NEAR-DUPLICATE NEWS DETECTION USING NAMED ENTITIES

A THESIS
SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Erkan Uyar
May, 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Prof. Dr. Fazlı Can (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Asst. Prof. Dr. Seyit Koçberber (Co-Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Asst. Prof. Dr. İlyas Çiçekli

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Ali Aydın Selçuk

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Dr. Kıvanç Dinçer

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet Baray
Director of the Institute

ABSTRACT

NEAR-DUPLICATE NEWS DETECTION USING NAMED ENTITIES

Erkan Uyar
M.S. in Computer Engineering
Supervisors
Prof. Dr. Fazlı Can
Asst. Prof. Dr. Seyit Koçberber
May, 2009

The number of web documents has been increasing in an exponential manner for more than a decade. In a similar way, partially or completely duplicate documents appear frequently on the Web. Advances in the Internet technologies have increased the number of news agencies. People tend to read news from news portals that aggregate documents from different sources. The existence of duplicate or near-duplicate news in these portals is a common problem. Duplicate documents create redundancy and only a few users may want to read news containing identical information. Duplicate documents decrease the efficiency and effectiveness of search engines. In this thesis, we propose and evaluate a new near-duplicate news detection algorithm: Tweezer. In this algorithm, named entities and the words that appear before and after them are used to create document signatures. Documents sharing the same signatures are considered as a near-duplicate. For named entity detection, we introduce a method called Turkish Named Entity Recognizer, TuNER. For the evaluation of Tweezer, a document collection is created using news articles obtained from Bilkent News Portal. In the experiments, Tweezer is compared with I-Match, which is a state-of-the-art near-duplicate detection algorithm that creates document signatures using Inverse Document Frequency, IDF, values of terms. It is experimentally shown that the effectiveness of Tweezer is statistically significantly better than that of I-Match by using a cost function that

combines false alarm and miss rate probabilities, and the F-measure that combines precision and recall. Furthermore, Tweezer is at least 7% faster than I-Match.

Keywords: Bilkent News Portal, I-Match, inverse document frequency (IDF), named entity recognition (NER), near-duplicate detection, t-test, Turkish Named Entity Recognizer (TuNER), Tweezer.

ÖZET

ADLANDIRILMIŞ NESNELER KULLANARAK YAKLAŞIK-AYNI HABERLERİ SAPTAMA

Erkan Uyar
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticileri
Prof. Dr. Fazlı Can
Y. Doç. Dr. Seyit Koçberber
Mayıs, 2009

Web dokümanlarının sayısı on yıldan fazla bir süredir katlanarak artmaktadır. Benzer şekilde, Web ortamında kısmen veya tamamen eşlenik dokümanlar sıklıkla görülmektedir. İnternet teknolojisindeki ilerlemeler beraberinde haber ajanslarının sayısını artırmıştır. İnsanlar haberleri farklı kaynaklardaki dokümanları bir araya toplayan haber portalları üzerinden okuma eğilimindedirler. Bu portallarda eşlenik veya yaklaşık aynı haberlerin bulunması yaygın bir problemdir. Eşlenik haberler fazlalık oluşturur ve çok az kullanıcı aynı bilgileri içeren haberleri okumak isteyebilir. Eşlenik dokümanlar arama motorlarının etkinliğini ve verimliliğini düşürmektedir. Bu tezde yeni bir yaklaşık aynı haberleri saptama algoritması olan Tweezer'ı önerip, değerlendirdik. Bu algoritmada adlandırılmış nesnelere karşılık gelen kelimeler ile bu kelimelerin öncesinde gelen ve onları izleyen kelimeler dokümanın imzasının oluşturulmasında kullanılmaktadır. Aynı imzayı paylaşan dokümanlar yaklaşık-aynı olarak kabul edilmektedir. Adlandırılmış nesnelerin saptanması için Türkçe Adlandırılmış Nesne Tanıyıcı, TuNER, yöntemi önerilmiştir. Tweezer'ın değerlendirmesi için Bilkent Haber Portalı'ndan sağlanan haberler kullanılarak hazırlanan doküman seti kullanılmıştır. Deneylerde Tweezer en gelişkin eşlenik saptama algoritmalarından birisi olan ve kelimelerin Ters Doküman Frekansı, IDF, değerlerini kullanarak doküman imzalarını çıkaran I-Match ile karşılaştırılmıştır. Yanlış ikaz ve kaçırma oranı olasılıklarını birleştiren bir maliyet fonksiyonu, ve anma ve duyarlılığı birleştiren F-ölçütü

kullanılarak Tweezer'ın I-Match'ten istatiksel olarak önemli ölçüde daha iyi olduğu deneysel şekilde gösterilmiştir. Bunun yanında Tweezer, I-Match'ten en az %7 daha hızlıdır.

Anahtar Sözcükler: adlandırılmış nesne tanıma, Bilkent Haber Portalı, eşlenik saptama, I-Match, ters doküman frekansı (IDF), t-test, Türkçe Adlandırılmış Nesne Tanıyıcı (TuNER), Tweezer.

Acknowledgements

I am deeply grateful to my supervisor Dr. Fazlı Can, who has helped me throughout my research, and encouraged me in my academic life. He always had time to discuss things and showed me the way when I felt lost in my research. It was a great opportunity to work with him. I am also grateful to my co-advisor Dr. Seyit Koçberber for his invaluable comments and contributions.

I would like to thank Dr. İlyas Çiçekli, who helped me a lot in my NLP project and gave me support whenever I needed. The ideas behind Named Entity Recognition part of this thesis emerged from that project.

I would like to thank Dr. Kıvanç Dinçer, my supervisor at TÜBİTAK-UEKAE/G222 Unit, who gave me opportunity to continue my academic career while still working and always supported me in this long path.

I would also like to thank Dr. A. Aydın Selçuk for his helpful comments.

Also, I am very glad that I have been a member of Bilkent Information Retrieval Group. I would like to thank my each friend, Özgür Bağlıoğlu, H. Çağdaş Öcalan and Süleyman Kardaş for their collaborations in our projects.

I am grateful to Bilkent University for providing me founding scholarship for my MS study. I would also like to address my thanks to The Scientific and Technological Research Council of Turkey (TÜBİTAK) for its scholarship during initial stages of my MS study.

Finally, I would like to thank my parents and brother for supporting my educational goals. Without their help and encouragement, this thesis has not been completed.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The digital information on the Internet and number of Internet users have been increasing in an exponential manner for more than a decade. According to [VAR2005] 90% of information currently produced is created in digital format and this trend will increase in the future. Many information technologies are emerged to make valuable information available to users. Information extraction, information retrieval, information filtering and document categorization are the examples of most common information technologies. The development in Internet technologies also carries some drawbacks along with its benefits. One of these is the existence of duplicate or near-duplicate documents on the Web. Retrieving documents from different sources on the Web generally results in duplication [CHO2002] and detection of such kinds of documents is studied under duplicate detection topics.

1.1 Motivations

Due to the rapid growth of electronic documents, redundant information increases on the Web. The replicated documents archived at different locations are one of the reasons of this problem. The number of news portals has increased in a parallel way with the increase of electronic information. In news portals, news articles coming from different sources are presented to users in a categorized manner. During this process the creation of partially or completely identical documents is inevitable, because news sites generally publish news coming from news agencies by either making small changes on the document or keeping it the same. In order to use the information available on the Web many technologies emerged, information retrieval systems is one of them. But the presence of duplicate documents decreases both effectiveness and efficiency of search engines [CHO2002]. Because duplicate results for user queries decrease the number of valid results of the query and this also decreases system effectiveness. Processing duplicate results is time-consuming and does not add any value to the information presented to the user. So, duplicate documents decrease the efficiency of a search engine.

Duplicate document detection has become a research field. Its purpose is to detect redundant documents to increase search effectiveness and storage efficiency of search engines. For example, Google does not show duplicate search results of a query. Google News again eliminates duplicate news at the first step. Detection of duplicate news documents in a fast way has great importance for users; because users do not want to wait in this process. They want to reach information as quickest as possible and if duplicate detection begins to slow down the access to the information, then they may choose to retrieve duplicate information. News portals offer elimination of duplicates fast by detecting duplicate news in indexing phase and performing duplicate removal in information retrieval process. Another option for accessing news documents is using news metasearch engines [LIU2007]. These search engines does not create document indexes as in the case of crawler-based search engines instead they uses several other search engines or databases of news sites. Since news documents are presented at the

time of user request, duplicate elimination should be done at this stage. Near-duplicate elimination also increases the diversity of search results by presenting only unique documents to the user.

1.2 Contributions

We developed a new duplicate document detection algorithm (Tweezer) using named entities. It uses signatures generated by using named entity centered word sequences for the comparison of documents. To the best of our knowledge, named entities have not been used in any of near-duplicate detection approaches so far. Tweezer is compared with I-Match. The I-Match algorithm uses IDF (Inverted Document Frequency) values of terms in order to select the terms to be used in document comparison. We prepared a test collection for duplicate document detection from the news documents obtained from Bilkent News Portal. According to experimental results Tweezer is statistically significantly more effective than I-Match and its duplicate processing time is at least 7% faster.

This research is a part of Bilkent Information Retrieval Group's studies and is used in the implementation of Bilkent News Portal that has the capabilities of new event detection and tracking, news categorization, information retrieval and information filtering [BAG2009, CAN2008a, CAN2009, KAR2009, OCA2009]. It provides support for automatic news text categorization using meta-data, multi-document summarization and near-duplicate news elimination. These are all innovative services for a news portal. We use Tweezer in this portal for detecting near-duplicate news documents after a query response and for selecting the news that will be displayed on the main page, see Figure 1.1.

Figure 1.1: Bilkent News Portal's main page.

1.3 Overview of the Thesis

For duplicate document detection first of all the features should be specified that will be used during comparison of two documents. In this thesis, we used named entities to determine our feature sets. In our approach, firstly named entities in the news stories are identified. After that named entity centered word sequences are generated and they are used as document descriptors. This process reduces the size of a document for comparison and subsequently the complexity of duplicate detection. Since news documents consist of an event and an event presents a story about the place, actor and

time of that event, we develop the Tweezer algorithm by using named entities. In our approach we do not just find the duplicate documents, but also identify clusters of them.

This thesis is organized as follows. In Chapter 2, we discuss existing approaches for named entity extraction and duplicate document detection. In Chapter 3, we introduce our named entity recognition approach, TuNER. Chapter 4 discusses the proposed approach for duplicate news detection, Tweezer. In Chapter 5 and 6, we respectively present the experimental environment and results. Finally, Chapter 7 concludes the thesis.

In this thesis the words news documents, news articles, news stories and documents are used interchangeably and also near-duplicate and duplicate are used interchangeably.

Chapter 2

Related Work

Duplicate document detection has become an important issue beginning with 1990s due to the growth of the Web. Several studies have been carried out in this area. The Web creates major plagiarism and copyright problems [HEI1996]. In this study our concern is the detection of near-duplicate news documents and we exploit the use of named entities in news articles. In this chapter, we give an overview of the studies related to named entity recognition and duplicate document detection.

2.1 Named Entity Recognition

The term "Named Entity" is used for the Sixth Message Understanding Conference (MUC-6) and it is extensively used in Natural Language Processing from that time [GRI1996]. Named Entity Recognition is developed as a subtask of Information Extraction, because people realized that information units like names including person,

location and organization names, and numeric expressions including time, date, money and percent expressions are the key points for information extraction.

Extraction of named entities from text is simple for humans. People firstly use orthographic rules in order to find named entities by looking at the first letter of a word. If it starts with a capital letter, then it is a candidate for a named entity. Up to this point this process is also simple for computer, but how it will identify a word as a named entity if it starts with a capital letter and in fact it is not a named entity. At this point, people use contextual clues to recognize named entities which they do not met before. For named entity recognition, there are two approaches from the point of view of computer: rule based approach and machine learning approach.

2.1.1 Rule-based Approaches

In rule-based approach, the entities are analyzed by experienced linguistics and hand-crafted rules are created. In order to extract entities mainly three phases are used: Linguistic Preprocessing, Named Entity Identification and Named Entity Classification [FAR2000].

Linguistic Preprocessing includes tokenizing, part of speech tagging, stemming and using the list of known names (database lookup). In order to identify named entities, boundaries of each named entity are detected. This includes the start and end structure of all the words that can be thought as named entity. In this phase possible named entities are generated by using punctuation marks or capitalization. Also, entities consisting of more than one word are identified at this stage. When possible named entities are identified, classification begins. Classification is performed in three stages: application of rules, database lookup classification and considering the matching of classified named entities with the unclassified ones. Rules are handcrafted and generated by experienced linguists. Rules are formed considering appositives or certain keywords that can precede or succeed a possible name. Classification starts by trying to match possible named entity with the generated rules. If there is no match with the rules, then database lookup is used. In these two stages, system’s aim is to define exact category of a named entity.

If classification cannot be performed in the previous two stages, then partial matching strategy is used as a final stage. This stage tries to identify truncated forms of names. For example, “Garanti Bank” is an organization name that is recognized in the task. Then, a truncated form of this phrase as “Garanti” can occur at the later part of the text. If this occurs, system tries to match this unclassified named entity with the classified one and finally determines its category.

One of the first researches in this area was performed by [RAU1991] and this study describes a system to extract and recognize company names by using heuristics and handcrafted rules. [WAN1992] developed a system to identify Chinese person names by using the concept of sublanguage. They designed a set of word formation rules in the light of most of the personal names appearing with a title or role noun. [WOL1995] introduces the knowledge representation structure based on conceptual graphs and represents the techniques to present known and unknown proper names. [FAR2000] study presents a NER system based on handcrafted lexical resources. Their proposed system was a part of Greek information extraction system and was tested on Greek corpus containing financial news.

2.1.2 Machine Learning-based Approaches

Machine learning approach is performed mainly in two stages: feature extraction and feature selection. In the feature extraction stage, previously generated training corpus is used. In this training corpus names and their categories are previously labeled. By using training corpus, features are extracted and classifier is trained with examples of sample names and their categories. After the classifier is trained by using training corpus, the system at this stage is tested by the real input. This time system tries to identify the category of unseen data. Machine learning approaches can be separated into three categories as supervised learning (SL), semi-supervised learning (SSL) and unsupervised learning (UL).

Supervised Learning

In SL the main purpose is to teach the system features of positive and negative examples on a large collection of annotated documents. SL is the most common approach used in NER for machine learning approach. For this purpose specific machine learning algorithms are used: Hidden Markov Models (HMM) [BIK1997], Maximum Entropy Models (ME) [BOR1998], Decision Trees [SEK1998], Support Vector Machines (SVM) [ASA2003] and Conditional Random Fields (CRF) [MCC2003]. HMM tries to predict hidden parameters from observable parameters. All these techniques are used in systems that read a large annotated training collection and create disambiguation rules. These rules are then applied to a different test collection to identify named entities.

Semi-supervised Learning

SL needs a large annotated corpus and it is not always possible to create such a corpus and preparing that kind of corpus is a very time consuming task. For this reason researchers prefer another option to perform named entity recognition work and this option is Semi-supervised Learning. Semi-supervised can also be called as weakly supervised and main technique for this approach is “bootstrapping”. In bootstrapping a small number of examples are given to the system and then system tries to find related sentences and contextual clues with the given examples. This process is iteratively applied in order to make the system find new clues with the help of newly discovered examples.

[BRI1998a] used seed examples and regular expressions to find author-title pairs on the Web. In his work he used examples like {Charles Dickens, Great Expectations} and his observation was that a site presents every author-title pair in the same format. If an example is found in a site then by applying the same rule with the found example several other author-title pairs can be found on that site.

[RIL1999] introduced mutual bootstrapping which includes growing set of entities and contexts in turn. But they reported low precision and recall rates in their

experiments. [CUC2001] and [PAS2006] are variants of mutual bootstrapping. [PAS2006] applied his technique on a very large collection containing 100 million web documents. He started with 10 example facts and succeeded to retrieve one million facts with a precision of 88%.

[HEN2006] showed how a NE classifier can be improved by using bootstrapping technique. He showed that using only very large corpus is not enough and he demonstrated that selecting documents in information retrieval like manner and using the documents that are rich in proper nouns brought better performance in the experiments.

Unsupervised Learning

Unsupervised learning is an alternative learning method as semi-supervised learning. In UL the most common technique is clustering, but there are also some techniques used depending on lexical resources or on statistics computed on large unannotated corpus. In this approach the main idea is to gather information related with named entities within the collection without having any clues from the outside.

[ALF2002] study the problem of assigning a named entity to an appropriate type. They used WordNet NE types in their work. When an unknown concept is found, first of all frequencies of words related with that concept is calculated for sample documents. Finally, the frequency of concept is compared with each topic signature in a top-down manner and concept is associated with the most similar topic during the comparison process.

[SHI2004] showed a way to detect named entities by using the distribution of words in news articles. Their observation is that named entities are likely to appear synchronously in news articles while common nouns are not. They detected rare named entities by only comparing a word's time series distributions in two news documents with an accuracy of 90%.

2.2 Duplicate Document Detection

Duplicate document detection became an interesting problem in late 1990s with the growth of Internet [SHI1998, BRO1997]. Most existing techniques for identifying duplicates or copies are divided into two categories, those of copy prevention and copy detection. Copy prevention techniques include physical isolation of the information and use of special hardware for authorization. Related work about copy prevention techniques will not be given because it is beyond of the scope of this thesis. Duplicate detection techniques try to identify duplicates. In this thesis techniques for detecting duplicate documents will be explained and a document will be considered as duplicate if it contains roughly the same semantic content whether or not it is a precise syntactic match [CHO2002].

Duplicate document detection can be achieved by calculating hash value for each document. Then each hash value will be compared with previously calculated hash values. In the case of hash equivalence, documents will be considered as duplicates. But this approach is very unsteady, because any change in the word order or existence of a typo will introduce different hashes and for this reason documents will not be considered as duplicates. This technique is suitable for detecting exactly the same documents. But we want to detect documents having slight changes in content or word order as duplicate and such documents are called near-duplicates.

Near-Duplicate detection techniques can be divided into three categories as similarity measures techniques, shingling techniques and fuzzy hashing techniques.

2.2.1 Techniques based on Similarity Measures

Techniques using similarity measures calculate a similarity value for each document pair and in order to understand a document is similar to another one its similarity value has to exceed some threshold value. In approaches using similarity measures the value associated with threshold is very important. Specifying a small value for the threshold will bring on false alarms in the case of duplicate detection and unrelated documents

will be identified as duplicates. On the contrary specifying a high value for the threshold will cause documents that are really duplicates to be missed. Several efforts have been made by researchers for determining the similarity of a document to another document. Well-known similarity measures can be divided into two categories: resemblance [BRO1997] and cosine similarity [SAL1975] measures.

In the resemblance approach the resemblance of two documents A and B is a number between 0 and 1 and two documents are considered as roughly the same when resemblance is close to 1. The notion of roughly the same is developed from the mathematical concept of resemblance. They defined the resemblance r of two documents A and B as

$$r(A, B) = \frac{|S(A) \cap S(B)|}{|S(A) \cup S(B)|}$$

- |A| denotes the size of set A.

Here the resemblance of two documents is the intersection of features over the union of features from two documents. They applied this approach to retrieve roughly the same documents which have the same content except for slight modifications. By this way they want to create a collection of documents in which closely related documents are gathered together in the same cluster. Resemblance approach is used by many researchers by specifying a threshold t to detect duplicate documents [BRI1995, SHI1995, SHI1996, SHI1998, FET2003].

The other most common similarity measure used in duplicate document detection is cosine [SAL1975]. Cosine similarity is the angle between two document vectors in n dimensional space. Given two document vectors d_i and d_j, the cosine similarity, θ, is represented using a dot product as

$$sim(d_i, d_j) = cosine(\theta) = \frac{d_i * d_j}{\|d_i\| \|d_j\|}$$

By using this similarity measure two documents' cosine angle is calculated and according to a given threshold value these two documents are defined as duplicates. It is important to specify a consistent value for the threshold; otherwise this will lead falsely identified duplicates. Many researchers made contributions to this approach in order to increase the effectiveness of similarity comparisons [SHI1995, HOA2003, BUC2000].

SCAM [SHI1995] stands for Stanford Copy Analysis Mechanism and it consist of a registration server composed of registered documents in order to be compared with new documents for checking overlap. Detection of copies is performed by comparing new documents on the basis of word frequencies with the registered ones. This system benefits from the chunking strategy. Chunking is the strategy of breaking up a document into more primitive units such as paragraphs, sentences or words. Chunking methodology that will be used during comparison is very important, because it may affect the search or/and storage cost. They used words as the unit of chunking in their research and used an inverted index structure for storing chunks. This strategy is an traditional IR approach and in this approach each entry of a chunk points to the set of documents in which that entry occurs. The set of documents pointed forms the posting list of entry and each item in this list has two attributes (docnum, frequency), where docnum is an unique identifier for registered document and frequency is the number of occurrences of chunk in that document.

In order to measure the overlap between a new document and a registered one, they proposed an updated version of cosine similarity which they called Relative Frequency Model (RFM). According to the experiments carried out with cosine similarity they saw that cosine measure is independent of the number of occurrences of a word in a document and they need a similarity measure in which the similarity decreases when the number of a word's occurrence increases. In order to incorporate this feature and detection of subset overlaps they first defined closeness set $c(d_1, d_2)$ that contains words

w_i in similar number of occurrences in two documents. A word w_i is inserted into the set $c(d_1, d_2)$ if it satisfies the following condition

$$\varepsilon - \left(\frac{F_i(d_1)}{F_i(d_2)} + \frac{F_i(d_2)}{F_i(d_1)} \right) > 0$$

- w_i denotes a chunk,
- d denotes a document,
- $F_i(d)$ is the number of occurrences of chunk w_i in d,
- ε = (2+, ∞) is a user tunable parameter.

By using closeness set they defined the subset measure of document d_1 to be a subset of document d_2 as

$$subset(d_1, d_2) = \frac{\sum_{w_i \in c(d_1, d_2)} \alpha_i^2 * F_i(d_i) * F_i(d_2)}{\sum_{i=1}^{N} \alpha_i^2 F_i^2(d_1)}$$

This expression is called asymmetric subset measure and it differs from the cosine similarity measure by normalizing the numerator of the expression with respect to the first document and only considering close words in the calculation of the numerator. With the help of asymmetric subset measure they defined the similarity of two documents d_1 and d_2 as follows

$$sim\ (d_1, d_2) = max\{subset(d_1, d_2), subset(d_2, d_1)\}$$

2.2.2 Shingling Techniques

Shingling is used for continuous subsequences of tokens in a document. The length of shingles used in the document is fixed and this type of shingling is used as w-shingling in the literature. A w-shingling is a set of unique shingles that can be used to predict similarity of two documents [BRO1997]. The idea of shingling first used in SIF

[MAN1994]. SIF is a tool for finding similar files in a large file system. In this system a document is seen as a set of all possible substrings of a certain length and if two documents have significant number of substrings in common, then they are considered as similar.

W-shingling resembles to N-grams. For example, the document “a cat is a cat is a cat” is tokenized as follows:

{a, cat, is, a, cat, is, a, cat}

This tokenized form can be interpreted as a set of continuous shingles in size four as

{{a, cat, is, a}, {cat, is, a, cat}, {is, a, cat, is}, {a, cat, is, a}, {cat, is, a, cat}}

When we remove the duplicates in this set we get the 4-shingling form of the document as

{{a, cat, is, a}, {cat, is, a, cat}, {is, a, cat, is}}

Well-known shingling techniques include COPS [BRI1995], KOALA [HEI1996] and DSC [BRO1997].

Since the number of digital documents is increasing in a fast way, in COPS researchers generate a system where original documents can be registered, and copies can be detected. This system will detect not just exact copies, but also documents that overlap in significant ways. They call this system as COpy Protection System (COPS).

The basic idea of COPS is as follows: There is a copy detection server. When an author creates a new work, he registers it at the server. As documents are registered, they are broken into small units. Each unit is hashed and a pointer to it is stored in a large hash table. When a document is to be checked, it is also broken into small units and each small unit is looked in hash table if it is seen before. If document that is compared shares

more than some threshold number of units, then a violation is flagged. Units can be paragraphs, sentences, words, or characters. They used sentences as units. Also they define chunks which are sequence of consecutive units in a document of a given unit type. There are four strategies considered in their approach (ABCDEF):

- One chunk equals one unit: (A, B, C, D, E, F).
- One chunk equals k non-overlapping units: (k=3, ABC, DEF).
- One chunk equals k units overlapping on k-1 units: (k=3, ABC, BCD, CDE, DEF).
- Use non-overlapping units: (AB, CDEF).

KOALA is an online system that is designed for textual matching and plagiarism detection. Their approach is based on the selection of subsequences of characters from the document and generating a fingerprint depending on a hash value for each subsequence. A similarity between two documents is calculated with the count of common subsequences. One of the alternatives in the generation of a fingerprint is to use every possible substrings of predefined length α. The size of this set is almost same with the size of the document. They called this type of fingerprinting as full fingerprinting. Using this technique is very expensive, because it needs more computation time and it consumes more storage space. For this reason they developed an alternative approach that removes frequently occurring subsequences from the fingerprint. They called this approach as selective fingerprinting. Detection of least frequently occurring substrings is again computationally expensive. In order to handle this problem they used only the first five letters of a substring. Their intuition behinds this was that the distribution of five letter sequences would give a useful approximation about the distribution of real substrings.

DSC (Digital Syntactic Clustering) is a mechanism to detect roughly the same documents on the web and in order to perform this approach it uses the resemblance similarity measure over the generated shingles on the document. Their algorithm is as follows:

1. Retrieve every document on the Web*.
2. Calculate the sketch for each document.
3. Compare the sketches for each pair of documents to see if they exceed a threshold for resemblance.
4. Combine the pairs of similar documents to make clusters of similar documents.

Figure 2.1: DSC Algorithm.

* Note that paper was published in 1995.

In order to retrieve documents located on the Web, they benefited from the AltaVista spider run. Their implementation of sketch is as follows:

1. Canonicalize documents by removing HTML formatting and converting all words to lowercase.
2. Generate shingles for every document by using shingle size *w* as 10 .
3. Use 40 bit fingerprint function based on Rabin fingerprints.
4. Use the "modulus" method for selecting shingles with mod 25.

Figure 2.2: Sketch implementation of DSC.

In shingling techniques if all generated shingles are used in the comparison of two documents, execution time of the algorithm is very long. In order to decrease comparison time, they do not use every shingle and use every 25^{th} shingle by using mod 25. But as they reported, this approach is also impractical, because DSC algorithm will require $O(10^{15})$ pairwise comparisons for 30 million documents.

In order to overcome the efficiency issues of DSC, they developed a different alternative called super shingles. Super shingles are calculated by sorting the shingles of documents and then shingling them again. If two documents share at least one super shingle, then they are considered as resembling to each other. They called this approach as DSC-SS (Digital Syntactic Clustering – Super Shingle) and algorithm is as follows:

1. Compute the list of super shingles for each document.
2. Expand the list of super shingles into a sorted list of <super shingle, ID> pairs .
3. Any documents that share a super shingle resemble each other are added into the cluster.

Figure 2.3: DSC-SS Algorithm.

Although this algorithm seems simple and more efficient method as compared to DSC, they report that it does not work well for short documents. Because short documents do not contain many shingles and expecting to generate a one common super shingle has a very low probability in short documents.

2.2.3 Fuzzy Hashing Techniques

Shingling and similarity approaches suffer from the efficiency issues. Fuzzy hashing is based on the whole document hashing and in this strategy main purpose is to produce a single document representation with characteristic features. I-Match [CHO2002] is the well known approach is this strategy.

I-Match filters documents based on collection statistics (Inverse Document Frequency - IDF). IDF is defined for each term as

$$tx = log\ (N / n)$$

- N is the number of documents in the collection,
- n is the number of documents containing the given term.

Their goal is to provide a duplicate detection algorithm that can scale to the size of the web and handle the short documents in the web. I-Match does not rely on strict parsing, but instead, uses collection statistics to identify which terms should be used as the basis for comparison. Their approach is removal of very infrequent terms or very common terms by this way resulting in a good document representation for identifying duplicate documents. Their algorithm is as follows:

1. Get document.
2. Parse document into a token stream, removing format tags.
3. Using term thresholds (IDF), retain only significant tokens.
4. Insert relevant tokens into Unicode ascending ordered tree of unique tokens.
5. Loop through token tree and add each unique token to the SHA1 diges. Upon completion of tree loop, a (doc_id, SHA1 Digest) tuple is defined.
6. The tuple (doc_id, SHA1 Digest) is inserted into the storage data structure based on SHA1 Digest key.
7. If there is a collision of digest values then the documents are similar.

Figure 2.4: I-Match Algorithm.

The runtime of I-Match is O(d log d) in the worst case when all documents are duplicates of each other and O(d) otherwise. According to test results they report that I-Match is five times faster than DSC-SS.

There are two options for the calculation of IDF values. First option is to use a generic collection and use IDF values from that collection in duplicate detection. The other option is to recalculate IDF values for each collection. Second option increases the actual runtime of algorithm.

Chapter 3

TuNER: Turkish Named Entity Recognizer

Due to rapid growth of electronic documents, many technologies emerged to make available the usage of information on the Internet by people. These technologies include automatic summarization, topic detection and tracking, and information retrieval. In these technologies core issue is to identify the main topics of a document. In such documents, topics are generally represented by words, sentences, concepts, and named entities [ZHA2004]. Named entities can be extracted with the help of named entity recognition techniques. “Named entity recognition is a subtask of information extraction that seeks to locate and classify atomic elements in text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc.” [WIK2009].

News are published in electronic domain and we may want to learn who has signed important contracts, information about terrorist events or companies may want to extract information about themselves and other companies from various newspapers. We should work on the retrieved documents by hand in order to extract this information or we can use named entity recognition techniques.

Extracted named entities are classified in three categories [POI2001]:

ENAMEX: Proper names that include names of persons, locations and organizations.

TIMEX: Temporal expressions such as dates and time.

NUMEX: Numerical expressions such as money amounts and percentages.

Our purpose is to detect near-duplicate news and news articles refer to events. An event can be described from the answers to the questions of who, where, when, why, what and how. Answer to the question of who gives us the persons that take a part in the event and where presents the event location. In near-duplicate document detection the key point is choosing the features that will be used instead of document itself and deciding how these features will be used in the comparison of two documents. So, we used our features in a way that they will represent the characteristics of documents. Named entities play an important role in the characterization of an event [KUM2004]. By extracting named entities in news, we can find the key items in that news, which specifies the characteristic of that document. In this study, we only deal with the extraction of "*ENAMEX*" types of named entities, because temporal expressions or numerical values may change in similar documents. There are various approaches in near-duplicate detection in the literature, but named entities have not been used in any of these approaches.

There are two common NER techniques: rule-based and machine learning approaches. We developed a rule-based NER system, called TuNER; because machine

learning approach needs detailed annotated named entity examples in order to train the NER system. It is not very easy to find prepared training examples and creating such an example list is costly. The structure of TuNER is explained in this chapter.

3.1 Named Entity Database Creation

For the implementation of TuNER, a list of person, location, and organization names are collected in separate tables for each category and a named entity database is constructed.

Person names table is generated by using the web site of "Türk Dil Kurumu" (TDK). TDK website provides a dictionary of person names. In addition to the TDK records, the personnel and student information database of Bilkent University are analyzed and name, surname, mother and father name fields of these records are extracted. Furthermore, Bilkent University sends documents to high school students for advertisement of the university each year and the names and surnames of these students are extracted and then inserted into the database.

In the case of location names, address records of personnel and student information databases are scanned, and city (şehir), county (ilçe) and district (semt) names are inserted into the database. Organization names table is created by using frequently used organization names as TRT, TÜBİTAK, MEB, etc. The number of named entities used in each category is given in Table 3.1.

Table 3.1: Named entity counts in the database

Named Entity Type	Count
Person	34,734
Location	19,504
Organization	46

3.2 Named Entity Grammar (Rule) Creation

Grammar used for TuNER is generated by handcrafted rules. Named entities can be identified by considering the words around them. Named Entity extraction rules are formed by taking into account the words that can precede or succeed a defined named entity.

Person names can be identified by a preceding title, such as

Sayın *Ali Öztürk, Belediye* ***Başkanı*** *Melih Gökçek, etc.*

or by a succeeding title, such as

Zeynep ***Hanım****, Mehmet* ***Efendi****, etc.*

Organization names can be identified generally by succeeding words, such as

Milli Eğitim ***Bakanlığı****, Bilkent* ***Üniversitesi****, Emniyet Genel* ***Müdürlüğü****, etc.*

Location names can be identified usually by succeeding words as in the organization names, such as

Atatürk ***Bulvarı****, Ali Sami Yen* ***Stadyumu****, Erciyes* ***Dağı****, etc.*

These rules are the parts of named entities most of the time. Some of them are used to detect named entities and some of them are perceived as a named entity when it is combined with the used rule. The number of generated rules in each category is given in Table 3.2 and complete list of the rules used in the study can be found in Table A.1, Table A.2, Table A.3 and Table A.4 in Appendix A.

Table 3.2: Rule counts used in TuNER

Named Entity Rule Type	Count
Person Prefix	16
Person Suffix	5
Location Suffix	66
Organization Suffix	39

3.3 TuNER

TuNER is a NER system, which tries to detect named entities located in a document. When a news document is given as an input to TuNER, first of all it is tokenized into small units as words starting with a capital letter. Consecutive uppercased words are evaluated together because it is possible that this word sequence denotes a named entity. After the document is tokenized, words containing apostrophes are treated different from others, because the possibility that a word containing an apostrophe to be a proper noun is high. After candidate named entities specified, it is time to determine which category they belong to. For this reason, extractors are developed for each category. TuNER tries to identify the category of a named entity by using the extractor methods of person names, location names, and organization names. In Turkish, some organization and location names may contain person names. For example:

Atatürk *Hastanesi,* ***Fatih Sultan Mehmet*** *Köprüsü.*

In order to prevent confusion in such a case, TuNER tries to extract location and organization names before person names. When each category is checked against candidate named entity sequence, a match is searched with the rule list of that named entity category. If a match is found with the rule list than that named entity is associated with the category of rule list, otherwise candidate named entity is compared with the prepared sample named entity tables in database. If a match is not found in the database, then partial matching technique is applied to the candidate sequence. In partial matching, candidate named entity is compared with the named entities detected earlier in the document. If a match is not found by using partial matching technique, then this

candidate named entity sequence is considered unclassified and it is not used in duplicate news detection.

Output of TuNER for a given sample input document is given in Figure 3.1. Partial matching may give successful results in some situations. For example, an organization name “Garanti Bankası” may exist at the beginning of a document. But this organization name may be used as only “Garanti” in the following sections. Named entities can be extracted by using partial matching techniques in such cases. Operation of TuNER is depicted in Figure 3.2.

INPUT:
Türkiye*'de en yüksek maaşı alan* ***CEO****'lar arasında* ***Shell*** *Genel Müdürü* ***Canan Ediboğlu****,* ***Microsoft Türkiye*** *Genel Müdürü* ***Çağlayan Arkın*** *ve* ***Unilever Türkiye Yönetim Kurulu*** *Başkanı* ***İzzet Karaca****'nın isimleri geçiyor.*

Mersin Üniversitesi*'nde karşıt görüşlü öğrenciler arasında dün başlayan gerginlik sürüyor.*

Santrali işleten şirkete bu yıl ***Muğla Çevre İl Müdürlüğü*** *tarafından 7 defa para cezası uygulandı.*

TuNER OUTPUT:

Person Names	Location Names	Organization Names	Unclassified
Canan Ediboğlu	Türkiye	Unilever Türkiye Yönetim Kurulu	CEO
Çağlayan Arkın	Mersin	Mersin Üniversitesi	Santrali
İzzet Karaca	Muğla	Muğla Çevre İl Müdürlüğü	
		Microsoft Türkiye	

Figure 3.1: TuNER output according to a sample input.

There are some problems regarding to the uppercase letters. The uppercase letters are the starting point for the detection of named entities, but every word starting with a capital letter may not be a proper noun. For example,

Deniz bugün çok soğuk, değil mi?

In this sentence “Deniz” is not a person name. Its first letter is capital letter, because it is at the beginning of a sentence, but system cannot understand this case and identifies “Deniz” as a person name, although it is not.

Also, missing punctuation marks reveals ambiguities. For example,

Atatürk Türk milletinin zeki olduğunu vurgulamıştır.

In this sentence, there should be a comma between “Atatürk” and “Türk”, but it is not included. So, our system identifies “Atatürk Türk” as person name and surname, although “Türk” is a name of a nation.

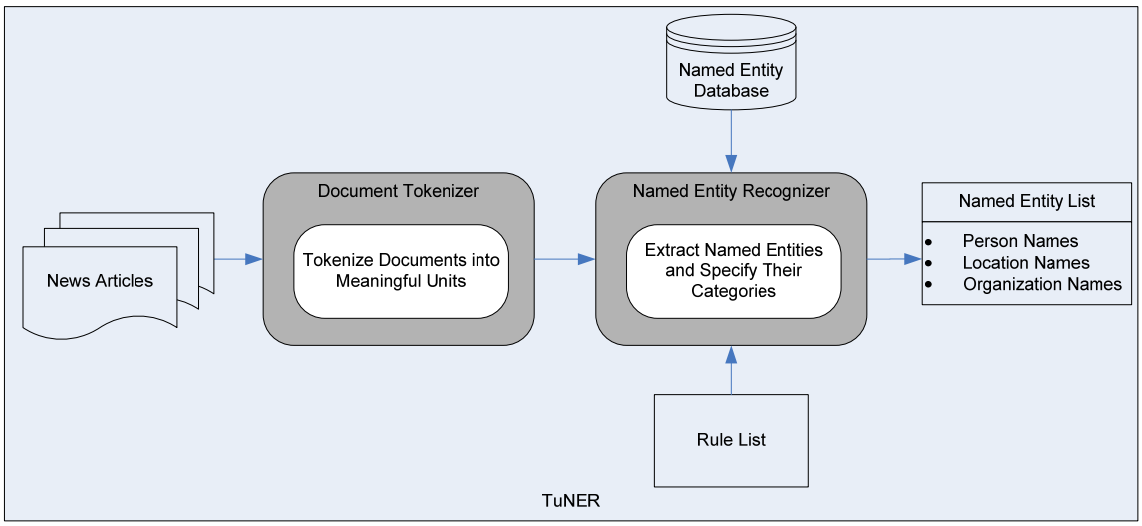

Figure 3.2: Operation of TuNER.

Chapter 4

Tweezer: Near-Duplicate News Detection Using Named Entities

In this study we developed a new near-duplicate detection algorithm, called Tweezer, by combining the characteristics of the shingling and fuzzy hashing techniques. Tweezer is based on the common use of named entities in news articles. This chapter presents the Tweezer algorithm.

4.1 Motivation for Using Named Entities in Near-Duplicate Detection

Duplicate document detection can be made by simply comparing the fingerprints of two documents, but this option is suitable for only detecting exact duplicates. Because any change of word order or the existence of a typo in one of the documents will change the

fingerprint of that document. So, two documents will differ from each other although they are not. In order to eliminate such problems, several techniques are developed and one of them is using similarity measures. By using similarity measure techniques two documents are compared with each other on the resemblance of features and if the resemblance value calculated according to the chosen similarity measure exceeds the specified threshold then these two documents are considered as duplicates. Features are selected to be words, sentences or paragraphs. The value assigned to threshold is a very important point in these techniques, because according to this threshold two documents are considered as duplicate or not. In document-to-document similarity each document is compared to every other document and thus theoretical runtime of these algorithms is O (d^2), where d is the number of documents.

The other approach for duplicate detection is using shingling techniques. A shingle is a set of w contiguous terms and shingling is the process of generating shingles for a document. The number of terms in each shingle is previously specified. In this approach, a document is represented as the collection of shingles and two documents are compared with each other according to the number of common shingles. The comparison is performed by using the similarity measures. In shingling approach rather than comparing two documents, generated subdocuments are compared. The number of shingles generated is approximately equal to the number of words in the document. It can be defined as

$$Shingle\ Count = n - w + 1$$

- n: Word count in the document.
- w: Shingle size in terms of number of words.

According to this definition, comparison of shingles of two documents is similar to the document-to-document similarity approach and for this reason their theoretical runtime complexity is O (d^2) as in the case of similarity measures, where d is the number of documents.

As we stated before named entities in news present the key points in events and they can be used to describe news for duplicate detection. The important point in named entity usage for duplicate detection is how they are employed to differentiate two documents. Named entities may not be used solely, because two different documents may contain the same named entities. In order to overcome this problem, we generated a new approach that contains named entities together with the words surrounding them. For this purpose, we create named entity centered word sequences and call it *pNEs* where

- *p*: Prefix Count. The number of words that are used before named entity.
- *NE*: Named Entity.
- *s*: Suffix Count. The number of words that are used after named entity.

The *pNEs* structure is a specialized form of a shingle. It can be seen as a named entity-based shingle structure. In this structure, the most important part is to specify the values for *p* and *s*. In order to choose values for *p* and *s*, an experiment is conducted for different *p* and *s* values over a set of 10,000 news documents. In the experiment, we ran I-Match algorithm and each *pNEs*-based duplicate detection approaches. In all of the runs, the number of generated duplicate clusters and duplicate news articles are recorded. The result of the experiments is given in Table 4.1.

Table 4.1: Experimental results for detection of *p* and *s* values

Method	Duplicate Cluster Count	Duplicate News Count
I-Match	527	1,075
1NE1	595	1,234
2NE2	577	1,191
3NE3	556	1,141
4NE4	542	1,112
5NE5	528	1,083
6NE6	528	1,082
7NE7	528	1,082

In Tweezer, we used five for *p* and *s* values. Because when we look at the results of the experiments, we see that the number of duplicate news detected tends to stabilize

after five is used for *p* and *s*. We see that using five is enough and when we increase *p* and *s* value, the size of word sequences in the document also increases. This may decrease the execution time efficiency of the system. On the other hand using smaller values for *p* and *s* may increase the number of false duplicates, because the number of duplicates detected decreases significantly up to the value of five and after that it remains nearly the same. The generated *5NE5* structured shingles for a given sample text is shown in Figure 4.1.

Bursa'nın Orhangazi İlçesi'nde apandisit ameliyatı sırasında doktor hatasına bağlı olarak kalın bağırsağının yırtıldığı ve bunun sonucunda vücuduna enfeksiyon yayıldığı için öldüğü ileri sürülen 13 yaşındaki Sevecan Ercan'ın dün toprağa verilen cesedi, bugün otopsi yapılmak üzere mezardan çıkartıldı.
• ***Bursa Orhangazi İlçesi*** *apandisit ameliyatı sırasında doktor hatasına* • *öldüğü ileri sürülen 13 yaşındaki* ***Sevecan Ercan*** *dün toprağa verilen cesedi bugün otopsi yapılmak üzere*

Figure 4.1: List of *5NE5* structured shingles for a sample text
(named entities in the lower box are shown in boldface).

We extend TuNER to detect *pNEs* structured shingles rather than detecting named entities only. Extended TuNER detects named entities in *5NE5* structured word sequences. Finally, the algorithm returns the list of all *5NE5* structured word sequences.

4.2 The Tweezer Algorithm

Our motivation is to detect near-duplicate news documents with the help of named entities in an efficient and effective way. Tweezer works in coordination with TuNER. The input document is processed by obtaining named entity-based shingles. The set of these named entity-based shingles is used instead of document itself. Our observation is that two documents containing the same named entity-based shingles are considered as near-duplicates. The idea behind this approach is that named entities are the lead actors in news articles. These named entities and the word sequences around them should resemble each other in two documents in order to be considered as near-duplicate.

After named entity-based shingles are generated all shingles are combined and a hash value is calculated for that shingle sequence by using SHA1 [NIS1995] hash algorithm. Then a <docId, hashValue> pair is inserted into a hash table. When a new document comes to the system as input, the same procedure is applied to that document; if a match occurs for <hashValue> terms then two documents are considered as near-duplicates.

The Tweezer algorithm is the combination of the shingling and I-Match approaches. The shingling side of our algorithm is stated in the previous section (the pNEs structure). I-Match uses IDF (Inverse Document Frequency) values of terms to identify which terms should be used as the basis for comparison. However, in our case, we use named entities as the basis and extend our starting point with words surrounding named entities. The pseudocode of Tweezer is given in Figure 4.2.

1. Parse document using TuNER and generate *pNEs* structured shingles.
2. Concatenate *pNEs* structured shingles in ascending order.
3. Retain only one of the *pNEs* structured shingles which are replication of each other.
4. Retrieve the hash of concatenated *pNEs* structured shingles by using the SHA1 hash function.
5. Insert <docId, hashValue> pairs into database.
6. Conclude that two documents are near-duplicates if a match occurs for "hashValue" in hash table.

Figure 4.2: Tweezer algorithm.

The complexity of Tweezer is O(d), where d is the number of documents, as in the case of I-Match, since identification of duplicates is performed during the insertion into the database. In this approach, all documents are visited only once in order to create the hash value and a check whether the same signature exists in the hash table is on the order of O(log d). The general working principals of Tweezer is depicted in Figure 4.3.

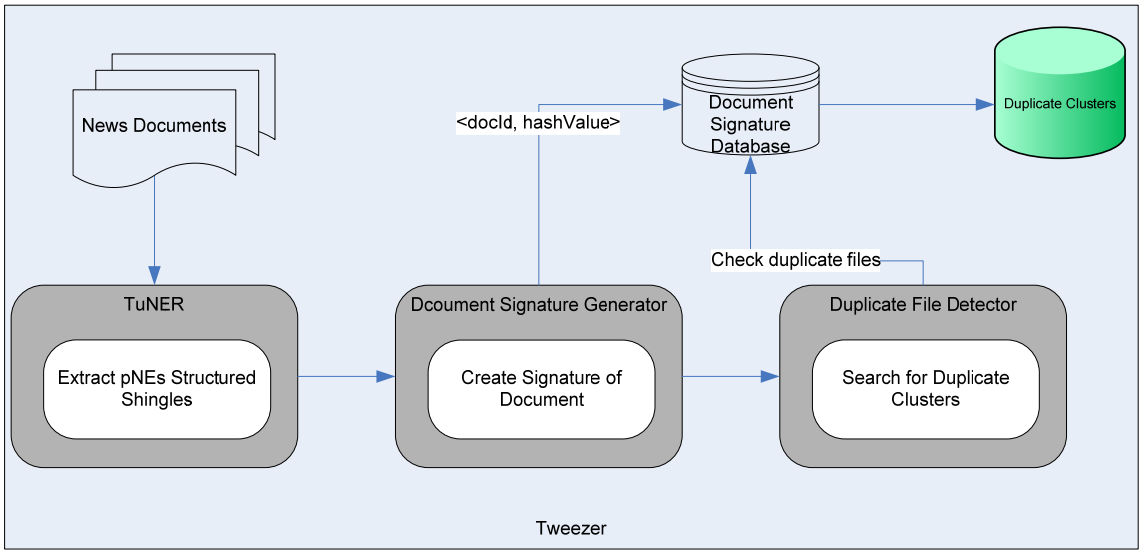

Figure 4.3: General working principals of Tweezer.

The Tweezer algorithm is currently used in Bilkent News Portal. In this portal each time a news story is added to the system, Twezeer algorithm generates a signature of that document. Every document signature is entered to the database. Near-duplicate news detection is performed in coordination with information retrieval (IR) system in Bilkent News Portal. When user enters the query details for his search, system prepares the documents related with his search. This operation is performed under the IR system. These query results are filtered by checking <docId, hashValue> pairs with the document signature table in the database and results without duplicates are returned to the user. The usage of Tweezer in Bilkent News Portal is depicted in Figure 4.4.

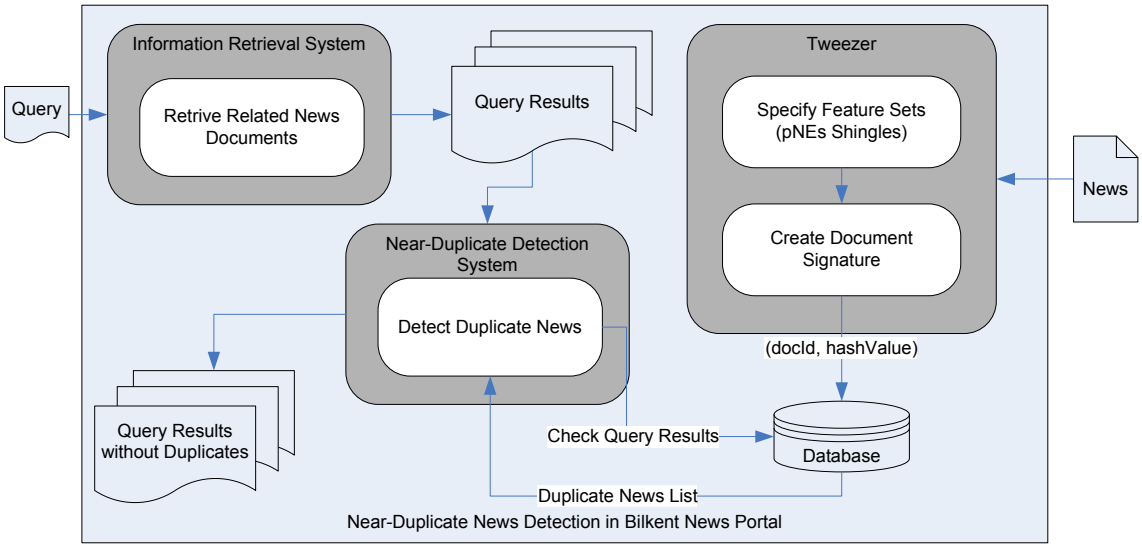

Figure 4.4: Near-duplicate news detection in Bilkent News Portal.

Non-existence of named entities in a document is a drawback for Tweezer, because it uses named entities in the creation of document signatures. In order to resolve this problem, Tweezer uses first and last twenty words of a document and combines these words together to create the document signature. If the document size is smaller than forty words, whole of the document is used in the creation of signature. The reason behind this approach is that important issues are given generally at the beginning and end of documents. The beginning section of a document gives some introductory information about the story in it and closing sections generally reach some conclusions in the documents. Therefore, beginning and closing sections can be used as a small summary of the document.

Chapter 5

Experimental Environment

In this chapter we define the architecture of our experiments. Most of the experiments of duplicate document detection approaches are performed on TREC data or ad hoc corpora constructed from collections of web pages. We used news documents of Bilkent News Portal that are coming from eight different sources in the experiments.

Bilkent News Portal crawls and indexes approximately 1,500 documents in a day and on each day we observe several near-duplicate documents. The distribution of news according to sources is given in Figure 5.1.

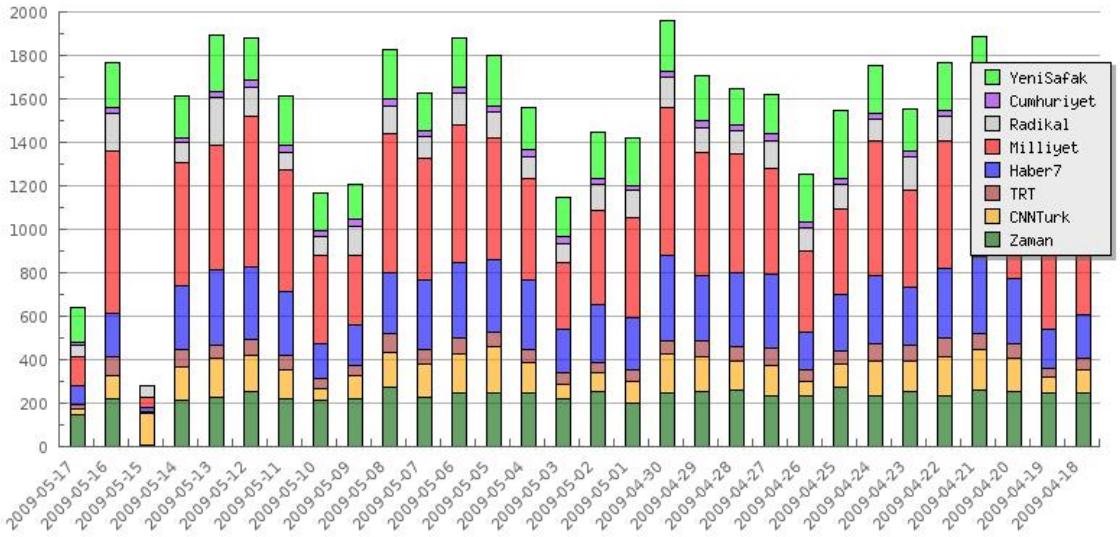

Figure 5.1: Distribution of news according to sources.

Our goal in this study is to create an effective and efficient system that can detect near-duplicate news documents. In the following sections we analyze these aspects of Tweezer.

5.1 Experimental Environment for Effectiveness Tests

Effectiveness relates to how well a proposed system works in practice. In order to perform effectiveness experiments we created test sets containing the news documents that are collected for Bilkent News Portal. News stories are crawled from eight different sources under twelve categories every day. We prepared two types of test sets. The first type of test sets (Test Collection A) contains thirty different sets and each one consists of 2,250 news stories. The number of documents in each test set news category is proportional to the total number of documents in that news category in the current news portal database. The number of documents in each category in database at the creation of test sets and their corresponding values in test sets are given in Table 5.1 (for text size of the test sets refer to Table 5.2).

Table 5.1: Number of documents in each database category and number of documents in each test set for Test Collection A

News Category	No. of Docs in Database	No. of Docs in Test Set
Ana Sayfa	43,659	300
Dış Haberler	3,712	100
Dünya	11,369	200
Ekonomi	22,219	200
Gündem	11,019	200
Kültür Sanat	4,098	100
Politika	3,457	100
Sağlık	2,013	50
Siyaset	4,974	100
Son Dakika	99,979	500
Spor	19,356	200
Türkiye	18,346	200

Table 5.2: Text size of documents in each test set

Test Set Name	Size of Test Set (MByte)
TestSet1	10.00
TestSet2	9.63
TestSet3	9.52
TestSet4	9.93
TestSet5	9.94
TestSet6	10.10
TestSet7	9.92
TestSet8	10.00
TestSet9	9.96
TestSet10	9.83
TestSet11	10.00
TestSet12	10.30
TestSet13	9.87

Test Set Name	Size of Test Set (MByte)
TestSet14	10.10
TestSet15	10.00
TestSet16	10.20
TestSet17	10.30
TestSet18	10.00
TestSet19	9.83
TestSet20	9.92
TestSet21	10.10
TestSet22	10.10
TestSet23	10.50
TestSet24	9.86
TestSet25	10.20
TestSet26	10.70
TestSet27	10.40
TestSet28	10.40
TestSet29	10.20
TestSet30	10.20

The second type of test sets (Test Collection B) again consist of thirty test sets, but this time each test set contains documents of the same category. Each test set contains 2,500 documents. The categories associated with each test set and the text sizes of documents in that test set are given in Table 5.3.

Table 5.3: Category and size of documents in each test set for Test Collection B

Test Set Name	News Category	Size of Test Set (Mbyte)
TestSet31	Ana Sayfa	11.00
TestSet32	Ana Sayfa	11.00
TestSet33	Ana Sayfa	11.30
TestSet34	Ana Sayfa	11.30
TestSet35	Ana Sayfa	11.00

Test Set Name	News Category	Size of Test Set (Mbyte)
TestSet36	Dış Haberler	10.80
TestSet37	Dünya	10.70
TestSet38	Ekonomi	13.30
TestSet39	Ekonomi	12.80
TestSet40	Gündem	10.40
TestSet41	Kültür Sanat	11.10
TestSet42	Politika	13.10
TestSet43	Sağlık	9.46
TestSet44	Siyaset	13.40
TestSet45	Son Dakika	11.30
TestSet46	Son Dakika	11.40
TestSet47	Son Dakika	11.10
TestSet48	Son Dakika	11.30
TestSet49	Son Dakika	11.20
TestSet50	Son Dakika	11.40
TestSet51	Son Dakika	11.30
TestSet52	Son Dakika	11.50
TestSet53	Son Dakika	11.20
TestSet54	Son Dakika	11.10
TestSet55	Spor	11.70
TestSet56	Türkiye	11.50
TestSet57	Türkiye	11.20
TestSet58	Türkiye	11.10
TestSet59	Türkiye	12.40
TestSet60	Türkiye	11.70

5.2 Experimental Environment for Efficiency Tests

Efficiency is related to implementing the work in most cost-effective way. For efficiency tests we created seven test sets (Test Collection C) in order to measure the runtime performance of Tweezer with the baseline approach, I-Match. We created test sets from

the news documents collected for large scale Turkish information retrieval experiments, since it provided us larger set of documents at the time of experimental setup [CAN2008b]. Number of documents in the test sets and corresponding text size are given in Table 5.4.

Table 5.4: Number and size of documents in test sets used in efficiency experiments

Test Set Name	No. of Docs	Size of Test Set (MB)
TestSet61	6,250	26.2
TestSet62	12,500	52.5
TestSet63	25,000	104.0
TestSet64	50,000	209.0
TestSet65	100,000	419.0
TestSet66	200,000	846.0
TestSet67	400,000	1,700.0

Chapter 6

Experimental Evaluation Measures and Results

In this chapter, we present the experimental results. Firstly we will define our evaluation measures and then continue with the results of effectiveness and efficiency experiments.

6.1 Evaluation Measures

Evaluation measures used in the effectiveness experiments are false alarm probability (rate) - miss probability (rate) and precision - recall. These measures are defined as follows (for definitions a, b, c, and d please refer to Table 6.1).

$$Miss\ Rate = M = \frac{c}{a+c}$$

$$False\ Alarm\ Rate = F = \frac{b}{b+d}$$

$$Precision = P = \frac{a}{a+b}$$

$$Recall = R = \frac{a}{a+c}$$

Table 6.1: False Alarm – Miss Rate structure

	Duplicate	**Not Duplicate**
Retrieved	a	b
Not Retrieved	c	d

In Table 6.1, the "Retrieved" documents are those that have been detected as duplicate by a duplicate detection algorithm, and the "Duplicate" documents are really duplicates manually labeled by annotators. In TDT2 [TDT2009], in order to analyze detection effectiveness a cost function was used. We modified the TDT cost function for duplicate document detection, accordingly duplicate cost function is defined as follows.

$$Cdup = costfa * P(fa) * (1 - P(duplicate)) + costm * P(m) * P(duplicate)$$

- *P(fa)* is the probability that a system produces false alarm,
- *P(m)* is the probability that a system produces miss,
- *P(duplicate)* is the ratio of duplicate news documents in Test Collection A and Test Collection B, that are found by both I-Match and Tweezer. *P(duplicate)* = 0.07 (7%) is used in the study. (In TDT2, for this case *P(event)* is used and it is the prior probability of a document to be related to an event.),
- $cost_{fa}$ and $cost_m$ are constants, and $cost_{fa} = cost_m = 1.0$ in cost function.

In order to make it easy to interpret cost values, normalize version of this cost function may be used. Normalize version of cost function, as it is done in the TDT studies, is defined as follows.

$$C_{dup-norm} = \frac{C_{dup}}{Minimum\{\cos t_m * P(duplicate), \cos t_{fa} * (1 - P(duplicate))\}}$$

By following the reasoning given in the above discussion, the cost function is defined as follows.

$$C_{dup-norm} = \frac{0.93 * F + 0.07 * M}{Minimum\{1 * 0.07, 1 * 0.93\}} = 13.29 * F + M$$

In the rest of the thesis C_{dup} is used instead of $C_{dup\text{-}norm}$.

In our Bilkent News Portal documents that are falsely identified as duplicates are more critical than missed duplicates and this is generally true for news portals. In the news portal, users retrieve documents after an information retrieval process and at this step documents identified as duplicates are not shown to them (since news consumers may want see duplicates for various reasons, they are kept in the collection). So, user will not be aware of the existence of a falsely identified document. Because of this reason higher factor is associated to false alarm than miss rate in the cost formula. The cost formula is adapted from Papka's approach used in new event detection and tracking [PAP1999].

We combined precision and recall values with F-measure [RIJ1979]. This measure is known as the F_1 measure in which recall and precision are evenly weighted. The F-measure used in the study is given as follows:

$$F = \frac{2 * P * R}{P + R}$$

We can clarify our cost measure with an example. Assume that we have documents d_1, d_2, ... d_{20} in our test set and each algorithm (I-Match and Tweezer) is run with this test set. Possible results of these algorithms are shown with the help of a Venn diagram in Figure 6.1 [VEN1880].

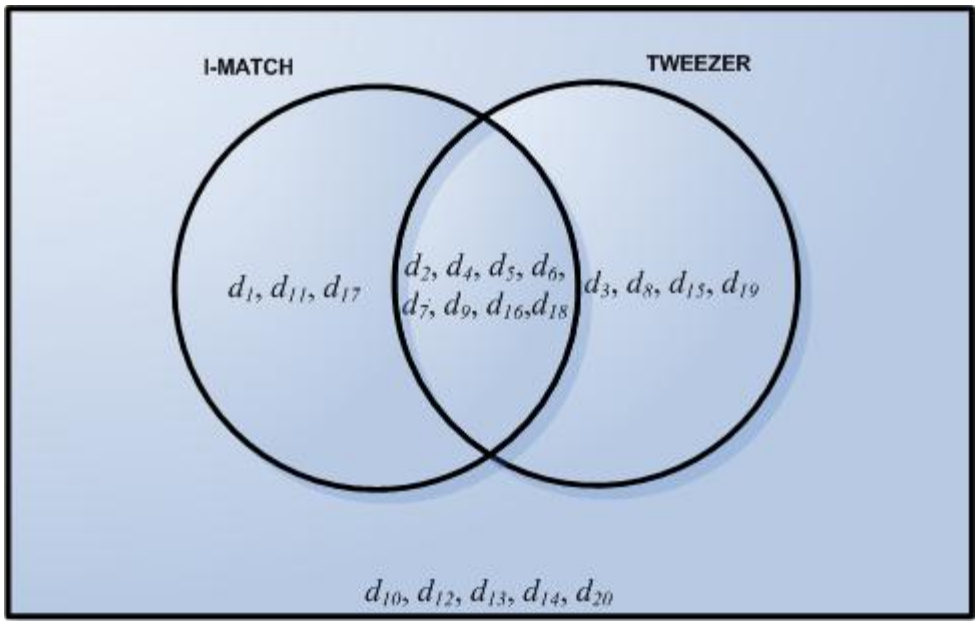

Figure 6.1: Possible results of I-Match and Tweezer algorithms for a sample test set.

According to this figure, documents in the intersection set (d_2, d_4, d_5, d_6, d_7, d_9, d_{16}, d_{18}) are detected as duplicates by both algorithms, but difference sets represent documents (d_1, d_{11}, d_{17} and d_3, d_8, d_{15}, d_{19}) are identified as duplicates by only one algorithm. (Example news detected as duplicates by either I-Match or Tweezer are given in Figure A.9 and Figure A.10 in Appendix D.) There are also documents out of the sets and these documents (d_{10}, d_{12}, d_{13}, d_{14}, d_{20}) are not considered as duplicates by any of the algorithms.

In order to calculate false alarm and miss rate, we must be sure about which documents are really duplicates. In this experiment, we assume that documents identified as duplicate by both methods are real duplicates and also the documents identified as non-duplicate by both methods are not duplicates. However, there are documents detected by only one method and hence we investigate which documents are correctly labeled in these cases. Also, identification of each document as duplicate or not is a very hard and time consuming process. For this reason, we only deal with the documents that are identified as duplicate by only one method; in other words, we use only the difference sets according to Figure 6.1 and manually examined only such documents. The identification of true duplicates is performed by three annotators. We

prepared a program for annotators to analyze whether a document is a duplicate or not. Before defining how that system works, we continue with our example.

Assume that using the documents of Figure 6.1, I-Match and Tweezer generates duplicate clusters as shown in Figure 6.2.

I-MATCH	TWEEZER
$C_1(d_1, d_4, d_7)$	$C_1(d_3, d_4, d_7)$
$C_2(d_2, d_6)$	$C_2(d_2, d_6, d_{15})$
$C_3(d_5, d_{11}, d_{18})$	$C_3(d_5, d_{18})$
$C_4(d_9, d_{16}, d_{17})$	$C_4(d_8, d_9, d_{16}, d_{19})$
$C_5(d_{10})$	$C_5(d_{10})$
$C_6(d_{12})$	$C_6(d_{12})$
$C_7(d_{13})$	$C_7(d_{13})$
$C_8(d_{14})$	$C_8(d_{14})$
$C_9(d_{20})$	$C_9(d_{20})$

Figure 6.2: Duplicate clusters generated by I-Match and Tweezer.

In Figure 6.1, d_1 is detected as duplicate by only I-Match, and d_1 is located under cluster C_1 according to Figure 6.2. Our program shows annotators document d_1 side by side with the documents (d_4, d_7) that it resides in the same cluster. By analyzing (d_4, d_7), annotators decide whether d_1 is a duplicate or unique (not-duplicate). If d_1 is identified as a duplicate document by the annotator, then it is a missed duplicate for Tweezer. If d_1 is identified as a unique document by the annotator, then it is a false duplicate for I-Match. In our example assume that documents d_3, d_{11} and d_{15} are identified as duplicates by annotators. According to this example our effectiveness measures are calculated as follows.

I-Match:

$$Miss\ Rate = M = \frac{2}{11} = 0.18.\ (d_3 \text{ and } d_{15} \text{ are missed})$$

$$\textit{False Alarm Rate} = F = \frac{2}{9} = 0.22.\ (d_1 \text{ and } d_{17} \text{ are false alarms})$$

$$C_{dup} = 13.29 * 0.22 + 0.18 = 3.10.$$

$$\textit{Precision} = P = \frac{9}{11} = 0.82.$$

$$\textit{Recall} = R = \frac{9}{11} = 0.82.$$

$$F_1 \textit{ Measure} = F = \frac{2 * 0.82 * 0.82}{0.82 + 0.82} = 0.82.$$

Tweezer:

$$\textit{Miss Rate} = M = \frac{1}{11} = 0.09.\ (d_{11} \text{ is missed})$$

$$\textit{False Alarm Rate} = F = \frac{2}{9} = 0.22.\ (d_8 \text{ and } d_{19} \text{ are false alarms})$$

$$C_{dup} = 13.29 * 0.22 + 0.09 = 3.01.$$

$$\textit{Precision} = P = \frac{10}{12} = 0.83.$$

$$\textit{Recall} = R = \frac{10}{11} = 0.91.$$

$$F_1 \textit{ Measure} = F = \frac{2 * 0.83 * 0.91}{0.83 + 0.91} = 0.87.$$

In the case of efficiency experiments we used runtime performance as the evaluation measure. For this reason duplicate processing time of each algorithm is recorded and compared with each other.

6.2 Effectiveness Results

For effectiveness experiments we run each algorithm using two types of test sets as described earlier. False alarm probability and miss probability values and their corresponding C_{dup} values for each algorithm are calculated according to annotators' evaluations. The effectiveness results for C_{dup} measure are given in Table 6.2 and Table 6.6.

Table 6.2: Effectiveness results for C_{dup} measure using Test Collection A *

Test Set Name	I-Match			Tweezer		
	False Alarm	Miss Rate	C_{dup}	False Alarm	Miss Rate	C_{dup}
TestSet1	0.00	15.00	0.15	0.09	0.00	0.01
TestSet2	0.09	45.71	0.47	0.27	8.57	0.12
TestSet3	0.21	3.68	0.06	0.21	0.92	0.04
TestSet4	0.00	18.28	0.18	0.19	6.45	0.09
TestSet5	0.00	30.99	0.31	0.09	2.82	0.04
TestSet6	0.00	10.00	0.10	0.32	2.22	0.07
TestSet7	0.05	18.75	0.19	0.09	2.08	0.03
TestSet8	0.18	37.10	0.40	0.09	6.45	0.08
TestSet9	0.14	31.71	0.34	0.18	9.76	0.12
TestSet10	0.00	15.28	0.15	0.09	5.56	0.07
TestSet11	0.19	9.43	0.12	0.09	0.00	0.01
TestSet12	0.10	8.82	0.10	0.00	0.98	0.01
TestSet13	0.14	8.70	0.11	0.19	2.48	0.05
TestSet14	0.09	46.15	0.47	0.09	15.39	0.17
TestSet15	0.05	3.56	0.04	0.10	2.14	0.03
TestSet16	0.05	3.29	0.04	0.10	2.47	0.04
TestSet17	0.05	12.93	0.14	0.00	0.00	0.00
TestSet18	0.49	8.85	0.10	0.00	0.89	0.01
TestSet19	0.00	2.63	0.03	0.00	0.66	0.01
TestSet20	0.00	18.87	0.19	0.00	1.89	0.02
TestSet21	0.00	27.87	0.28	0.00	0.00	0.00
TestSet22	0.10	7.86	0.09	0.10	0.00	0.01
TestSet23	0.00	1.94	0.02	0.00	0.97	0.01
TestSet24	0.05	13.24	0.14	0.00	1.47	0.01

Test Set Name	I-Match			Tweezer		
	False Alarm	Miss Rate	C_{dup}	False Alarm	Miss Rate	C_{dup}
TestSet25	0.00	3.33	0.03	0.00	0.00	0.00
TestSet26	0.00	5.56	0.06	0.00	0.00	0.00
TestSet27	0.00	14.00	0.14	0.00	0.00	0.00
TestSet28	0.00	4.18	0.04	0.05	0.00	0.01
TestSet29	0.00	10.35	0.10	0.00	0.00	0.00
TestSet30	0.11	4.19	0.06	0.00	0.60	0.01
Average	0.07	14.74	0.16	0.08	2.49	0.04

*False alarm and miss rate values are multiplied by 10^2.

The first set of experiments show that Tweezer is more effective than I-Match in most cases of Test Collection A. The results of Table 6.2 are summarized in Table 6.3. The C_{dup} magnitudes of I-Match and Tweezer from TestSet1 to TestSet30 are depicted in Figure 6.3. (Similar figures of comparisons for false alarm and miss rate are given in Figure A.1 and Figure A.2 in Appendix B.)

Table 6.3: Summarized results for C_{dup} measure using Test Collection A *

		Min	Max	Average	Median	Standard Deviation
I-Match	**False Alarm**	0.00	0.49	0.07	0.05	0.10
	Miss Rate	1.94	46.15	14.74	10.18	12.35
	C_{dup}	0.02	0.47	0.16	0.12	0.13
Tweezer	**False Alarm**	0.00	0.32	0.08	0.09	0.09
	Miss Rate	0.00	15.39	2.49	0.98	3.55
	C_{dup}	0.00	0.17	0.04	0.01	0.04

*False alarm and miss rate values are multiplied by 10^2.

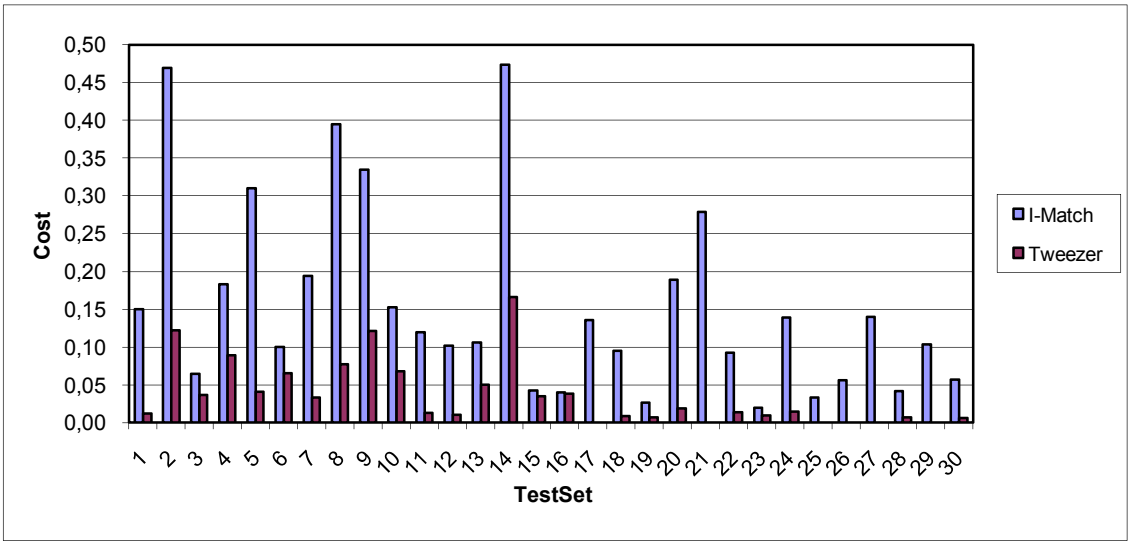

Figure 6.3: Cost comparisons of I-Match and Tweezer using Test Collection A.

The results of the experiments using precision, recall and F_1 measure with Test Collection A are given in Table 6.4.

Table 6.4: Effectiveness results for F_1 measure using Test Collection A

TestSet Name	I-Match			Tweezer		
	Precision	Recall	F_1 Measure	Precision	Recall	F_1 Measure
TestSet1	1.000	0.850	0.919	0.952	1.000	0.976
TestSet2	0.905	0.543	0.679	0.842	0.914	0.877
TestSet3	0.987	0.963	0.975	0.988	0.991	0.989
TestSet4	1.000	0.817	0.899	0.956	0.935	0.946
TestSet5	1.000	0.690	0.817	0.972	0.972	0.972
TestSet6	1.000	0.900	0.947	0.926	0.978	0.951
TestSet7	0.987	0.813	0.891	0.979	0.979	0.979
TestSet8	0.907	0.629	0.743	0.967	0.935	0.951
TestSet9	0.903	0.683	0.778	0.902	0.902	0.902
TestSet10	1.000	0.847	0.917	0.971	0.944	0.958
TestSet11	0.960	0.906	0.932	0.981	1.000	0.991
TestSet12	0.989	0.912	0.949	1.000	0.990	0.995
TestSet13	0.980	0.913	0.945	0.975	0.975	0.975
TestSet14	0.875	0.538	0.667	0.917	0.846	0.880
TestSet15	0.996	0.964	0.980	0.993	0.979	0.986
TestSet16	0.996	0.967	0.981	0.992	0.975	0.983
TestSet17	0.990	0.871	0.927	1.000	1.000	1.000

TestSet Name	I-Match			Tweezer		
	Precision	Recall	F_1 Measure	Precision	Recall	F_1 Measure
TestSet18	0.995	0.912	0.952	1.000	0.991	0.996
TestSet19	1.000	0.974	0.987	1.000	0.993	0.997
TestSet20	1.000	0.811	0.896	1.000	0.981	0.990
TestSet21	1.000	0.721	0.838	1.000	1.000	1.000
TestSet22	0.992	0.921	0.956	0.993	1.000	0.996
TestSet23	1.000	0.981	0.990	1.000	0.990	0.995
TestSet24	0.992	0.868	0.925	1.000	0.985	0.993
TestSet25	1.000	0.967	0.983	1.000	1.000	1.000
TestSet26	1.000	0.944	0.971	1.000	1.000	1.000
TestSet27	1.000	0.860	0.925	1.000	1.000	1.000
TestSet28	1.000	0.958	0.979	0.998	1.000	0.999
TestSet29	1.000	0.897	0.945	1.000	1.000	1.000
TestSet30	0.996	0.958	0.977	1.000	0.994	0.997
Average	0.982	0.853	0.909	0.977	0.975	0.976

The results of Table 6.4 are summarized in Table 6.5. The F_1 measure magnitudes of I-Match and Tweezer from TestSet1 to TestSet30 are depicted in Figure 6.4. (Similar figures of comparisons for precision and recall are given in Figure A.5 and Figure A.6 in Appendix C.)

Table 6.5: Summarized results for F_1 measure using Test Collection A

		Min	Max	Average	Median	Standard Deviation
I-Match	**Precision**	0.875	1.000	0.982	0.996	0.034
	Recall	0.538	0.981	0.853	0.899	0.124
	F_1 Measure	0.667	0.990	0.909	0.939	0.087
Tweezer	**Precision**	0.842	1.000	0.977	0.993	0.036
	Recall	0.846	1.000	0.975	0.990	0.036
	F_1 Measure	0.877	1.000	0.976	0.991	0.034

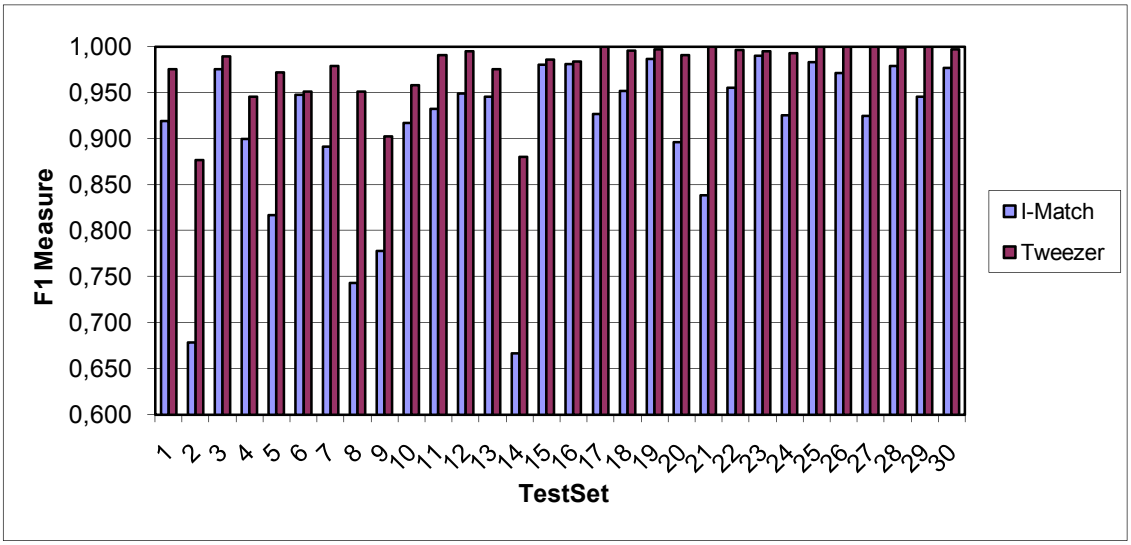

Figure 6.4: F_1 measure comparisons of I-Match and Tweezer using Test Collection A.

The results of the experiments for C_{dup} measure with Test Collection B are shown in Table 6.6.

Table 6.6: Effectiveness results for C_{dup} measure using Test Collection B *

Test Set Name	I-Match			Tweezer		
	False Alarm	Miss Rate	C_{dup}	False Alarm	Miss Rate	C_{dup}
TestSet31	0.00	0.00	0.00	0.08	0.00	0.01
TestSet32	0.00	0.00	0.00	0.00	15.39	0.15
TestSet33	0.00	50.00	0.50	0.00	0.00	0.00
TestSet34	0.00	12.50	0.13	0.00	25.00	0.25
TestSet35	0.08	0.00	0.01	0.00	0.00	0.00
TestSet36	0.59	0.00	0.08	0.00	0.00	0.00
TestSet37	0.00	7.41	0.07	0.00	7.41	0.07
TestSet38	1.60	6.45	0.28	0.74	4.84	0.15
TestSet39	1.31	13.73	0.31	0.29	1.96	0.06
TestSet40	0.04	1.16	0.02	0.00	0.58	0.01
TestSet41	0.05	0.31	0.01	0.00	0.00	0.00
TestSet42	0.38	2.90	0.08	0.00	1.45	0.01
TestSet43	0.00	2.12	0.02	0.10	0.71	0.02
TestSet44	0.04	0.00	0.01	0.00	21.43	0.21
TestSet45	0.05	59.06	0.60	0.31	3.94	0.08
TestSet46	0.13	46.19	0.48	0.22	3.39	0.06
TestSet47	0.18	52.68	0.55	0.09	4.46	0.06
TestSet48	0.05	51.31	0.52	0.00	4.58	0.05
TestSet49	0.09	40.76	0.42	0.09	4.62	0.06

Test Set Name	I-Match			Tweezer		
	False Alarm	Miss Rate	C_{dup}	False Alarm	Miss Rate	C_{dup}
TestSet50	0.44	52.27	0.58	0.40	5.00	0.10
TestSet51	0.09	51.39	0.53	0.61	5.09	0.13
TestSet52	0.00	45.88	0.46	0.82	4.12	0.15
TestSet53	0.09	48.62	0.50	0.35	3.67	0.08
TestSet54	0.17	49.76	0.52	0.31	2.90	0.07
TestSet55	0.12	0.00	0.02	0.08	0.00	0.01
TestSet56	0.17	2.41	0.05	0.08	2.41	0.04
TestSet57	0.37	18.87	0.24	0.00	0.00	0.00
TestSet58	0.38	3.92	0.09	0.00	1.31	0.01
TestSet59	0.07	0.73	0.02	0.78	0.27	0.11
TestSet60	0.04	3.61	0.04	0.00	0.80	0.01
Average	0.22	20.80	0.24	0.18	4.18	0.07

*False alarm and miss rate values are multiplied by 10^2.

According to the results of Table 6.6, Tweezer is more effective than I-Match in most cases of Test Collection B. However, in the experiments with Test Collection B, I-Match is more effective than that of the experiments with Test Collection A. For example, I-Match is more effective than Tweezer in five test sets (TestSet31, TestSet32, TestSet34, TestSet44 and TestSet59). The results of Table 6.6 are summarized in Table 6.7. The C_{dup} magnitudes of I-Match and Tweezer from TestSet31 to TestSet60 are given in Figure 6.5. (Similar figures of comparisons for false alarm and miss rate are given in Figure A.3 and Figure A.4 in Appendix B.)

Table 6.7: Summarized results for C_{dup} measure using Test Collection B *

		Min	Max	Average	Median	Standard Deviation
I-Match	**False Alarm**	0.00	1.60	0.22	0.09	0.36
	Miss Rate	0.00	59.06	20.80	6.93	22.64
	C_{dup}	0.00	0.60	0.24	0.11	0.23
Tweezer	**False Alarm**	0.00	0.82	0.18	0.08	0.25
	Miss Rate	0.00	25.00	4.18	2.66	5.97
	C_{dup}	0.00	0.25	0.07	0.06	0.07

*False alarm and miss rate values are multiplied by 10^2.

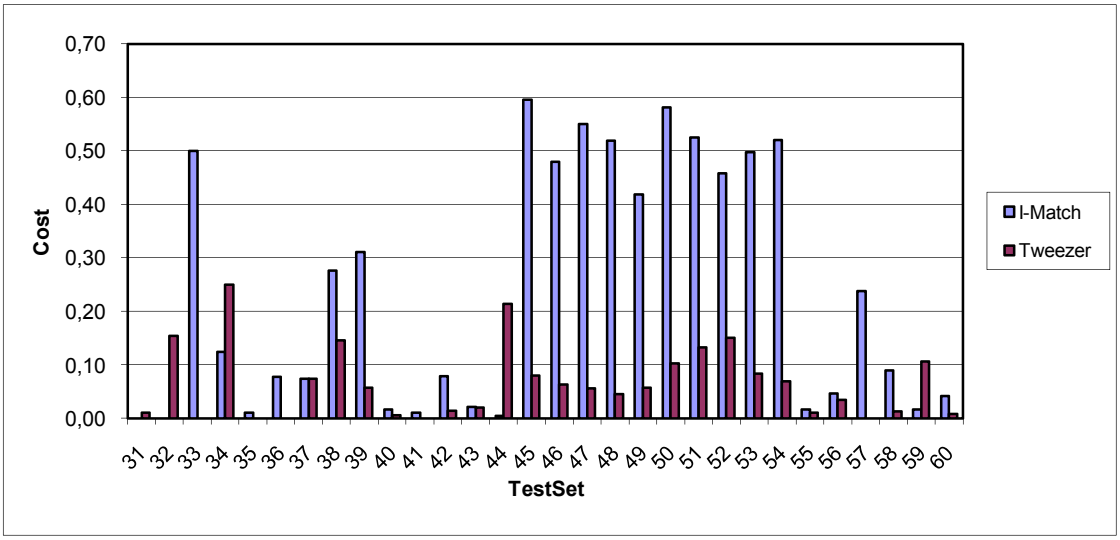

Figure 6.5: The cost comparisons of I-Match and Tweezer using Test Collection B.

The results of the experiments using precision, recall and F_1 measure with Test Collection B are given in Table 6.8.

Table 6.8: Effectiveness results for F_1 measure using Test Collection B

TestSet Name	I-Match			Tweezer		
	Precision	Recall	F_1 Measure	Precision	Recall	F_1 Measure
TestSet31	1.000	1.000	0.875	1.000	1.000	0.933
TestSet32	1.000	1.000	1.000	0.846	1.000	0.917
TestSet33	1.000	0.500	1.000	1.000	0.667	1.000
TestSet34	1.000	0.875	1.000	0.750	0.933	0.857
TestSet35	0.947	1.000	1.000	1.000	0.973	1.000
TestSet36	0.983	1.000	1.000	1.000	0.991	1.000
TestSet37	1.000	0.926	1.000	0.926	0.962	0.962
TestSet38	0.598	0.935	0.766	0.952	0.730	0.849
TestSet39	0.579	0.863	0.877	0.980	0.693	0.926
TestSet40	0.994	0.988	1.000	0.994	0.991	0.997
TestSet41	0.998	0.997	1.000	1.000	0.998	1.000
TestSet42	0.937	0.971	1.000	0.986	0.954	0.993
TestSet43	1.000	0.979	0.993	0.993	0.989	0.993
TestSet44	0.933	1.000	1.000	0.786	0.966	0.880
TestSet45	0.990	0.409	0.972	0.961	0.579	0.966
TestSet46	0.977	0.538	0.979	0.966	0.694	0.972

TestSet Name	I-Match			Tweezer		
	Precision	Recall	F_1 Measure	Precision	Recall	F_1 Measure
TestSet47	0.964	0.473	0.991	0.955	0.635	0.973
TestSet48	0.993	0.487	1.000	0.954	0.654	0.977
TestSet49	0.986	0.592	0.991	0.954	0.740	0.972
TestSet50	0.913	0.477	0.959	0.950	0.627	0.954
TestSet51	0.981	0.486	0.936	0.949	0.650	0.943
TestSet52	1.000	0.541	0.907	0.959	0.702	0.932
TestSet53	0.982	0.514	0.963	0.963	0.675	0.963
TestSet54	0.963	0.502	0.966	0.971	0.660	0.969
TestSet55	0.945	1.000	0.963	1.000	0.972	0.981
TestSet56	0.953	0.976	0.976	0.976	0.964	0.976
TestSet57	0.827	0.811	1.000	1.000	0.819	1.000
TestSet58	0.942	0.961	1.000	0.987	0.951	0.993
TestSet59	0.999	0.993	0.990	0.997	0.996	0.994
TestSet60	0.996	0.964	1.000	0.992	0.980	0.996
Average	0.946	0.792	0.970	0.958	0.838	0.962

The results of Table 6.8 are summarized in Table 6.9. The F_1 measure magnitudes of I-Match and Tweezer from TestSet31 to TestSet60 are depicted in Figure 6.6. (Similar figures of comparisons for precision and recall are given in Figure A.7 and Figure A.8 in Appendix C.)

Table 6.9: Summarized results for F_1 measure using Test Collection B

		Min	Max	Average	Median	Standard Deviation
I-Match	**Precision**	0.579	1.000	0.946	0.983	0.102
	Recall	0.409	1.000	0.792	0.931	0.227
	F_1 Measure	0.767	1.000	0.970	0.992	0.051
Tweezer	**Precision**	0.750	1.000	0.958	0.974	0.060
	Recall	0.579	1.000	0.838	0.942	0.153
	F_1 Measure	0.849	1.000	0.962	0.973	0.041

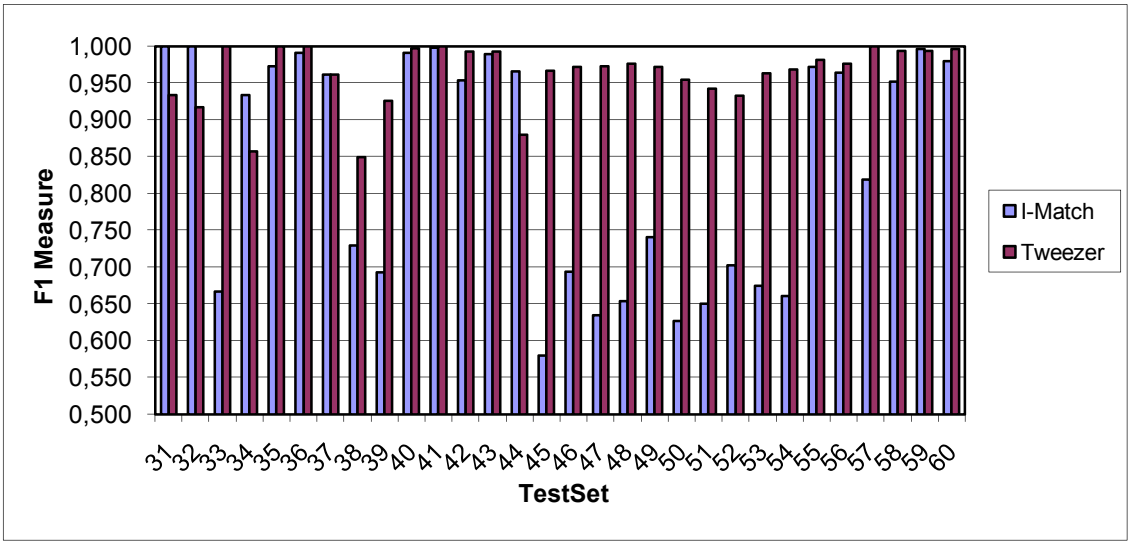

Figure 6.6: F_1 measure comparisons of I-Match and Tweezer using Test Collection B.

We conducted pair-wise comparisons on the cost values of I-Match and Tweezer in two sets of experiments for effectiveness issues in order to see whether Tweezer's results are statistically significantly smaller than those of I-Match. We applied one sided matched pair t-tests using alpha level of 0.05 for significance to the results of effectiveness experiments and corresponding p-values of these statistical tests are given in Table 6.10.

Table 6.10: p-values of t-tests for effectiveness experiments

	p-Values			p-Values	
	Test Collection A	**Test Collection B**		**Test Collection A**	**Test Collection B**
False Alarm	0.407	0.287	**Precision**	0.161	0.050
Miss Rate	$1.21 * 10^{-7}$	$3.38 * 10^{-4}$	**Recall**	$1.21 * 10^{-7}$	$3.38 * 10^{-4}$
C_{dup}	$1.65 * 10^{-7}$	$1.45 * 10^{-4}$	**F_1 Measure**	$1.31 * 10^{-6}$	$7.35 * 10^{-5}$

These p-values imply that Tweezer is statistically significantly more effective than I-Match.

6.3 Efficiency Results

For the efficiency experiments we run each algorithm on a computer which has 2.5 GHz Intel Xeon CPU with 8 cores and 4 GB of memory. Each algorithm is run on the computer and its runtime is recorded. We record two different runtime for I-Match, because I-Match can be implemented in two different ways. As stated previously, I-Match chooses the terms that will be used during the comparison with the help of IDF values of terms. In order to perform these, IDF values of all terms in the collection must be calculated. There are two options to calculate these IDF values. The first option is to use IDF values of a generic collection and other option is to recalculate them in each collection [CHO2002]. The second approach increases the overall runtime of I-Match algorithm.

By considering these issues, we record two execution times either IDF calculation time is included or excluded. In each step of the test, we double the collection size. The execution times of each algorithm using Test Collection C is given in Table 6.11.

Table 6.11: Duplicate processing times of I-Match and Tweezer using Test Collection C

Test Set Name	No. of Docs	Time (msec)			Approximate Performance Increase (%) *
		I-Match		Tweezer	
		IDF included	IDF excluded		
TestSet1	6,250	7,000	4,235	3,921	7
TestSet2	12,500	12,937	7,828	6,568	16
TestSet3	25,000	23,938	14,282	12,312	14
TestSet4	50,000	47,875	28,500	24,843	13
TestSet5	100,000	98,328	57,579	48,266	16
TestSet6	200,000	196,062	120,297	98,657	18
TestSet7	400,000	394,640	242,531	200,141	17

* With respect to the IDF excluded case.

According to the execution times given in Table 6.11, it is obvious that Tweezer is faster than I-Match in the range of 7% to 18%.

6.4 Chapter Summary

In this chapter, we show that using named entities in duplicate document detection is both efficient and effective. We separated our experiments into two categories. In the effectiveness experiments, we prepared two test collections: Test Collection A, B. One sided matched paired t-tests are performed on the results of effectiveness tests and the results show that cost values of Tweezer is statistically significantly smaller than those of I-Match. Finally, we performed experiments to see duplicate processing times of each algorithm using Test Collection C. According to the results of efficiency experiments Tweezer decreases the duplicate processing time at least 7% and up to 18%. The outcomes of experimental results may be summarized as using named entities in duplicate document detection increases the effectiveness and efficiency of duplicate detection process in news documents.

Chapter 7

Conclusions

In this thesis, we propose a new near-duplicate document detection algorithm called Tweezer. It uses signatures generated by using named entity centered word sequences for the comparison of documents. For this purpose, we propose a new signature generation technique using *5NE5* shingles which uses named entities together with five (5) preceding and five (5) succeeding words with respect to named entities (NE). Therefore, this approach is referred as a named entity-based shingle. All document shingles are used to generate a document signature using SHA1. Two documents sharing a common signature value are considered as near-duplicate.

7.1 Discussion of Experimental Results

We evaluated Tweezer using multiple test sets and in these experiments we used I-Match as our baseline. I-Match chooses terms to be used in the comparison of documents according to IDF values of terms. This approach avoids the use of most and least frequent terms, since they do not distinguish documents from each other. After significant terms are decided for a document, document signature is generated according to hash values of all significant tokens by using SHA1. Any two documents containing the same signature are considered as duplicate of each other.

We conducted experiments to evaluate both effectiveness and efficiency of Tweezer and compare them with those of I-Match. For this purpose, we created sixty test sets to evaluate effectiveness of both algorithms. We divided experiments into two parts. Firstly, we performed experiments on thirty test sets with a total of 67,500 documents and each test consisting of 2,250 documents (Test Collection A). Documents in each category are distributed in each test set as proportional to the size of that category in the database. After that we performed experiments on another thirty test sets with a total of 75,000 documents and each test consisting of 2,500 documents (Test Collection B). These test sets are created by using documents belonging to the same category for each test set. The results show that cost values of Tweezer is statistically significantly smaller than those of I-Match.

In order to evaluate efficiency of both algorithms we created seven test sets (Test Collection C) consisting of collections in size ranging from 6,250 to 400,000 documents. According to efficiency experiments Tweezer decreases the duplicate processing time at least 7% and up to 18% with respect to I-Match.

The experimental results show that using named entities in near-duplicate document detection increases the effectiveness and efficiency of this process in news documents.

7.2 Contributions of the Study

In this study we propose a novel approach that uses named entities for duplicate news detection. To the best of our knowledge, there is no previous study that uses named entities for duplicate document detection. We evaluate our approach with Turkish news documents. This thesis is the first duplicate detection study on Turkish. We show one of the ways of using named entities for the purpose of duplicate elimination with an algorithm called Tweezer. We hope that this study will help researchers to develop new directions for duplicate detection using named entities.

References

[ALF2002] E. Alfonseca and S. Manandhar. An unsupervised method for general named entity recognition and automated concept discovery. In *Proceedings of the 1st International Conference on General WordNet*, 2002.

[ASA2003] M. Asahara and Y. Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pp. 8-15, 2003.

[BAG2009] Ö. Bağlıoğlu. New event detection using chronological term ranking. Master Thesis, Computer Engineering Department, Bilkent University, May 2009.

[BIK1997] D. M. Bikel, S. Miller, R. Schwartz, R. Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of 5th Conference on Applied Natural Language Processing*, pp. 194-201, 1997.

[BOR1998] A. Borthwick, J. Sterling, E. Agichteini R. Grishman. NYU: description of the MENE named entity system used in MUC-7. In *Proceedings of the 7th Message Understanding Conference*, 1998.

[BRI1995] S. Brin, J. Davis, H. Garcia-Molina. Copy detection mechanisms for digital documents. In *Proceedings of the ACM SIGMOD Annual Conference*, pp. 398-409, 1995.

[BRI1998] S. Brin. Extracting patterns and relations from the World Wide Web. In *Proceedings of the WebDB Workshop at 6th International Conference on Extending Database Technology*, pp. 172-173, 1998.

[BRO1997] A. Z. Broder, S. C. Glassman, M. S. Manasse, G. Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, Vol. 29, No. 8-13, pp. 1157-1166, 1997.

[BUC2000] C. Buckley, J. Walz, C. Cardie, S. Mardis, M. Mitra, A. Mitra, D. Pierce, K. Wagstaff. The Smart/Empire TIPSTER IR system. TIPSTER Phase III Proceedings, pp. 107-121, Morgan Kaufmann, 2000.

[CAN2008a] F. Can, S. Kocberber, O. Baglioglu, S. Kardas, H. C. Ocalan, E. Uyar. Bilkent news portal: a personalizable system with new event detection and tracking capabilities. In *Proceedings of 31th annual international ACM SIGIR conference on research and development in information retrieval*, p. 805, 2008.

[CAN2008b] F. Can, S. Kocberber, E. Balcik, C. Kaynak, H. C. Ocalan, O. M. Vursavas. Information retrieval on Turkish texts. *Journal of the American Society for Information Science and Technology*. Vol. 59, No. 3, pp. 407-421, 2008.

[CAN2009] F. Can, S. Koçberber, Ö. Bağlıoğlu, G. Ercan, S. Kardaş, H. Ç. Öcalan, E. Uyar, L. Koç. Haber portallarında yenilikçi yaklaşımlar. Akademik Bilişim 2009 (in publication).

[CHO2002] A. Chowdury, O. Frieder, D. Grossman, M. C. McCabe. Collection statistics for fast duplicate document detection. *ACM Transactions on Information Systems*, Vol. 20, No. 2, pp. 171-191, 2002.

[CUC2001] A. Cucchiarelli and P. Velardi. Unsupervised named entity recognition using syntactic and semantic contextual evidence. *Computational Linguistics*, Vol. 27 No. 1, pp. 123-131, Cambridge: MIT Press, 2001.

[FAR2000] D. Farmakiotou, V. Karkaletsis, J. Koutsias, G. Sigletos, C. D. Spyropoulos , P. Stamatopoulos. Rule-Based named entity recognition for Greek finantial texts. In *Proceedings of the Workshop on Computational Lexicography and Multimedia Dictionaries*, pp. 75-78, 2000.

[FET2003] D. Fetterly, M. Manasse, M. Najork. On the evolution of clusters of near-duplicate web pages. In *Proceedings of the 1st Conference on Latin American Web Congress*, p. 37, 2003.

[GRI1996] R. Grishman and B. Sundheim. Message Understanding Conference – 6: A Brief History. In *Proceedings of the 16th International Conference on Computational Linguistics*, pp. 466-471, 1996.

[HEI1996] N. Heintze. Scalable document fingerprinting. In *Proceedings of USENIX Work-shop on Electronic Commerce*, 1996.

[HEN2006] J. Heng and R. Grishman. Data selection in semi-supervised learning for name tagging. In *Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics*, 2006.

[HOA2003] T. Hoad and J. Zobel. Methods for identifying versioned and plagiarized documents. *Journal of the American Society for Information Science and Technology*, Vol. 54, No. 3, pp. 203-215, 2003.

[KAR2009] S. Kardaş. New event detection and tracking in Turkish. Master Thesis, Computer Engineering Department, Bilkent University, May 2009.

[KUM2004] G. Kumaran and J. 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*, pp. 297-304, 2004.

[LIU2007] K. Liu, W. Meng, J. Qiu, C. Yu, V. Raghavan, Z. Wu, Y. Lu, H. He, H. Zhao. AllInOneNews: development and evaluation of a large-scale news metasearch engine. In *Proceedings of the 2007 ACM SIGMOD international conference on management of data*, pp. 1017-1028, 2007.

[MAN1994] U. Manber. Finding similar files in large file system. In *Proceedings of USENIX Winter Technical Conferences*, 1994.

[MCC2003] A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, features induction and web-enhanced lexicons. In *Proceedings of the 7th Conference on Natural Language Learning*, pp. 188-191, 2003.

[NIS1995] NIST. Secure Hash Standard. U.S. Department of Commerce/National Institute of Standards and Technology, FIBS PUB 180-1, 1995.

[OCA2009] H. Ç. Öcalan. Bilkent news portal: a system with new event detection and tracking capabilities. Master Thesis, Computer Engineering Department, Bilkent University, May 2009.

[PAP1999] R. Papka. Online new event detection, clustering and tracking. Phd Dissertation UMASS, 1999.

[PAS2006] M. Pasca, D. Lin, J. Bigham, A. Lifchits, A. Jain. Organizing and searching the World Wide Web of facts – step one: the one-million fact extraction challenge. In *Proceedings of the National Conference on Artificial Intelligence*, 2006.

[POI2001] T. Poibeau and L. Kosseim. Name extraction from non-journalistic texts. In *Proceedings of Computational Linguistics in the Netherlands*, pp. 144-157, 2001.

[RAU1991] L. F. Rau. Extracting company names from text. In *Proceedings of 7th Conference on Artificial Intelligence Applications of IEEE*, pp. 29-32, 1991.

[RIJ1979] C. J. Van Rijsbergen. Information Retrieval, 2nd ed. London: Butterworths, 1979.

[RIL1999] E. Riloff and R. Jones. Learning dictionaries for information extraction using multi-level bootstrapping. In *Proceedings of the 16thNational Conference on Artificial Intelligence*, pp. 474-479, 1999.

[SAL1975] G. Salton, C. S. Yang, A. Wong. A vector space model for automatic indexing. *Communications of the ACM*, Vol. 18, No. 11, pp. 613-620, 1975.

[SEK1998] S. Sekine. Nyu: description of the Japanese NE system used for Met-2. In *Proceedings of the 7th Message Understanding Conference*. 1998.

[SHI1995] N. Shivakumar and H. Garcia-Molina. SCAM: a copy detection mechanism for digital documents. In *Proceedings of the 2nd Annual Conference on the Theory and Practice of Digital Libraries*, 1995.

[SHI1996] N. Shivakumar and H. Garcia-Molina. Building a scalable and accurate copy detection mechanism. In *Proceedings of the 1st ACM International Conference on Digital Libraries*, pp. 160-168, 1996.

[SHI1998] N. Shivakumar and H. Garcia-Molina. Finding near-replicas of documents on the web. In *Proceedings of the International Workshop on the Web and Databases*, 1998.

[SHI2004] Y. Shinyama and S. Sekine. Named entity discovery using comparable news articles. In *Proceedings of the 20th International Conference on Computational Linguistics*, 2004.

[TDT2009] TDT Phase 2. Topic Detection and Tracking Phase 2. Retrieved May 13, 2009, from http://projects.ldc.upenn.edu/TDT2/.

[VAR2005] H. R. Varian. Universal access to information. Communications of the ACM, Vol. 48, No. 10, October 2005.

[VEN1880] J. Venn. On the diagrammatic and mechanical representation of propositions and reasonings. *Dublin Philosophical Magazine and Journal of Science*, Vol. 9, pp. 1-18, 1880.

[WAN1992] L. Wang, W. Li, C. Chang. Recognizing unregistered names for Mandarin Word Identification. In *Proceedings of the 14th Conference on Computational Linguistics*, pp. 1239-1243, 1992.

[WIK2009] Wikipedia. Named entity recognition. Retrieved May 11, 2009, from http://en.wikipedia.org/wiki/Named_entity_recognition.

[WOL1995] F. Wollinski, F. Vichot, B. Dillet. Automaic processing of proper names in texts. In *Proceedings of the 7th Conference on European Chapter of the Association for Computational Linguistics*, pp. 23-30, 1995.

[ZHA2004] L. Zhang, Y. Pan, T. 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*, pp. 281-288, 2004.

Appendices

Appendix A: Rule Lists Used in TuNER

Table A.1: Prefix rule lists for person names used in TuNER

Bay	Bayan	Başbakan	Başbakanı	Başkanvekili
Başkanı	Cumhurbaşkanı	Doktor	Kaymakam	Mareşal
Müdürü	Ortağı	Sayın	Uzmanı	Vali
Yüzbaşı	-	-	-	-

Table A.2: Suffix rule lists for person names used in TuNER

Bey	Efendi	Hanım	Hoca	Kaptan

Table A.3: Suffix rule lists for location names used in TuNER

Abidesi	Anadolu	Anıtı	Bakkaliyesi	Bakkalı
Başbakanı	Beldesi	Boğazı	Bulvarı	Bölgesi
Caddesi	Camii	Dağı	Denizi	Doğu
Geçidi	Gölü	Hamamı	Han	Havaalanı
Havalimanı	Irmağı	Kalesi	Kanalı	Kaplıcaları
Kaplıcası	Karayolu	Kilisesi	Kitabevi	Kulesi
Köprüsü	Körfezi	Köyü	Kırtasiyesi	Lokali
Lokantası	Mahallesi	Manastırı	Merkezi	Meydanı
Misafirhanesi	Müzesi	Nehri	Otel	Oteli
Parkı	Sahne	Sahnesi	Salonu	Sarayı
Sineması	Sitesi	Sokağı	Stadyumu	Stadı
Tepesi	Tesisi	Tesisleri	Tiyatrosu	Türbesi
Yaylası	Yöresi	Çifliği	Ögretmenevi	İlçesi

Table A.4: Suffix rule lists for organization names used in TuNER

Adliyesi	Bakanlığı	Bankası	Başkanlığı	Başsavcılığı
Belediyesi	Birliği	Borsası	Bölümü	Dekanlığı
Derneği	Fakültesi	Hastanesi	Kaymakamlığı	Komutanlığı
Kulübü	Kurulu	Kurumu	Kütüphanesi	Lisesi
Mahkemesi	Meclisi	Merkezi	Müdürlüğü	Müsteşarlığı
Ocağı	Odası	Ofisi	Okulu	Parti
Partisi	Savcılığı	Teşkilatı	Valiliği	Üniversitesi
İdaresi	İlkokulu	İnşaat	Şubesi	-

Appendix B: Pair-wise Comparisons of False Alarm and Miss Rate between Tweezer and I-Match

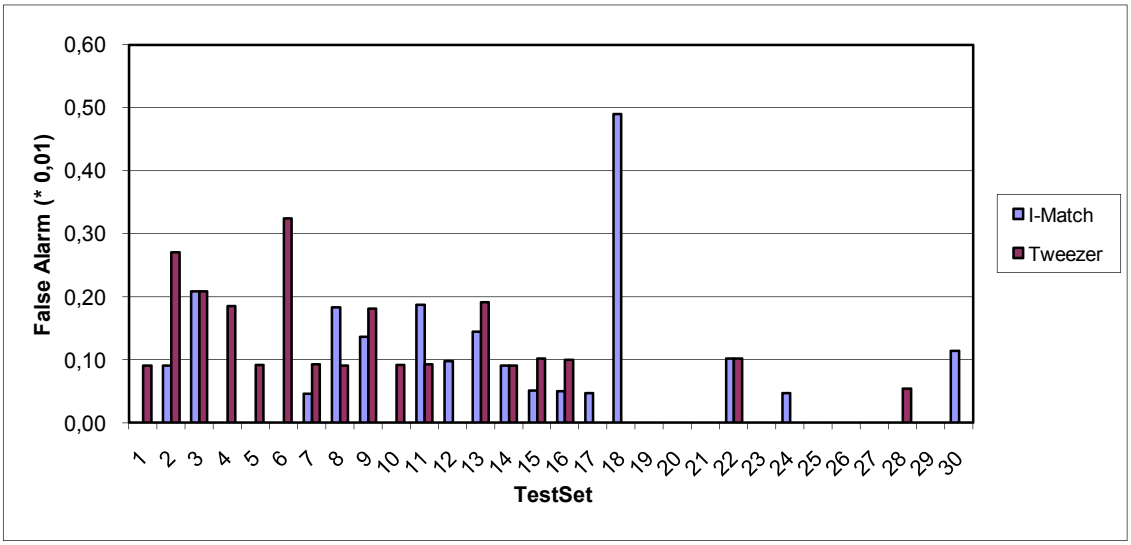

Figure A.1: False alarm comparisons of I-Match and Tweezer using Test Collection A.

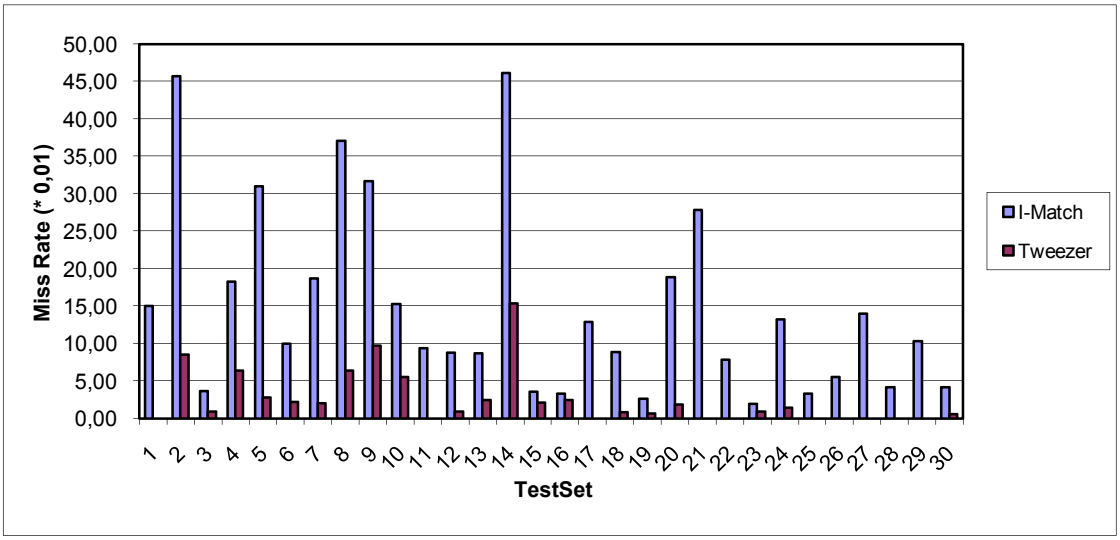

Figure A.2: Miss rate comparisons of I-Match and Tweezer using Test Collection A.

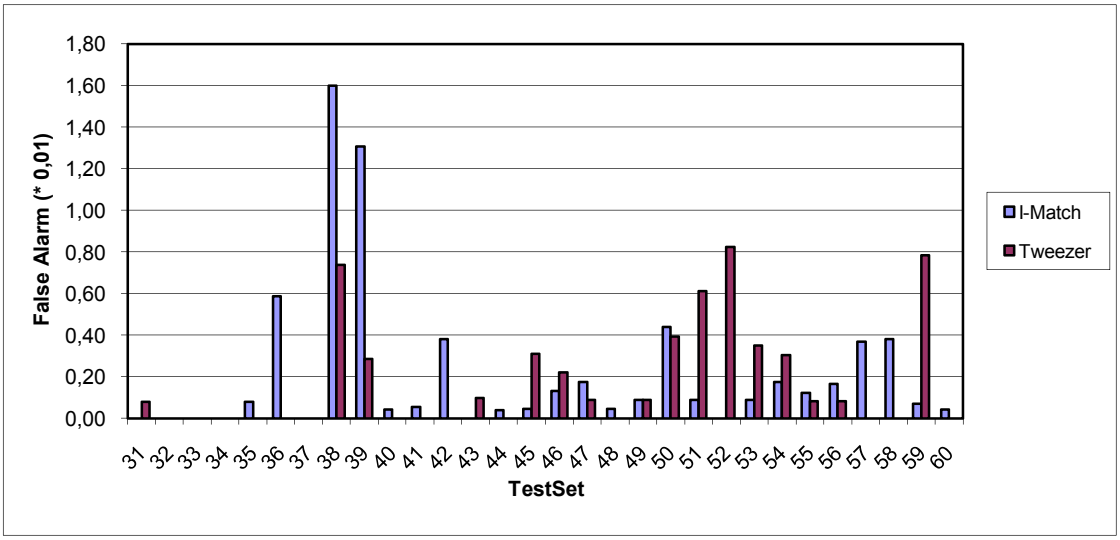

Figure A.3: False alarm comparisons of I-Match and Tweezer using Test Collection B.

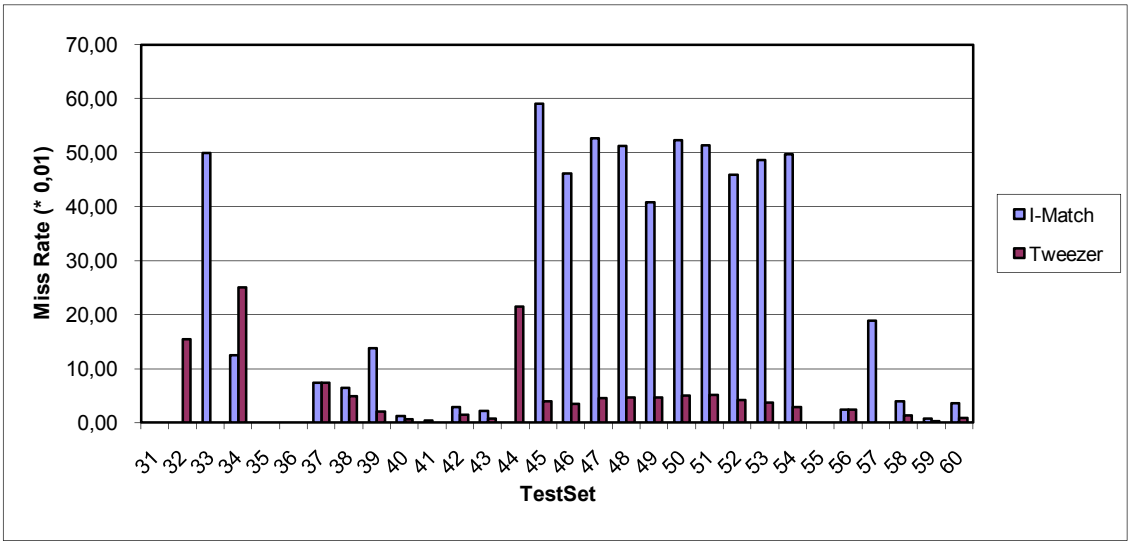

Figure A.4: Miss rate comparisons of I-Match and Tweezer using Test Collection B.

Appendix C: Pair-wise Comparisons of Precision and Recall between Tweezer and I-Match

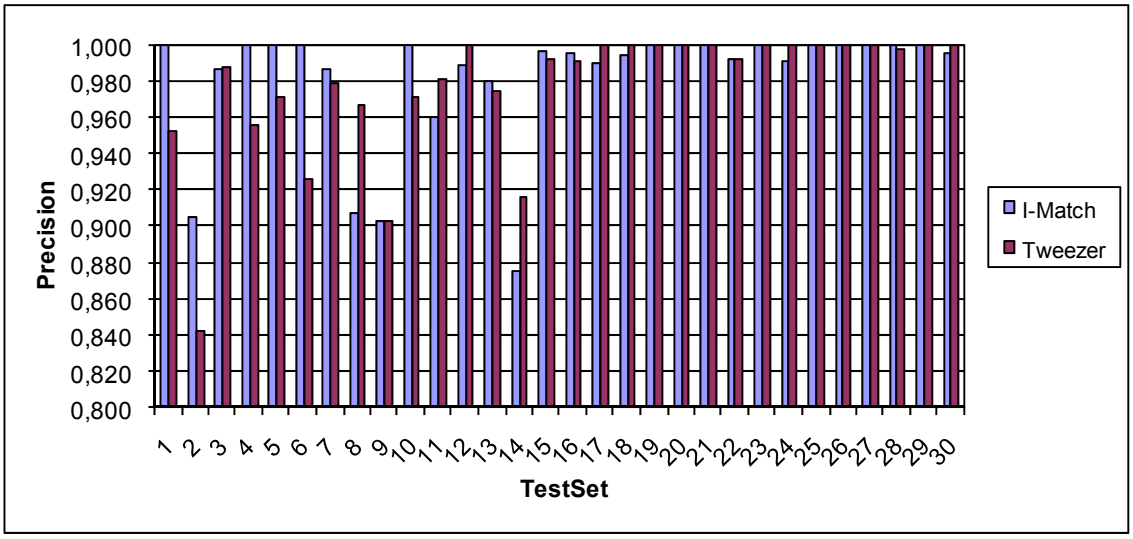

Figure A.5: Precision comparisons of I-Match and Tweezer using Test Collection A.

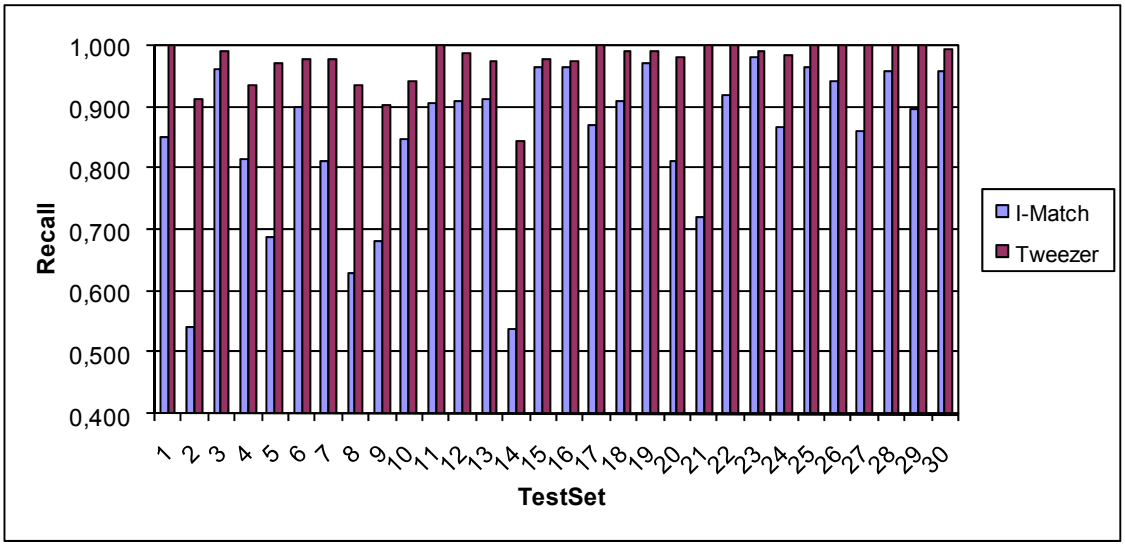

Figure A.6: Recall comparisons of I-Match and Tweezer using Test Collection A.

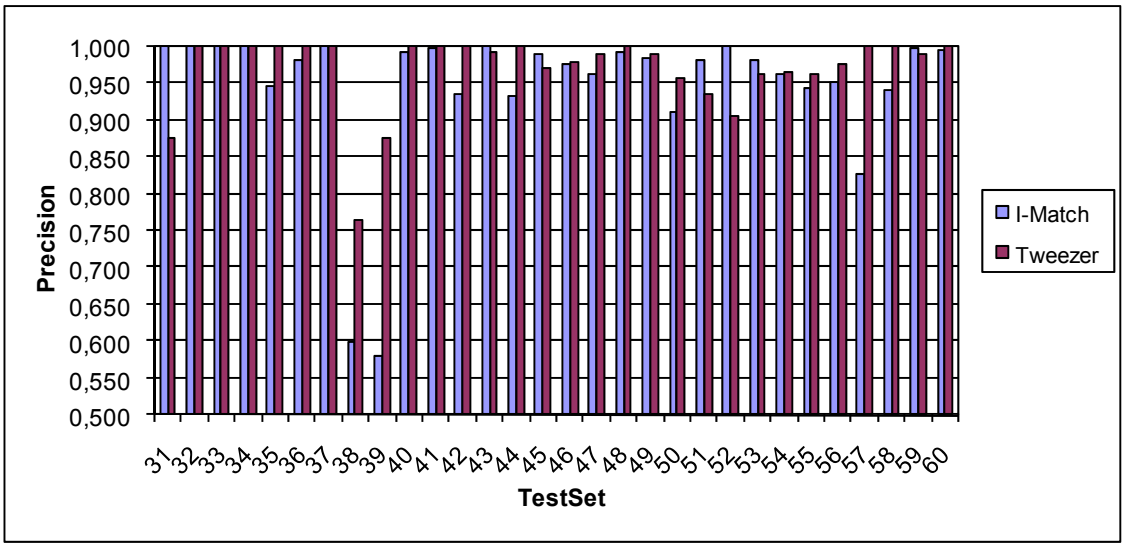

Figure A.7: Precision comparisons of I-Match and Tweezer using Test Collection B.

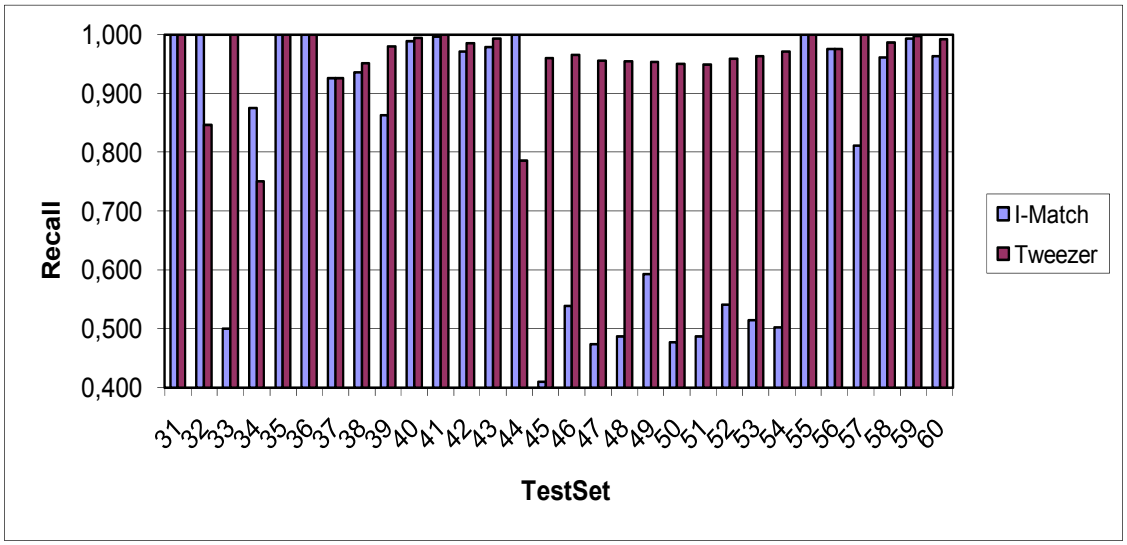

Figure A.8: Recall comparisons of I-Match and Tweezer using Test Collection B.

Appendix D: Near-Duplicate Samples of Tweezer and I-Match

CRR'de müzik başlıyor

Cemal Reşit Rey Konser Salonu (CRR) 11 Ekim'de kapılarını açıyor.

Salon 2008-2009 konser sezonunda 200'e yakın etkinlik ve 2000'in üzerinde sanatçıyla kültür sanat hayatına damgasını vuracak. *CRR, ekim, kasım ve aralık aylarını kapsayan sezonun ilk bölümünde, 60 etkinlik ve 750 sanatçıyla İstanbullu sanatseverlerin karşısına çıkacak. Yehudi Menuhin'in "Gerçekten dinlediğim en mükemmel kemancı" dediği Vadim Repin'le sezona adım atacak CRR konser sezonunda bu yıl dinleme şansı yakalayacağımız isimlerden bazıları şöyle: Chick Corea&John McLaughlin, James Galway&Lady Galway, Giora Feidman, David Russell, Manolo Sanlucar, Lubna Saleme, Ilya Gringolts, Carmen Lundy, Sa Chen, Sharon Isbin, Konstantin Moskovich, Kremerata Baltica, Nigel Kennedy Quintet, Talich Quartet, Strauss Ensemble, CRR İstanbul Senfoni Orkestrası, Yansımalar, Fazıl Say, Ayla Erduran, Cihat Aşkın, Arto Tunç Boyacıyan, Melihat Gülses, Meral Uğurlu ve Nevzat Sümer. Uluslararası müzik arenasında da isminden övgüyle söz ettiren ülkemizin en prestijli konser salonu Cemal Reşit Rey, ocak ayında başlayacak ve 2009 Mayıs sonuna kadar devam edecek olan sezonun 2. yarısında da; ağırlayacağı 1.500'ün üzerinde sanatçı ve ayda gerçekleştireceği 20'nin üzerinde etkinlikle sanatseverlerle buluşacak. Kültür-Sanat*

CRR'de müzik başlıyor

Cemal Reşit Rey Konser Salonu (CRR) 11 Ekim'de kapılarını açıyor. Salon 2008-2009 konser sezonunda 200'e yakın etkinlik ve 2000'in üzerinde sanatçıyla kültür sanat hayatına damgasını vuracak.

CRR, ekim, kasım ve aralık aylarını kapsayan sezonun ilk bölümünde, 60 etkinlik ve 750 sanatçıyla İstanbullu sanatseverlerin karşısına çıkacak. Yehudi Menuhin'in "Gerçekten dinlediğim en mükemmel kemancı" dediği Vadim Repin'le sezona adım atacak CRR konser sezonunda bu yıl dinleme şansı yakalayacağımız isimlerden bazıları şöyle: Chick Corea&John McLaughlin, James Galway&Lady Galway, Giora Feidman, David Russell, Manolo Sanlucar, Lubna Saleme, Ilya Gringolts, Carmen Lundy, Sa Chen, Sharon Isbin, Konstantin Moskovich, Kremerata Baltica, Nigel Kennedy Quintet, Talich Quartet, Strauss Ensemble, CRR İstanbul Senfoni Orkestrası, Yansımalar, Fazıl Say, Ayla Erduran, Cihat Aşkın, Arto Tunç Boyacıyan, Melihat Gülses, Meral Uğurlu ve Nevzat Sümer. Uluslararası müzik arenasında da isminden övgüyle söz ettiren ülkemizin en prestijli konser salonu Cemal Reşit Rey, ocak ayında başlayacak ve 2009 Mayıs sonuna kadar devam edecek olan sezonun 2. yarısında da; ağırlayacağı 1.500'ün üzerinde sanatçı ve ayda gerçekleştireceği 20'nin üzerinde etkinlikle sanatseverlerle buluşacak. Kültür-Sanat

Figure A.9: Sample near-duplicate news detected by only Tweezer.

News title and description are not important for duplicate detection process. In Figure A.9 two news documents are given that are identified as near-duplicate by only Tweezer. The same content in two documents is given in *italic* font. The difference between two documents is that there is an extra sentence at the beginning of content of first document. Since named entities do not exist in this extra sentence, these two documents

are considered as near-duplicate by Tweezer. Because of this extra sentence I-Match does not detect these documents as near-duplicate.

İstanbul'da yarış kıran kırana

İstanbul'da seçim AK Parti ile CHP arasında son dakikaya kadar sürdü. Oy oranlarının birbirine yakın seyretmesi üzerine AK Parti ve CHP'de gerilim had safhaya yükseldi. CHP adayı Kemal Kılıçdaroğlu, hedeflerinin % 40 olduğunu belirterek, bunun üzerindeki bir sonucun büyük başarı olacağını söyledi.

Seçim sonuçlarının açıklanmasına başlanmasıyla birlikte İstanbul'da CHP ve AK Parti arasında psikolojik bir savaş yaşandı. CHP İstanbul eski İl Başkanı Gürsel Tekin, ellerindeki değerlendirmeye göre AK Parti'yi geçtiklerini açıkladı. Tekin, oy oranlarını % 42 olarak ilan etti. AK Parti cephesi bu açıklamaya karşı açıklama ile cevap verdi. AK Parti İstanbul İl Başkanı Aziz Babuşcu, CHP'nin sonuçları manipüle etmeye çalıştığını savundu. Sandık başında bekleyen müşahitlere yönelik bir psikolojik müdahalede bulunulduğunu savunan İl Başkanı Kılıçdaroğlu ile CHP İl Başkanı Gürsel Tekin'in yaptığı açıklamayı 'ucuz ve basit' olarak niteledi. CHP kanadının, kaybetmiş olmanın verdiği travma ile toplumu manipüle etmeye yönelik, ciddiyetten uzak açıklamalar yaptığını dile getirdi. Buna kayıtsız kalmamak adına basın toplantısı yapmayı tercih ettiğini ifade eden Babuşcu, kendilerine gelen sonuçlar itibarıyla AK Parti'nin oy oranının yüzde 49 olduğunu açıkladı. "Son gülen iyi güler." diyen İl Başkanı, CHP'lilerin bu tür çıkışlarla ancak kısa bir süre kendilerini tatmin edebileceğini kaydetti. Son açıklama yine CHP'den Gürsel Tekin'den geldi. Tekin, Babuşcu'yu suçlayarak, "Bildiği bir şey varsa açıklasın. Biz rakamları söylüyoruz. Ellerinde rakam varsa çıkıp açıklamalılar. Bu, seçimi kaybetme psikolojisidir." dedi. İstanbul Büyüşehir Belediye Başkanlığı dışında metropol ilçelerde de kıran kırana bir yarış yaşandı. Sonuçlar son dakikaya kadar belli olmadı. Saatler dün 22.50'yi gösterdiğinde resmi olmayan rakamlara göre İstanbul'da AK Parti'nin oy oranı yüzde 42,9, CHP'nin yüzde 39,8'di.

Topbaş, ikinci kez kazandı

İstanbul'da AK Parti ile CHP arasında nefes kesen bir yarış vardı. Oy oranlarının birbirine yakın seyretmesi üzerine AK Parti ve CHP'de gerilim had safhaya yükseldi. İlk başlarda başabaş giden yarışı İstanbul'un mevcut başkanı ve AK Parti'nin adayı Kadir Topbaş resmi olmayan rakamlara göre, yüzde 6'ya varan bir farkla CHP'nin önünde bitirdi.

Seçim sonuçlarının açıklanmasına başlanmasıyla birlikte İstanbul'da CHP ve AK Parti arasında psikolojik bir savaş yaşandı. CHP İstanbul eski İl Başkanı Gürsel Tekin, ellerindeki değerlendirmeye göre AK Parti'yi geçtiklerini açıkladı. Tekin, oy oranlarını % 42 olarak ilan etti. AK Parti cephesi bu açıklamaya karşı açıklama ile cevap verdi. AK Parti İstanbul İl Başkanı Aziz Babuşcu, CHP'nin sonuçları manipüle etmeye çalıştığını savundu. Sandık başında bekleyen müşahitlere yönelik bir psikolojik müdahalede bulunulduğunu savunan İl Başkanı Kılıçdaroğlu ile CHP İl Başkanı Gürsel Tekin'in yaptığı açıklamayı 'ucuz ve basit' olarak niteledi. CHP kanadının, kaybetmiş olmanın verdiği travma ile toplumu manipüle etmeye yönelik, ciddiyetten uzak açıklamalar yaptığını dile getirdi. Buna kayıtsız kalmamak adına basın toplantısı yapmayı tercih ettiğini ifade eden Babuşcu, kendilerine gelen sonuçlar itibarıyla AK Parti'nin oy oranının yüzde 49 olduğunu açıkladı. "Son gülen iyi güler." diyen İl Başkanı, CHP'lilerin bu tür çıkışlarla ancak kısa bir süre kendilerini tatmin edebileceğini kaydetti. Son açıklama yine CHP'den Gürsel Tekin'den geldi. Tekin, Babuşcu'yu suçlayarak, "Bildiği bir şey varsa açıklasın. Biz rakamları söylüyoruz. Ellerinde rakam varsa çıkıp açıklamalılar. Bu, seçimi kaybetme psikolojisidir." dedi. İstanbul Büyüşehir Belediye Başkanlığı dışında metropol ilçelerde de kıran kırana bir yarış yaşandı. Sonuçlar son dakikaya kadar belli olmadı. Saatler 01.30'u gösterdiğinde resmi olmayan rakamlara göre İstanbul'da AK Parti'nin oy oranı yüzde 44, CHP'nin yüzde 37,9'du.

Figure A.10: Sample near-duplicate news detected by only I-Match.

In Figure A.10 two news documents are given that are identified as near-duplicate by only I-Match. The same content in two documents is given in *italic* font. These documents are about an election and the only difference between them is the last sentence. In the last sentence uncertain results of election is given, but since the time of news are different, so these numerical values are. These numerical values are eliminated by I-Match in the term selection phase, so these two documents are considered as near-duplicate by I-Match. However there is a named entity, “İstanbul'da AK Parti'nin”, in the last sentence and because of this named entity and the difference of numerical values in two documents, Tweezer generates two different signatures. So, two documents are not identified as near-duplicate by Tweezer.