime G. Carbonell, and James P. Callan. Creating and evaluating multi-document sentence extract summaries. In *Proceedings of the 2000 ACM International Conference on Information and Knowledge Management*, pages 165–172, 2000, McLean, VA.

BIBLIOGRAPHY

- [19] Zhou Guo and Su Jian. A high-performance coreference resolution system using a multi-agent strategy. In *Proceedings of 20th International Conference on Computational Linguistics*, 2004, Switzerland.
- [20] Scott Huffman. Learning information extraction patterns from examples. In *Proceedings of International Joint Conferences on Artificial Intelligence Workshop on New Approaches to Learning for Natural Language Processing*, 1995, Montreal, Quebec, Canada.
- [21] Jennifer Jie Xu, Hsinchun Chen, Yilu Zhou, and Jialun Qin. On the topology of the dark web of terrorist groups. In *Proceedings of IEEE International Conference on Intelligence and Security Informatics*, 2006, San Diego, California.
- [22] Jun-Tae Kim and Dan I. Moldovan. Acquisition of linguistic patterns for knowledge-based information extraction. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):713–724, 1995.
- [23] Krupka, George R. Description of the SRA system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference*, 1995, Maryland, Columbia.
- [24] Giridhar Kumaran and James Allan. Text classification and named entities for new event detection. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 297–304. ACM Press, 2004, Sheffield, UK.
- [25] Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *Proceedings of the Sixth Message Understanding Conference*, 1995, Maryland, Columbia.
- [26] Harsha V. Madhyastha, N. Balakrishnan, and K. R. Ramakrishnan. Event information extraction using link grammar. In *Proceedings of the 13th International WorkShop on Research Issues in Data Engineering: Multi-lingual Information Management*, 2003, Hyderabad, India

BIBLIOGRAPHY

- [27] David Masterson and Nicholas Kushmerick. Information extraction from multi-document threads. In *Proceedings of Workshop on Adaptive Text Extraction and Mining*, 2003, Cavtat-Dubrovnik, Croatia.
- [28] Diana Maynard, Valentin Tablan, Cristian Ursu, Hamish Cunningham, and Yorick Wilks. Named entity recognition from diverse text types. In *Proceedings of Recent Advances in Natural Language Processing*, 2001, Tzigov Chark, Bulgaria.
- [29] Ruslan Mitkov. Anaphora resolution: The state of the art. University of Wolverhampton, 1999.
- [30] Ruslan Mitkov. Robust pronoun resolution with limited knowledge. In *Proceedings of the 18th International Conference on Computational Linguistics*, 1998, Montreal, Quebec, Canada.
- [31] Ion Muslea. Extraction patterns for information extraction tasks: A survey. In *Proceedings of American Association for Artificial Intelligence Workshop on Machine Learning for Information Extraction*, 1999, Orlando, Florida.
- [32] David D. Palmer and David S. Day. A statistical profile of the named entity task. In *Proceedings of the Fifth Association for Computational Linguistics Conference for Applied Natural Language Processing*, 1997, California, USA.
- [33] Adwait Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference*, 1996, Philadelphia, PA.
- [34] Dennis Reidsma, Jan Kuper, Thierry Declerck, Horacio Saggion, and Hamish Cunningham. Cross document ontology based information extraction for multimedia retrieval. In *Supplementary proceedings of the 11th International Conference on Conceptual Structures*, 2003, Dresden, Germany.
- [35] Ellen Riloff. Automatically constructing a dictionary form information extraction tasks. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, 1993, Washington, D.C.

BIBLIOGRAPHY

- [36] Ellen Riloff. Automatically generating extraction patterns from untagged text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996, Oregon, Portland.
- [37] Ellen Riloff and Rosie Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 1999, Orlando, Florida.
- [38] Ellen Riloff and Jay Shoen. Automatically acquiring conceptual patterns without an annotated corpus. In *Proceedings of the Third Workshop on Very Large Corpora*, 1995, Massachusetts, USA.
- [39] Stephen Soderland. Learning to extract text-based information from the world wide web. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, 1997, Newport Beach, California.
- [40] Stephen Soderland, David Fisher, Jonathan Aseltine, and Wendy Lehnert. Crystal: Inducing a conceptual dictionary. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995, Montreal, Quebec, Canada
- [41] Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544, 2001.
- [42] Klaus Zechner. A literature survey on information extraction and text summarization. Term Paper, Carnegie Mellon University, 1997.
- [43] Michael Thelen and Ellen Riloff. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, 2002, Philadelphia, USA.
- [44] Yiming Yang, Jaime Carbonell, Ralf Brown, Tom Pierce, Brian T. Archibald, and Xin Liu. Learning approaches for detecting and tracking news events. *IEEE Intelligent Systems*, 14(4):32–43, 1999.

BIBLIOGRAPHY

- [45] Su Jian, Yang Xiao Feng, Zhou Guo Dong and Tan Chew Lim. Improving noun phrase coreference resolution by matching strings. In *Proceedings of 1st International Joint Conference on Natural Language Processing*, 2004, Geneva, Switzerland.
- [46] Li Zhang, Yue Pan, and Tong Zhang. Focused named entity recognition using machine learning. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 281–288, 2004, Sheffield, UK.
- [47] Yilu Zhou, Jialun Qin, Guanpi Lai, Edna Reid, and Hsinchun Chen. Building knowledge management system for researching terrorist groups on the web. In *Proceedings of the Americas Conference on Information Systems*, 2005, Omaha, Nebraska, USA.
- [48] Henry G. Goldberg and Ted E. Senator. Restructuring databases for knowledge discovery by consolidation and link formation. In *Proceedings of First International Conference on Knowledge Discovery and Data Mining*, pages 136–141, 1995, Montreal, Quebec, Canada.
- [49] Donald Brown, Jason Dalton, and Heidi Hoyle. Spatial forecast methods for terrorist events in urban environments. In *Proceedings of the Second Symposium on Intelligence and Security Informatics*, page 426–435, 2004, Tucson, Arizona.
- [50] Wai Yee Peter Wong and Dik Lun Lee. Implementations of partial document ranking using inverted Files. *Information Processing and Management*, 29(5):647–669, 1993.