KNOWLEDGE GRAPH BASED VISUAL INTERPRETATION OF WEB CONTENT

by

Murat Kalender

Submitted to Graduate School of Natural and Applied Sciences

in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Computer Engineering

Yeditepe University

2016

KNOWLEDGE GRAPH BASED VISUAL INTERPRETATION OF WEB CONTENT

APPROVED BY:

Assoc. Prof. Dr. Emin Erkan Korkmaz .... [illegible]
(Thesis Supervisor)

Prof. Dr. Semih Bilgen .... [illegible]

Prof. Dr. Tunga Güngör .... [illegible]

Assoc. Prof. Dr. Nafiz Arıca .... [illegible]

Assist. Prof. Dr. Dionysis Goularas .... [illegible]

DATE OF APPROVAL: ..../..../2016

ACKNOWLEDGEMENTS

I wish to express my gratitude to all the people who have given me encouragement and helped me in the completion of this study. Especially, I would like to thank my thesis supervisor, Emin Erkan Korkmaz, for his guidance, understanding, and support. He gave me a free hand with my research, always had time for discussions, and asked all the right questions that provided valuable focus for this thesis.

I would like to thank the members of my committee, Tunga Güngör and Nafiz Arıca for their time and perceptive comments, which have improved the quality of this thesis.

My special thanks go to my colleagues, Ömer Ozan Sönmez, Çağlar Tırkaz, Mustafa Tolga Eren, Özgün Çırakman, Sezer Kutluk and Günay Gültekin at Huawei whom worked collaboratively with me for development of the Videolization project.

I would like to express my gratitude to the Scientific and Technical Research Council of Turkey (TUBITAK) for supporting my research with TUBITAK 2211-C The Priority Areas Related to Domestic Doctoral Scholarship.

Finally, I would like to thank my parents and family for their support and encouragement.

ABSTRACT

KNOWLEDGE GRAPH BASED VISUAL INTERPRETATION OF WEB CONTENT

Web content nowadays can also be accessed through new generation Internet Connected TVs. However these products failed to change users' behavior for consuming online content. Users still prefer their personal computers instead of their TVs when they access Web content. Certainly, most of the online content is still designed to be presented with a personal computer or mobile devices. In order to overcome usability problem of Web content consumption on TVs, this thesis presents Videolization, a knowledge graph based visual interpretation system that automatically interprets visually given Turkish or English textual Web content by using Semantic Web based technologies. The system visualizes textual Web content by utilizing visual representations of extracted entities from the content. The generated visual interpretation of a given Web content could be automatically converted into a video by using Computer Graphics based technologies. Therefore the main focus of this study is entity linking, which is the most critical task in Content Curation process. Entity linking is the generation of assignments from knowledge graph entities to documents. In contrast to many successful applications for English, there is currently no publicly available entity linking system for Turkish. In order to visualize Turkish content, this thesis presents Thinker, a novel entity linking system for linking Turkish text content with entities defined in the Turkish dictionary or Turkish Wikipedia. The effectiveness of Videolization is validated empirically over opinion surveys and the effectiveness of Thinker is validated empirically over generated data sets. The experimental results show that Thinker greatly outperforms previous methods in terms of disambiguation performance.

ÖZET

ANLAMSAL BİLGİ TABANLI INTERNET İÇERİKLERİNİN GÖRSELLEŞTİRİLMESİ

İnternetin popülerleşmesi ile internet içeriğine yeni nesil televizyonlar üzerinden erişilmektedir. Ancak internet içeriğinin geniş ekran televizyonlar için tasarlanmamış olmalarından dolayı kullanıcılar internet içeriğine erişmek için televizyonlarını tercih etmemektedir. Kullanıcıların büyük bölümü hala televizyon yerine internet içeriğine erişmek için kişisel bilgisayarlarını kullanmaktadır. Bu tez kapsamında, TV'lerde internet içerik tüketimi ve kullanılabilirlik sorunlarını aşmak için Videolization isimli görselleştirme sistemi geliştirilmiştir. Videolization sistemi Türkçe veya İngilizce internet içeriğinin Anlamsal Ağlar teknolojilerini kullanarak otomatik olarak görselleştirilmesini hedeflemektedir. Sistem internet içeriğinden çıkarımı yapılan anlamsal varlıkların görsel ve anlamsal bilgilerini kullanarak görsel sunum yapabilmektedir. Tez çıktısı internet içeriğinin görsel yorumundan, Bilgisayar Grafiği teknolojilerini kullanarak otomatik video üretilebilmektedir. Bu nedenle, bu çalışmanın ana odak noktası Videolization sisteminin arkasındaki varlık bağlama (entity linking) sistemidir. İngilizce için birçok başarılı varlık bağlama uygulamaları bulunmaktadır. Fakat Türkçe dili için kamuya açık kullanılabilir bir varlık bağlama sistemi bulunmamaktadır. Türkçe içerikleri görselleştirmek için bu tez kapsamında Thinker isimli Türkçe varlık bağlama sistemi geliştirilmiştir. Önerilen Thinker sisteminin başarımını ölçmek için deneyler yapılmıştır. Deneylerde Thinker sistemi belirsizlik giderme performansı açısından daha önceki yöntemlere göre çok daha iyi performans göstermiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

AI	Artificial Intelligence
BBS	Bulletin Board System
BOW	Bag-Of-Words
CBOW	Continuous Bag-of-Words
CRF	Conditional Random Field
DAWG	Direct Acyclic Word Graph
DES	Description Embedding Similarity
DSSM	Deep Structured Semantic Modeling
DT−RNN	Dependency Tree Recursive Neural Network
DW2VS	Description Word2Vec Similarity
GLOVE	Global Vectors for Word Representation
HCI	Human Computer Interaction
LCS	Letter Case Similarity
LESK	Lesk
LS	Link Similarity
LW2VS	Link Word2Vec Similarity
MES	Metadata Embedding Similarity
NER	Named Entity Recognition
NGD	Normalized Google Similarity Distance
NLP	Natural Language Processing
NSGA−II	Non-dominated Sorting Genetic Algorithm II
NSS	Name String Similarity
OWL	Web Ontology Language
POJO	Plain Old Java Object
RDF	Resource Description Framework
SAAS	Software as a Service
SDW2VS	Simple Description Word2Vec Similarity
SLESK	Simplified Lesk
SS	Suffix Similarity
SVM	Support Vector Machine

TC	Type Classifier Similarity
TCS	Type Content Similarity
TDK	Türk Dil Kurumu
TLA	Turkish Language Association
TS	Type Similarity
T−SNE	T-distributed Stochastic Neighbor Embedding
TTS	Text to Speech
UI	User Interface
VVDL	Videolization Video Description Language
WSD	Word Sense Disambiguation

1. INTRODUCTION

Web content nowadays can also be accessed through the new generation Internet Connected TVs. These TV sets include a web browser and a virtual keyboard so that users can browse and search online content using their TV sets. Consuming the Web on giant screens is a promising utility of these TV sets.

Although average American consumer spends five hours in front of the TV every day[1], it has been found out that only ten percent of the Internet Connected TV owners have ever used their built-in browsers[2]. We believe there are two reasons behind this finding; (i) built-in browsers are inferior to their Desktop or mobile counterparts, (ii) interaction methods are not intuitive as users are accustomed to. In relation to these issues, the users prefer their PCs or mobile devices for Web consumption. Furthermore, we argue that this is inherently related to the design philosophy of the Web content.

Many of the most popular websites on the Web such as Facebook and Reddit feature infinite scrolling style browsing. Additionally more static websites can come in varying sizes. A Web content may be very tall or wide. In turn, it leads to an undesirable experience to use a TV remote as an input device, causing the user to repeatedly scroll down or across a Web page in order to read it. This study addresses problem of how to effectively and intuitively consume Web content on Internet Connected TVs. We can specify this problem further in selected domains as well; how to turn news websites into news shows, how to turn e-commerce sites into shopping TV broadcasts and how to turn Bulletin Board System (BBS) into a talk show or debate style broadcast. We believe that with a change in design philosophy, it is possible to deconstruct the Web content and then it can be reconstructed in a TV friendly format. This would allow the users to "surf the Web" in completely different yet satisfying way.

In order to investigate this problem, it is imperative to comprehend the Web content first. The Web content can be textual, visual or audial. These contents are encountered as a part of the user experience on the Web. It is possible to have text, images, sounds, videos and animations on a Web page, however a great portion of the Web content is predominantly

[1] http://www.nydailynews.com/life-style/average-american-watches-5-hours-tv-day-article-1.1711954
[2] https://www.npdgroupblog.com/internet-connected-tvs-are-used-to-watch-tv-and-thats-aboutall/

composed of text. Thus a workflow to convert textual information into multimedia format is needed. Motivated by sayings like, "A picture is worth a thousand words", this thesis presents the Videolization system. Videolization is a knowledge graph based visual interpretation system that automatically interprets visually given Turkish or English textual Web content by using Semantic Web based technologies. Figure 1.1 shows the system architecture of Videolization system. It has three major modules: Content Curation, Repository and Videolization Channels.

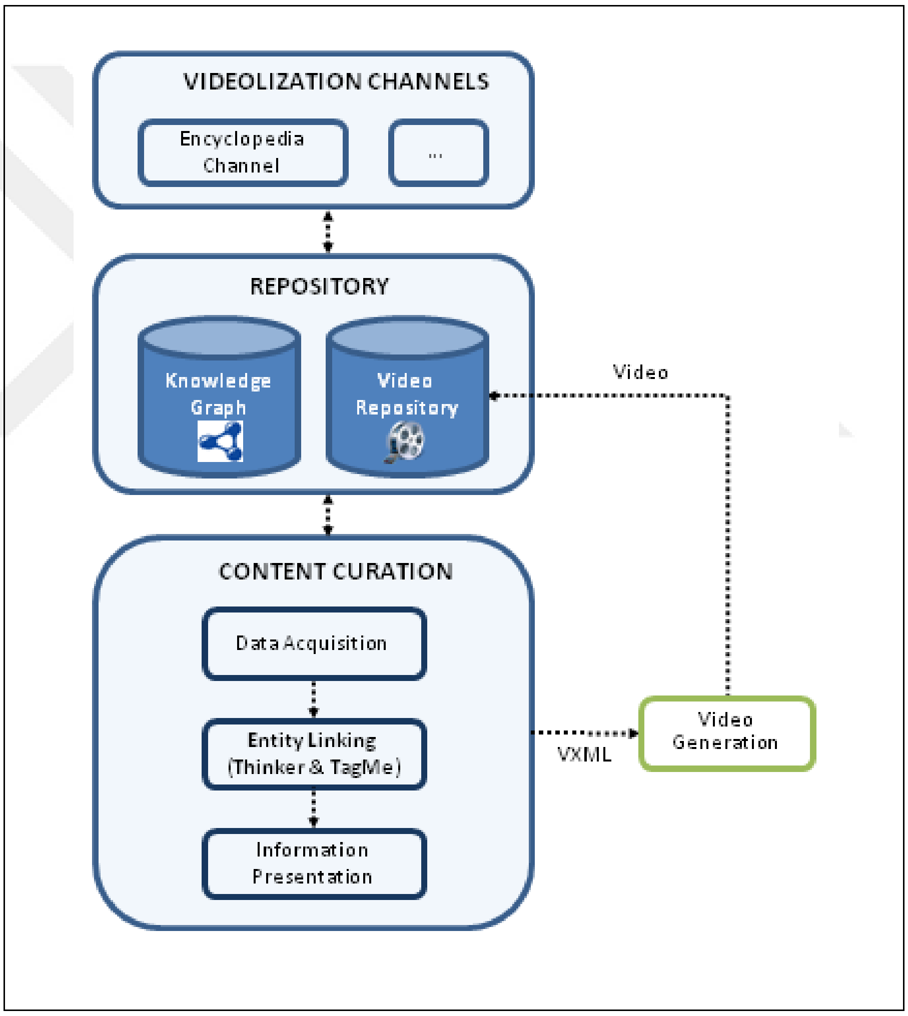


Figure 1.1. Architecture of Videolization system.

The Content Curation module of Videolization system visualizes text content by utilizing visual representations of extracted entities. One of the main steps of this workflow is entity linking which is the generation of assignments from knowledge graph entities to documents. In contrast to many successful applications for English, there is currently no publicly available entity linking system for Turkish. In order to visualize Turkish content, this thesis presents Thinker, a novel entity linking system for linking Turkish text content with entities defined in the Turkish dictionary (tdk.gov.tr) or Turkish Wikipedia (tr.wikipedia.org). Key challenges in entity linking systems such as knowledge base coverage, word sense disambiguation and weighting significant entities within a context are addressed in this study.

Entity linking process has to handle the ambiguity of the natural language since an entity mention in a text content might have more than one corresponding entity defined in the utilized knowledge base. For example, Figure 1.2 shows an example mapping of a piece of textual content to entities defined in Turkish Wikipedia (Vikipedi). A spotter (entity detector) would detect the two entity mentions: "Arsenal" and "pas" in this sentence. Once the entity mentions are detected, the main challenge is to cope with the ambiguous natural language mentions. The Turkish entity mention "pas" refers to more than one entity, which are passing in football and rust. In fact, it is quite easy to map the entity mention "pas" to the correct entity "Pass (football)" in this case, since the other entity "Arsenal FC" has one referring entity and it has a similar context with the entity "Pass (football)".

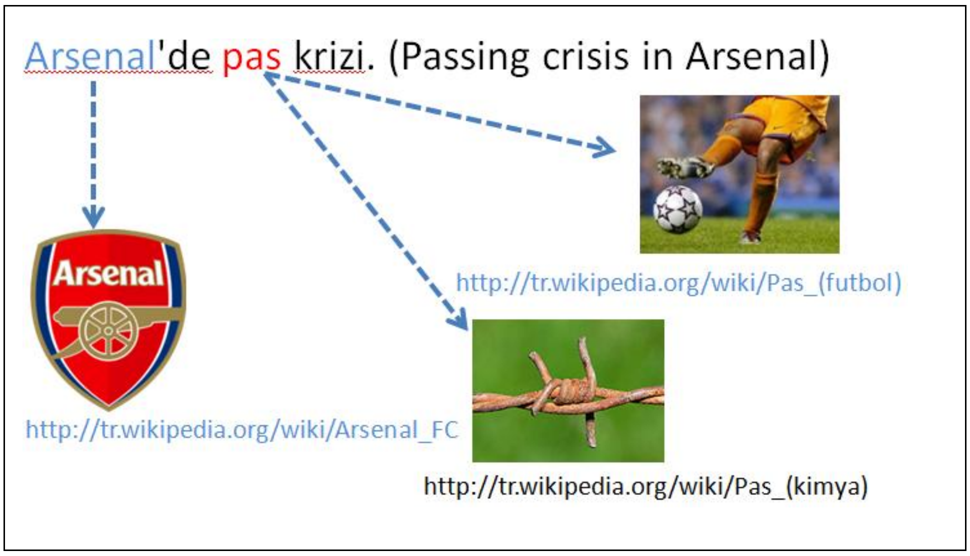

Arsenal'de pas krizi. (Passing crisis in Arsenal)
Arsenal
http://tr.wikipedia.org/wiki/Pas_(futbol)
http://tr.wikipedia.org/wiki/Arsenal_FC
http://tr.wikipedia.org/wiki/Pas_(kimya)

Figure 1.2. An example linking of a piece of textual content to entities defined in Vikipedi.

The success of an entity linking system clearly depends on the term coverage of the knowledge base utilized. In order to effectively process domain-independent documents a comprehensive, up-to-date, and evolving knowledge base is required. Wikipedia is one of the popular knowledge bases that satisfies these requirements and therefore it is widely used in entity linking systems. However, the long tail of entities is not popular enough to have their own Wikipedia articles. For example, "Yetenek Sizsiniz Türkiye" (Turkish version of the Got Talent series) television show exists in Vikipedi and it is typed as television show, however "İbo Show", another famous Turkish television show, is not defined as a Vikipedi article. In this study, an entity discovery system is also proposed that semi-automatically detects entity mentions that are not defined in Vikipedi, in a given Turkish text. The system also discovers the semantic typing of detected unlinkable entity mentions. For informative knowledge, we aim to type new entities in a fine-grained manner (e.g., basketball player, economist, airport, as opposed to generic types like person, organization, event).

After the entity linking step, the visualized elements are displayed and animated by using an external video visual effects and compositing application. The audio component of the video can be produced with a Text to Speech (TTS) system. Combining this audio with visual representations of most significant entities in the given sentence produces a video segment. Stitching segments into each other with appropriate transitions yields the final video. Figure 1.3 shows example of a generated video by analyzing Wikipedia article of the USA.

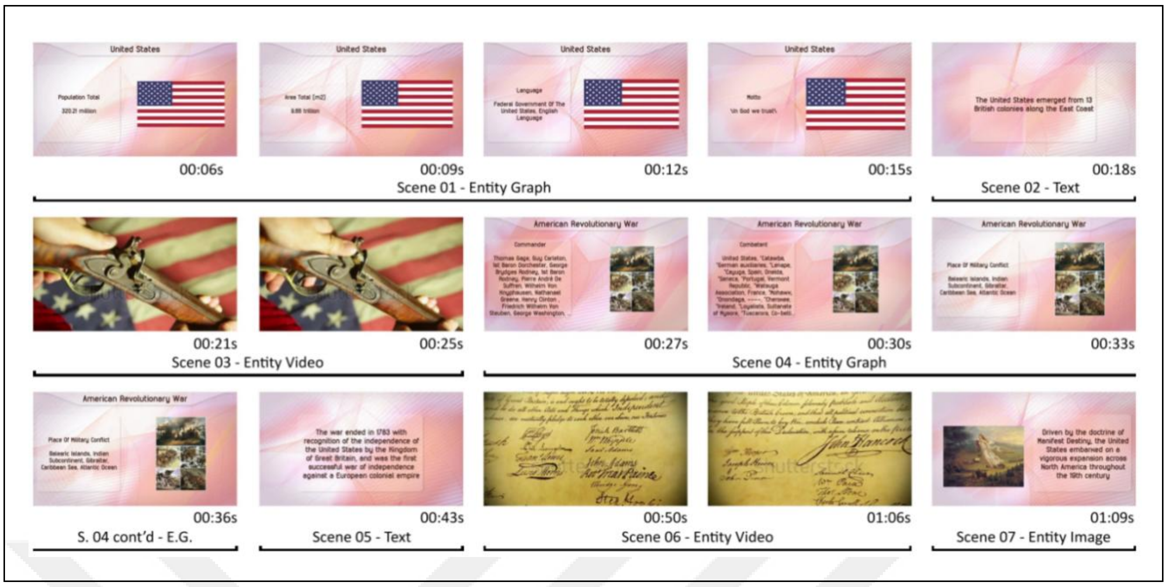


Figure 1.3. Thumbnails from a video generated by analyzing Wikipedia article of the USA are shown. The proposed system videolizes each sentence as a scene with one of the following types: Entity Graph (Scenes 1 and 4), Entity Video (Scenes 3 and 6), Entity Image (Scene 7) and Text (Scene 5).

Through our experiments and evaluations, we show that TV friendly videos could be generated by semantic analysis of Web content in order to understand its context and decide how to visualize in a video. We detail and demonstrate our system through a use case based on Wikipedia[3] articles. Furthermore, we evaluate the user friendliness and effectiveness of the system with a qualitative user study. The survey results show that majority of the users prefer using Videolization to consume Web content on their TVs.

Additionally, the effectiveness of Thinker is validated empirically over generated data sets. The experimental results show that Thinker greatly outperforms previous methods in terms of disambiguation performance.

The main contributions of this PhD study are summarized as follows:

- A novel visualization method to convert Web content into videos. Videolization Video Description Language (VVDL) is proposed to describe videos in XML format and provide interdependence between video content and its visualization.

[3] http://en.wikipedia.org/

- A novel Turkish entity linking system is proposed for linking Turkish text content with entities defined in a generated Turkish knowledge base by integrating Turkish Wikipedia and the Turkish dictionary.
- A novel Turkish entity discovery system is proposed that discovers semantic typing of entities that are not defined in Turkish Wikipedia in order to overcome the knowledge base coverage problem.

1.1. OUTLINE

The outline of this thesis is organized as follows. Chapter 2 gives background information and related work about entity linking, entity discovery and text visualization. Chapter 3 and Chapter 4 present detailed models of the Videolization and Thinker systems, respectively. Experimental results and evaluation of the proposed approaches are presented in Chapter 5 and we conclude in Chapter 6.

2. BACKGROUND AND RELATED WORK

This chapter provides the background information regarding the thesis work. In order to achieve the main purpose of this thesis, visual interpretation of Web content, Natural Language Processing (NLP) and Deep Learning methods, and Semantic Web technologies are utilized. Therefore, we first give a brief overview of these research areas. Then, we introduce and review the related work on four specific tasks that we deal with in this study: entity linking, fine-grained entity recognition, text visualization and video generation approaches and solutions.

NLP is required to process input text content in human language format. In this study, in order to process Internet content we have used commonly used NLP functions such as sentence detection, morphological analysis, morphological disambiguation, etc.. The main focus of this thesis is visual interpretation of Turkish text content. Therefore, the structure of Turkish language and Turkish NLP studies are also covered in this chapter.

Deep learning has recently shown much promise for NLP with the state-of-the-art results obtained in NLP applications. In this study, we also be nefited from deep learning approaches in order to realize the proposed entity linking system.

We also give a brief overview of Semantic Web technologies in this chapter. We utilized semantic databases to visualize given input text content. Metadata about identified entities in the input text are shown visually in generated videos in order to create more informative videos.

The proposed video generation system visualizes text content by utilizing visual representations of extracted entities. Therefore, we introduce the underlying methods, systems and algorithms on entity linking and fine-grained entity recognition. We also present alternative visualization approaches in sections 2.6 and 2.7.

2.1. NATURAL LANGUAGE PROCESSING

The field of Natural Language Processing aims to convert human language into a formal representation that can be processed by computers easily. Current end applications include many information retrieval and extraction tasks such as semantic search, question answering and summarization systems.

While complete natural language understanding is still a distant goal, researchers have applied a divide and conquer approach and identified several sub-tasks useful for application development and analysis. The tasks range from the syntactic, such as sentence detection, tokenization and part-of-speech tagging, to the semantic, such as word sense disambiguation, semantic-role labeling and named entity recognition [13]. Functionalities of commonly used NLP functions are explained below in detail:

- *Sentence detection* function splits the given content into sentences. Sentence detection is harder than it may appear. While sentences end with symbols like the period and the question mark, these symbols do not necessarily terminate sentences. The presence of abbreviations and numbers that include such characters complicates the sentence detection process. For example, consider the following sentence: Youtube.com is a video sharing website. The website name includes a period which does not end the sentence.
- *Tokenization* function splits a sentence into tokens (words). Tokenizing cannot be simply handled by detection of the space character. A tokenizer is required to split words that consist of contractions (e.g. doesn't).
- *Part-of-speech tagging* function labels the tokens with the parts of speech such as noun, verb, adverb, etc. Consecutive noun tokens, which form noun phrases, are extracted with their occurrence frequency. For example, part-of-speech tagging of "Istanbul is located in Turkey" sentence is "Istanbul/NNP is/VBZ located/VBN in/IN Turkey/NNP". The words are tagged with the parts of speech: *NNP* (Proper noun, singular), *VBZ* (Verb, 3rd person singular present) and *VBN* (Verb, past participle). There are totally 36 different part-of-speech tags.[4]
- *Morphological analysis* function identifies morphemes and other linguistic units, such as root words, affixes, part-of-speech of the given input word. Table 2.1 shows morphological analysis of input English words.

Table 2.1. Morphological analysis of sample English words.

[4] https://cs.nyu.edu/grishman/jet/guide/PennPOS.html

Input	**Morphological Analysis**
Dogs	Dog + Noun + Plural
Dog	Dog + Noun + Singular
Eating	Eat + Verb + Present
Caught	Catch + Verb + Past

- *Morphological disambiguation* function selects the correct morphological parse for a given input word in a given context. There might be several different morphological parses of a word. Especially in Turkish, almost half the words in text are morphologically ambiguous [14]. For example, there are four different analyses for the Turkish word "kalemi". Depending on the context, it might mean one of the following meanings: “my Castle”, “my castle”, “his/her pencil” or “the pencil”.
- *Word Sense Disambiguation (WSD)* is the process of automatically mapping a polysemous word (a word having many meanings) to an appropriate sense (meaning) according to the context in which it is used. In most cases, it is easy for humans to disambiguate words. However, WSD is a complex problem that is difficult to solve automatically. For example, the word “java” has different meanings. The word “java” may refer to an island, a programming language or to a coffee brand.
- *Named-entity recognition (NER)* is the task of identifying elements in text and classifying them into pre-defined categories such as the names of persons, companies, locations, expressions of times, quantities, monetary values, percentages, etc. For example, named-entity recognition of "Istanbul is located in Turkey" sentence is "<location>Istanbul</location> is located in <location>Turkey</location>". As a result of this operation, the words Istanbul and Turkey are identified as location entities.

2.1.1. Turkish NLP

Turkish is the official language of Turkey and it is the most widely spoken among the Turkic languages, with around 75 million native speakers. The distinctive characteristics of Turkish are vowel harmony, free word order and extensive agglutination [15].

- Vowel harmony is the principle by which a native Turkish word comprises either exclusively front vowels (e, i, ö, and ü) or exclusively back vowels (a, ı, o, and u).
- Turkish language has free word order. The most common word order is Subject-Object-Verb for Turkish language. However, other possible word orders are also used frequently in Turkish.
- Turkish is an agglutinative language and Turkish words are generated by addition of derivational and inflectional affixes to roots or stems. For example, the following Turkish word "uygarlaştıramayabileceklerimizdenmişsinizcesine", which means "(behaving) as if you were one of those whom we might not be able to civilize" in English, is produced from the root "uygar (civilized)" and gets 11 affixes [16].

In contrast to many successful NLP applications and studies for English, there are currently limited NLP applications and studies for Turkish, because its free word order and extensive agglutination structure make processing of Turkish complex. Currently, NLP studies for Turkish are more focused on morphological analysis [17–19], morphological disambiguation [14, 20, 21], word sense disambiguation [22–24] and named entity recognition [25,26]. The Turkish NLP studies on semantic tasks are very limited. There is currently no publicly available fine-grained entity recognition, entity linking or semantic-role labeling system for Turkish.

There are currently two publicly available popular NLP systems for Turkish, which are Zemberek [1] and ITU Turkish NLP Web Service [2]. Zemberek is one of the popular open source NLP libraries for Turkish. Zemberek provides the most common NLP tasks, such as sentence detection, tokenization, morphological analysis and morphological disambiguation. Zemberek uses rule based algorithms in order to find the possible root and suffixes of a given word. Simply it handles morphological analysis in four steps [1]:

(i) Firstly, Zemberek preprocesses the input word. The pre-processing operation includes removing accents, hyphens etc. and converting it to lowercase. If the preprocessed word contains non Turkish characters, the morphological analysis operation terminates.

(ii) Secondly, Zemberek finds root candidates for the preprocessed word by using an appropriate root selector. There are three different root selectors in Zemberek. The first one is used for normal strict root selection for a given word. For example, for the Turkish word "elmaslar" the system determines three possibilities; el:noun, elma:noun and elmas:noun. Second selector uses a string similarity algorithm in order to apply a tolerance level when selecting candidate roots. As a result, it generates more results compared to the strict selector. The third one has a tolerance for letters which do not exist in the ASCII encoding. These root selectors use a special Direct Acyclic Word Graph (DAWG) tree as root dictionary in order to find candidate root words. The dictionary contains approximately 30.000 root words for Turkish language. Figure 2.1 shows a simplified structure of such a tree. For example, the Turkish word "balerin" is connected with the Turkish root words "bal" and "bale".

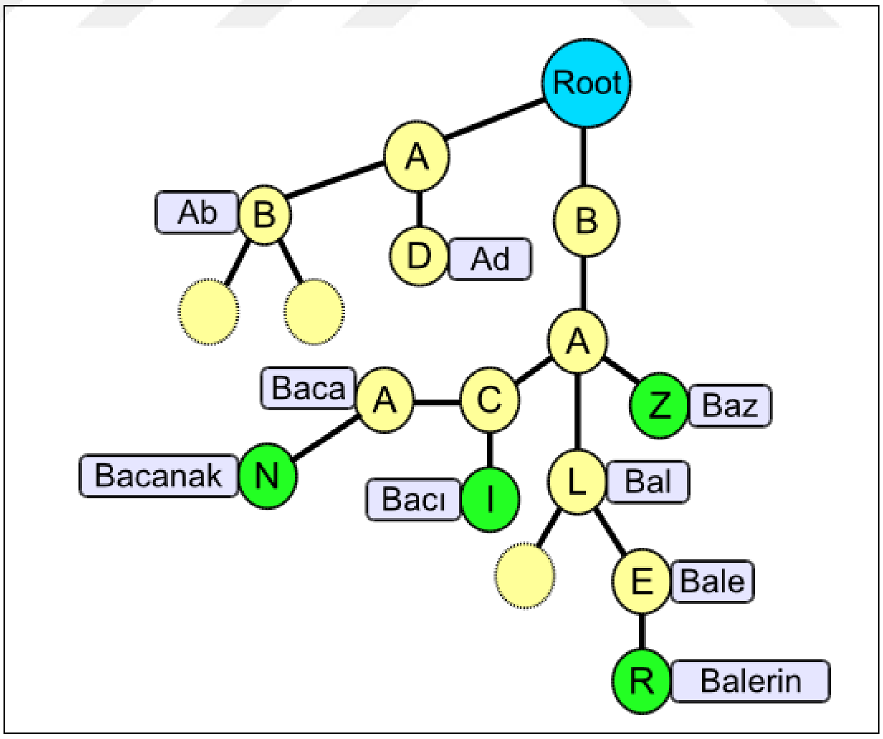


Figure 2.1. A simplified version of Zemberek root dictionary tree [1].

(iii) Thirdly, Zemberek adds possible suffixes to the candidate root words until either the input word is constructed, or there are no suffix alternatives left.

(iv) Finally, Zemberek post processes the parser results in order to check whether the symbols, or uppercase letters are used correctly in the input word. This step is necessary for parsing abbreviations and words containing special letters such as "Ahmet'in" or "prof.e".

Zemberek morphological disambiguation function uses an exact implementation of the model proposed by Hakkani-Tür et al. [21]. The authors propose a probabilistic model that scores the probability of each distinct morphological parse of a word by considering statistics over the individual inflectional groups and surface roots in 3-gram models.

ITU Turkish NLP Web Service is another publicly available Turkish NLP library which operates as a SaaS (Software as a Service) and provides the state of the art NLP tools in many layers: pre-processing, morphology, syntax and entity recognition. Figure 2.2 shows functionalities and flow of the ITU Turkish NLP pipeline.

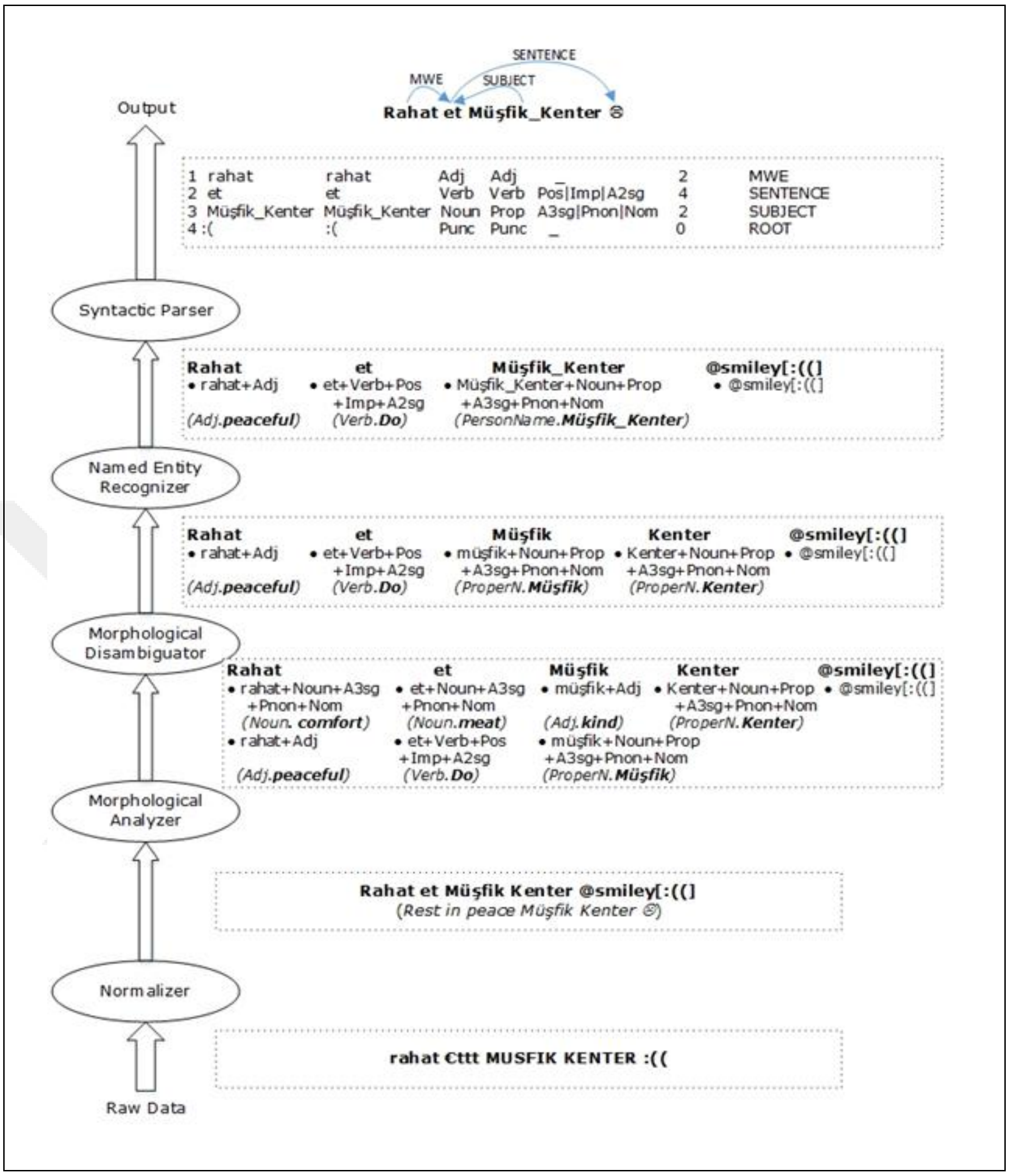

SENTENCE
MWE
SUBJECT
Output
Rahat et Müşfik_Kenter ☹
1 rahat rahat Adj Adj _ 2 MWE
2 et et Verb Verb Pos|Imp|A2sg 4 SENTENCE
3 Müşfik_Kenter Müşfik_Kenter Noun Prop A3sg|Pnon|Nom 2 SUBJECT
4 :(:(Punc Punc _ 0 ROOT
Syntactic Parser
Rahat • rahat+Adj (Adj.peaceful)
et • et+Verb+Pos +Imp+A2sg (Verb.Do)
Müşfik_Kenter • Müşfik_Kenter+Noun+Prop +A3sg+Pnon+Nom (PersonName.Müşfik_Kenter)
@smiley[:(() • @smiley[:(()
Named Entity Recognizer
Rahat • rahat+Adj (Adj.peaceful)
et • et+Verb+Pos +Imp+A2sg (Verb.Do)
Müşfik • müşfik+Noun+Prop +A3sg+Pnon+Nom (ProperN.Müşfik)
Kenter • Kenter+Noun+Prop +A3sg+Pnon+Nom (ProperN.Kenter)
@smiley[:(() • @smiley[:(()
Morphological Disambiguator
Rahat • rahat+Noun+A3sg +Pnon+Nom (Noun. comfort) • rahat+Adj (Adj.peaceful)
et • et+Noun+A3sg +Pnon+Nom (Noun.meat) • et+Verb+Pos +Imp+A2sg (Verb.Do)
Müşfik • müşfik+Adj (Adj.kind) • müşfik+Noun+Prop +A3sg+Pnon+Nom (ProperN.Müşfik)
Kenter • Kenter+Noun+Prop +A3sg+Pnon+Nom (ProperN.Kenter)
@smiley[:(() • @smiley[:(()
Morphological Analyzer
Rahat et Müşfik Kenter @smiley[:(()
(Rest in peace Müşfik Kenter ☹)
Normalizer
rahat Cttt MUSFIK KENTER :((
Raw Data

Figure 2.2. ITU Turkish NLP pipeline [2].

In this study, we have preferred to use the Zemberek library mainly due to performance considerations. Making a web service request is much slower compared to native library usage. The sentence detection, morphological parsing and disambiguation processes are carried out by the utilization of Zemberek library functions.

2.2. DEEP LEARNING

The success of many current machine learning methods is based on human-designed representations and input features. When machine learning is applied to the attentively selected input features, it becomes merely about optimizing weights to make the best final prediction. Deep learning can be seen as putting back together representation learning with machine learning. It attempts to jointly learn good features, across multiple levels of increasing complexity and abstraction and the final prediction [27].

Deep learning is part of a broader family of machine learning methods based on learning representations of data. If machine learning could learn representations (features) automatically, the entire learning process could be automated more easily and many more tasks could be solved. Deep learning provides one way of automated feature learning. For example, an observation of an image could be represented in different ways such as a vector of pixels. However some representations make it easier to learn tasks of interest like the image of a human face from example images. Research in this area tries to define what makes better representations and how to create models in order to learn these representations.

Deep learning algorithms are founded on distributed representations. The underlying assumption behind distributed representations is that observed data is created by the interactions of many different features on different levels. Deep learning adds the assumption that these features are organized into multiple levels, corresponding to different levels of abstraction or composition. Varying numbers of layers and layer sizes could be used to provide different amounts of abstraction [28]. As shown in Figure 2.3, first layer of a deep neural network learns simple edge filters, the second layer captures primitive shapes and higher levels combine these to form objects.

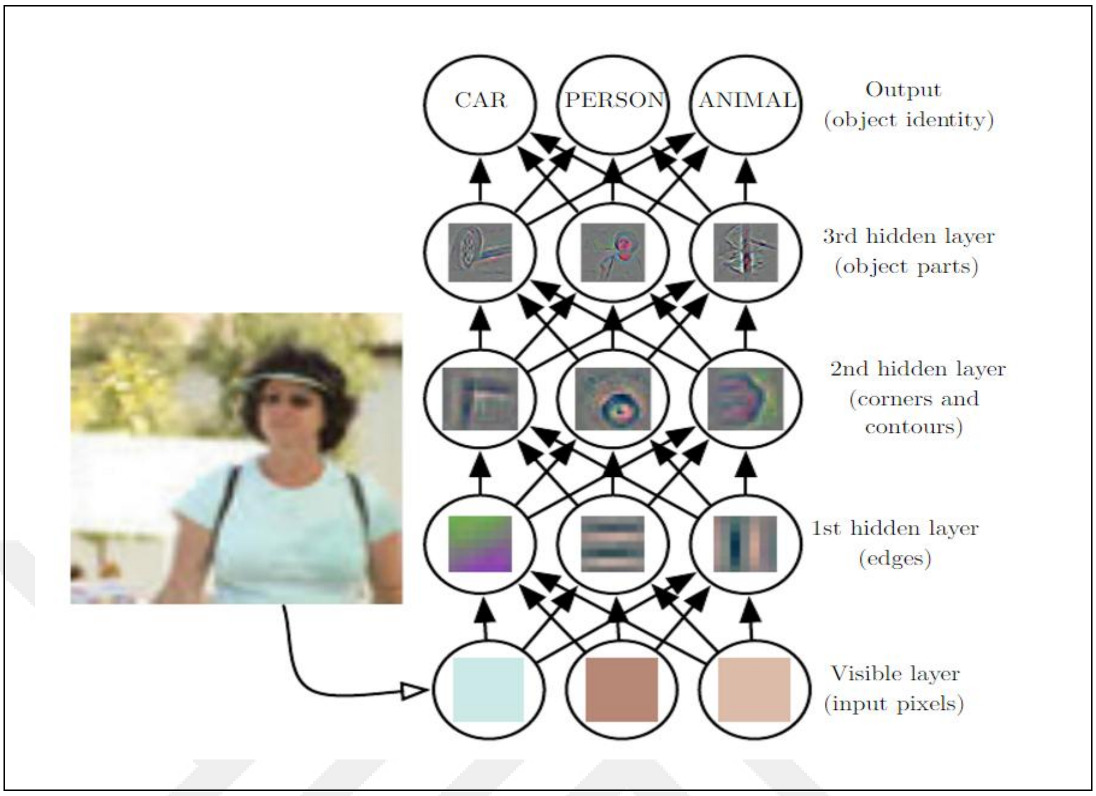


Figure 2.3. Each successive layer in a neural network uses features in the previous layer to learn more complex features [3].

Neural networks have been around for many decades [29]. However, until 2006, deep and fully connected neural networks were commonly outperformed by shallow architectures that used feature engineering. In that year, Hinton and Salakhutdinov [30] introduced a novel way to pre-train deep neural networks. The idea was to use restricted Boltzmann machines to initialize weights of one layer at a time. This greedy procedure can initialize the weights of the full neural network step by step in basins of attraction and it becomes possible to obtain better local optima [31]. Later, Vincent et al. [32] showed that similar effects can be obtained by using autoencoders.

Autoencoders are simple learning circuits which aim to transform inputs into outputs with the least possible amount of distortion in order to learn a compressed, distributed representation (encoding) for a set of data. Typically, the purpose is dimensionality reduction. As a concrete example, suppose the inputs are the pixel intensity values from a 20×20 image (400 pixels). So in layer $L1$, there would be 400 input units. If the unit size of the hidden layer $L2$ is set to 100, the network would be forced to learn a "compressed" representation of the input (encoding). Given only the 100 length vector of hidden unit activations, the network tries to reconstruct the $400 - pixel$ input (decoding). When there is structure in the data, for example, if some of the input features are correlated, then this algorithm will be able to discover some of those correlations. In fact, this simple autoencoder often ends up learning a low-dimensional representation very similar to PCAs [4]. Figure 2.4 shows example of an autoencoder architecture where x_i represents the input given to the network and $\hat{x}_1$ is the reconstructed input.

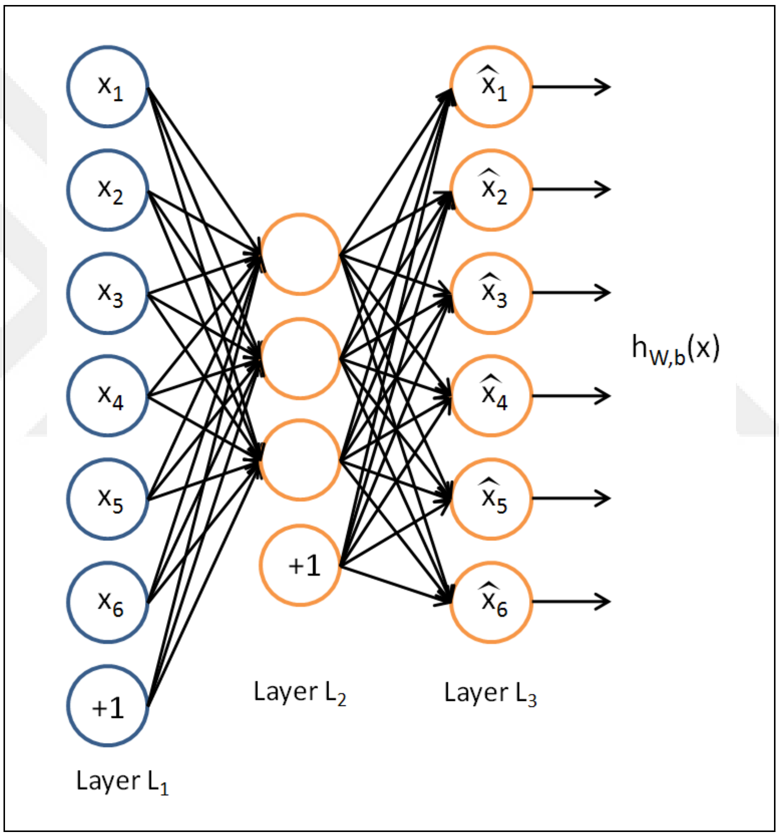


Figure 2.4. An Autoencoder architecture [4].

With advances in large datasets, faster, parallel computers and deep learning architectures such as deep neural networks, convolutional deep neural networks and deep belief networks, it has been possible to apply deep learning to fields like computer vision, automatic speech recognition, natural language processing (NLP), and music/audio signal recognition where they have been shown to produce state-of-the-art results on various tasks [27].

2.2.1. Deep Learning for NLP

Deep learning has recently shown much promise for NLP with the state-of-the-art results obtained in applications such as speech recognition [33], part-of-speech tagging [13], named entity recognition [13] and neural network based language models [34,35].

Neural network based language models use distributed vector representations for words, namely word embeddings, rather than discrete word counts. Word embedding is a distributed representation of a word with a high dimensional vector, where each dimension corresponds to a latent feature of the word [36]. Thus a word embedding can capture both semantic and syntactic information of the word. The statistics of word occurrences in a corpus constitute the primary source of information available to all unsupervised methods for learning word embeddings [12]. Resulting distributed representation of words have several advantages compared to traditional language models such as bag-of-words (BOW) in terms of compactness and sparsity. Also semantically similar words are represented with closer vectors. For example, if we consider two semantically similar words "capital" and "city", it is expected to have similar values at least in some dimensions of their corresponding vectors. There are various methods to generate word embeddings and this study established word embeddings by using GloVe [12] and Word2Vec [5].

GloVe (Global Vectors for Word Representation) is a tool[5] recently released by Stanford NLP Group researchers Jeffrey Pennington, Richard Socher, and Chris Manning for learning continuous-space vector representations of words. It is an unsupervised learning algorithm that is carried out in global word-word co-occurrence statistics counted from the corpus. The GloVe model is trained on the non-zero entries of a global word-word co-occurrence matrix X, where a cell Xij is a "strength" which represents how often the word i appears in the context of the word j. The matrix is populated by a single pass through the entire corpus to collect the statistics. For large corpora, this operation can be computationally expensive, but it is a one-time up-front cost. Following training iterations are much faster because the number of non-zero matrix entries is generally much smaller than the total number of words in the corpus [12]. Table 2.2 shows the calculated co-occurrence probabilities for target words ice and steam from a six billion word corpus.

[5] http://nlp.stanford.edu/projects/glove/

Table 2.2. Co-occurrence probabilities constructed using Glove model for target words ice and steam with selected context words from a six billion token corpus [12].

Probability and Ratio	**k = solid**	**k = gas**	**k = water**	**k = fashion**
P(k|ice)	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
P(k|steam)	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
P(k|ice) / P(k|steam)	8.9	8.5×10^{-2}	1.36	0.96

Once co-occurrence matrix X is constructed, the next step is to decide vector values in continuous space for each word in the corpus such that their dot product equals the logarithm of the words' probability of co-occurrence. The authors [12] derive the following function J to train the model using a gradient descent algorithm:

$$\mathrm{J} = \sum_{i,j=0}^{V} f(X_{ij})(\mathrm{v}_{\mathrm{wi}} \cdot u_{wj} + bi + bj - \log X_{ij})^2 \tag{2.1}$$

where V is the word vocabulary, X_{ij} is the number of times the word w_j appears in the context of word w_i, b_i and b_j are bias terms, f is a weighting function that cuts low co-occurrences, which are usually noisy, and also avoiding to overweight high co-occurrences.

Word2Vec [5] is a linguistic model based on a neural network and the model provides an alternative approach to produce word embeddings. Word2Vec learns a low dimensional continuous vector for each word from their distributional properties observed in a given corpus. Word2Vec uses a simple log-linear classification network, and provides skip-gram and continuous bag-of-words (CBOW) architectures (see Figure 2.5).

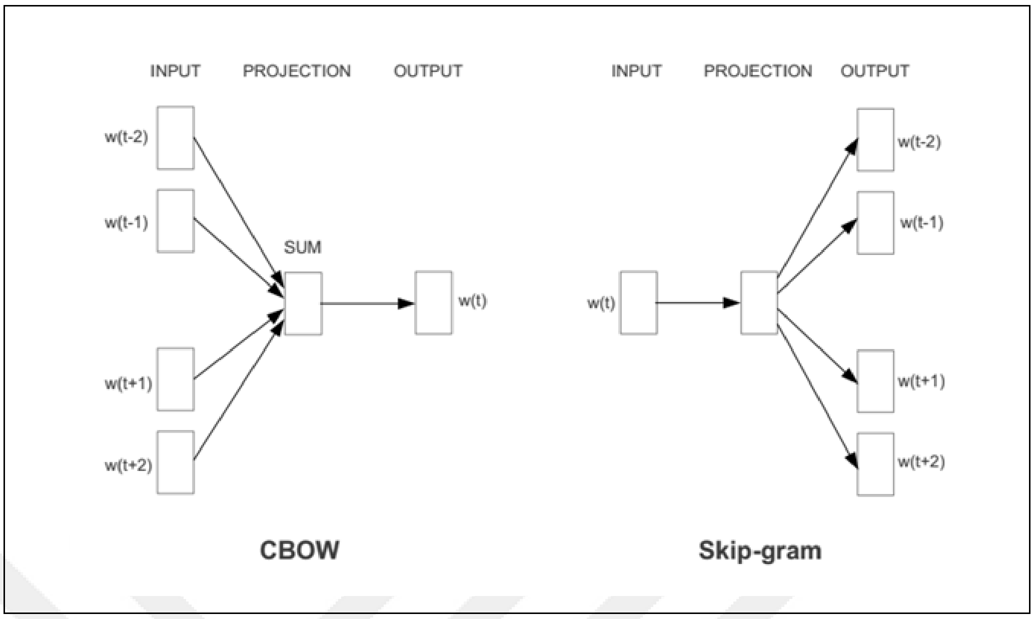


Figure 2.5. The CBOW and Skip-gram architectures [5].

The CBOW network predicts each word based on neighbouring words. The input layer of CBOW is projected in the hidden layer in a linear form and it is represented as a bag of words. Input words get projected into the same position, that is, vectors are averaged. By this way, order of words in the history does not influence the projection. CBOW can be learned using extremely massive data that cannot be processed in another neural network bag-of-words model [34].

Skip-gram is a practical word representation in predicting neighbouring words in a document or a sentence. It predicts the neighbouring words or context when a single word is provided. Skip-gram learns the averaged co-occurrence of two words in a training set. Contrary most of the previously used neural network architectures for learning word vectors, training of the skip-gram model does not involve dense matrix multiplications. When given the sequence of words w_1, w_2, w_3..., w_n, the purpose of skip-gram is to maximize the average log probability with the formula below [37]:

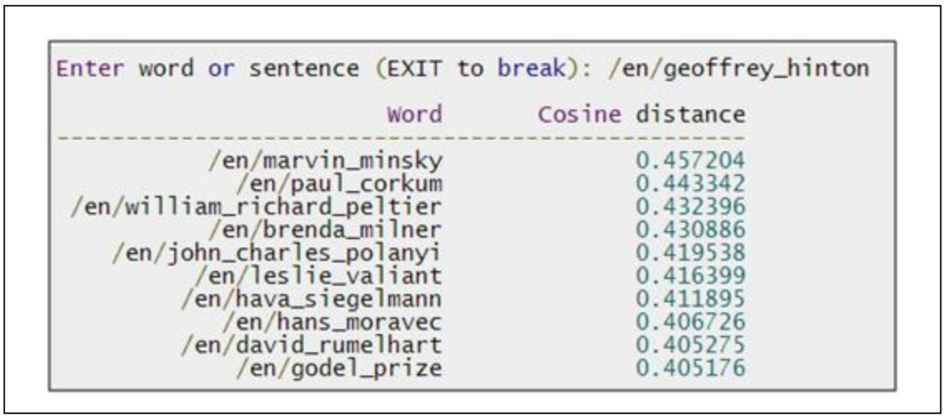
$$\frac{1}{\mathrm{N}}\sum_{\mathrm{n}=1}^{\mathrm{N}}\sum_{-\mathrm{c}<j<c,j\neq 0}\log\mathrm{p}(\mathrm{w}_{\mathrm{n}+\mathrm{j}}|\mathrm{w}_{\mathrm{n}}) \tag{2.2}$$

where c is the size of the training context (which can be a function of the center word w_n). Higher c results in more training examples and thus can result to a better accuracy, at the expense of the training time. The basic Skip-gram formulation defines $p(w_{n+j}|w_n)$ using the softmax function [37]:

$$\mathrm{p}(\mathrm{w}_0|\mathrm{w}_1) = \frac{\exp\left(u_{w0}^T\, v_{w1}\right)}{\sum_{k=1}^{V} \exp\left(u_k^T v_{w1}\right)} \tag{2.3}$$

where the input and output vector representations of the w are u_w and v_w and V is the size of the vocabulary.

The authors [37] report that the Skip-gram model can be trained in a day from a corpus of containing more than 100 billion words using a single machine. Implementation of the Skip-gram model is publicly available.[6] Also Google News and Freebase word vectors are provided in this website. The Google News vectors are 300 dimensional vectors covering three million words and phrases and the Freebase vectors are 1000 dimensional vectors covering 1.4M entities. For example, Figure 2.6 shows the most similar ten entities with their cosine distances to the input entity "/en/geoffrey hinton" within the provided Freebase entity vectors.



```
Enter word or sentence (EXIT to break): /en/geoffrey_hinton

                        Word       Cosine distance
------------------------------------------------------
          /en/marvin_minsky              0.457204
            /en/paul_corkum              0.443342
 /en/william_richard_peltier              0.432396
          /en/brenda_milner              0.430886
    /en/john_charles_polanyi              0.419538
          /en/leslie_valiant              0.416399
        /en/hava_siegelmann              0.411895
           /en/hans_moravec              0.406726
        /en/david_rumelhart              0.405275
            /en/godel_prize              0.405176
```

Figure 2.6. The Freebase word vector entity distance calculation example.

[6] http://code.google.com/p/word2vec/

Recently, Le and Mikolov [38] have adapted the Skip-gram model for pieces of texts and introduced the Paragraph Vectors. This model maps sentences and documents to high dimensional vectors instead of single words. Dai et al. [39] used this model to learn vector representations of Wikipedia articles. Figure 2.7 shows the visualization of vector representations of Wikipedia articles using t-distributed stochastic neighbor embedding (t-SNE) [40], which is a machine learning algorithm for dimensionality reduction. The articles categorized with different categories (people, animals, plants, films, actors and directors) are highlighted with various colors. We can observe that articles having common categories are represented with similar vector representations. This figure is generated using a publicly accessible web application.[7]

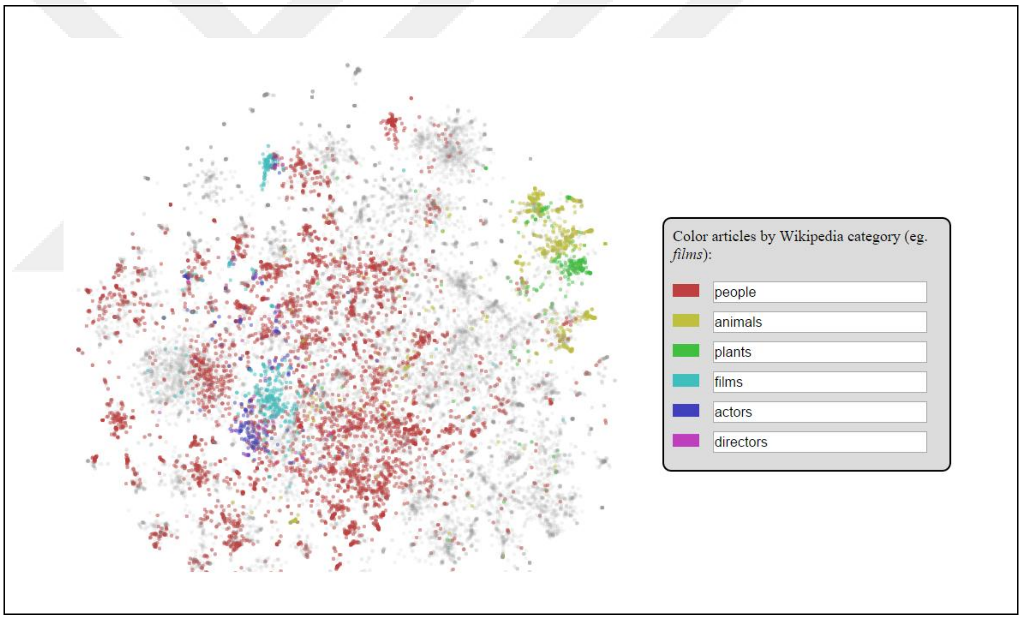


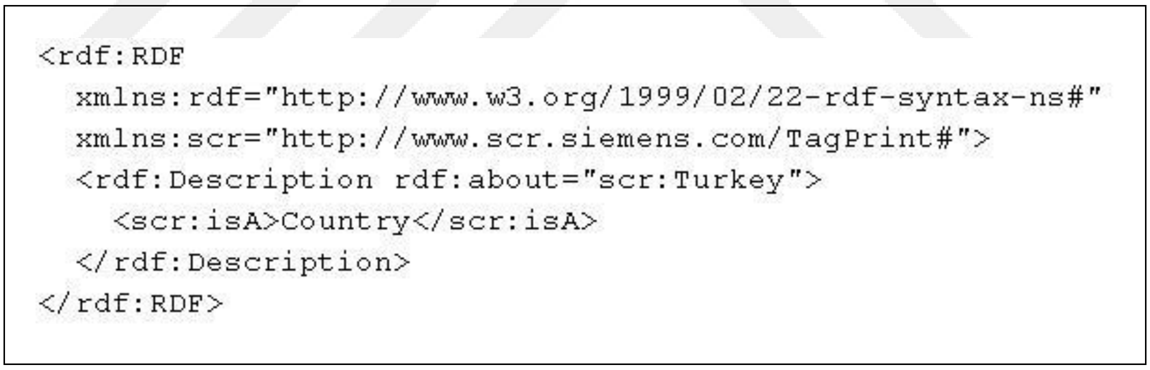
Figure 2.7. Visualization of vector representations of Wikipedia pages using t-distributed stochastic neighbor embedding (t-SNE).

Logically, CBOW must be superior since it includes more words that are right for the situation, but skip-gram performs better in terms of accuracy [41]. Therefore, this study generated word embedding using the skip-gram model of Word2Vec to learn vector representations of Wikipedia articles and these vectors are utilized in the entity linking process.

[7] http://colah.github.io/posts/2015-01-Visualizing-Representations/big_vis/wiki.html

2.3. SEMANTIC WEB TECHNOLOGIES

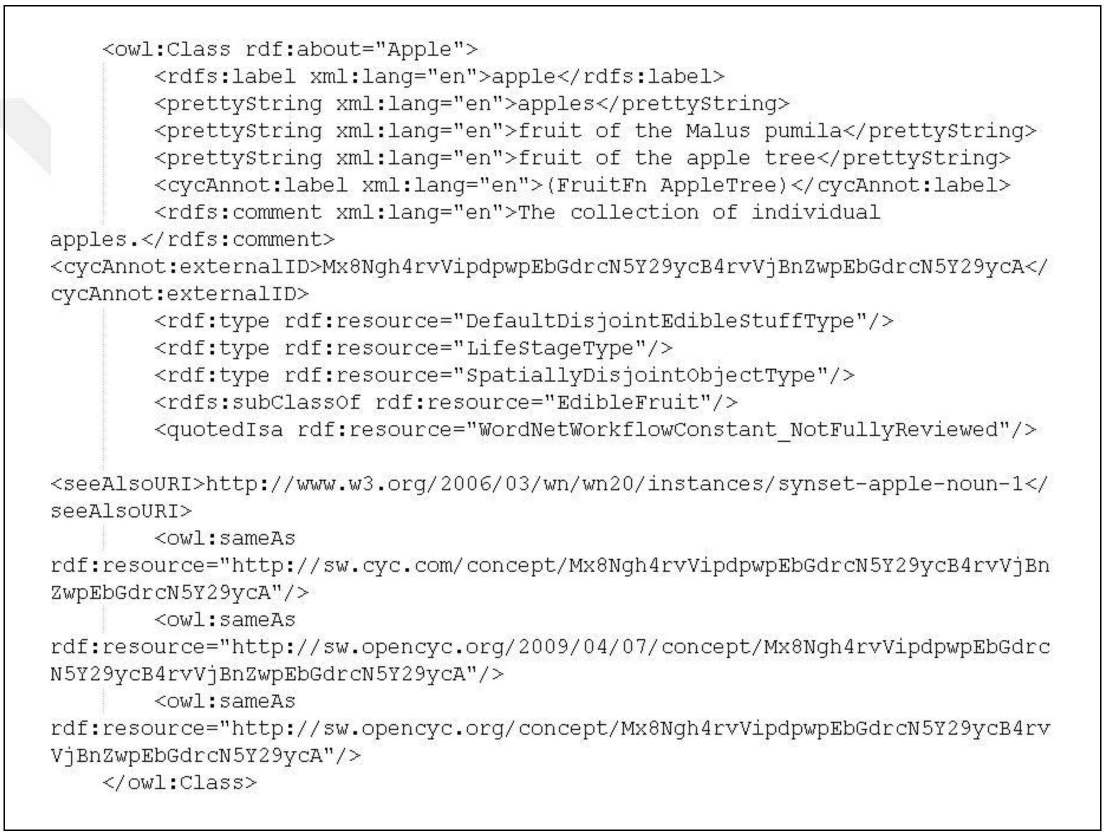
Semantic Web is an extension of the current Web, in which data is represented in a standard format together with metadata that allows integration and processing of different data sources automatically by computer programs [42]. Metadata (data about data) is a structured information about the content and properties of documents in order to support their automatic processing [43]. Semantic Web covers several standard languages for metadata representation [44]. In this study, the Resource Description Framework (RDF) [45] and the Web Ontology Language (OWL) are utilized for representing metadata. RDF is an XML-based language which is essentially a data model for knowledge representation. It represents attributes and relationships with a statement which consists of subject-predicate-object [46]. Figure 2.8 shows how the information "Turkey is a country" could be represented in the standard RDF/XML format (scr:Turkey is the subject, scr:isA is the predicate, and Country is the object).



```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:scr="http://www.scr.siemens.com/TagPrint#">
  <rdf:Description rdf:about="scr:Turkey">
    <scr:isA>Country</scr:isA>
  </rdf:Description>
</rdf:RDF>
```

Figure 2.8. RDF representation of the statement Turkey is a country.

OWL [47] is an XML-based language used for describing ontologies. An ontology is a knowledge source that consists of representational primitives, which is used to model a domain. The representational primitives are usually concepts (classes), instances (objects), relations, and properties [48]. Concepts are the basic elements of an ontology. A concept is described with a definition and a set of properties [49]. Concepts are generally formed in a hierarchical arrangement (taxonomy) by isA-relations (type). For example, the concept "computer science" is a type of the concept "engineering" in WordNet [50]. Instances are named entities of some of the concepts such as people, organizations, geographic locations, books and songs. For example, “Istanbul” is defined as an instance of the concept “city” in WordNet. OWL language is capable of describing concepts, instances, and relationships among them. Figure 2.9 shows the description of the concept “apple (fruit)” in OpenCyc ontology. The label property states the concept name, the prettyString properties state the synonym names, the subClassOf property states the parent concept, and seeAlsoURI states the corresponding WordNet entry of the apple concept, in Figure 2.9.



```
    <owl:Class rdf:about="Apple">
        <rdfs:label xml:lang="en">apple</rdfs:label>
        <prettyString xml:lang="en">apples</prettyString>
        <prettyString xml:lang="en">fruit of the Malus pumila</prettyString>
        <prettyString xml:lang="en">fruit of the apple tree</prettyString>
        <cycAnnot:label xml:lang="en">(FruitFn AppleTree)</cycAnnot:label>
        <rdfs:comment xml:lang="en">The collection of individual
apples.</rdfs:comment>
<cycAnnot:externalID>Mx8Ngh4rvVipdpwpEbGdrcN5Y29ycB4rvVjBnZwpEbGdrcN5Y29ycA</
cycAnnot:externalID>
        <rdf:type rdf:resource="DefaultDisjointEdibleStuffType"/>
        <rdf:type rdf:resource="LifeStageType"/>
        <rdf:type rdf:resource="SpatiallyDisjointObjectType"/>
        <rdfs:subClassOf rdf:resource="EdibleFruit"/>
        <quotedIsa rdf:resource="WordNetWorkflowConstant_NotFullyReviewed"/>

<seeAlsoURI>http://www.w3.org/2006/03/wn/wn20/instances/synset-apple-noun-1</
seeAlsoURI>
        <owl:sameAs
rdf:resource="http://sw.cyc.com/concept/Mx8Ngh4rvVipdpwpEbGdrcN5Y29ycB4rvVjBn
ZwpEbGdrcN5Y29ycA"/>
        <owl:sameAs
rdf:resource="http://sw.opencyc.org/2009/04/07/concept/Mx8Ngh4rvVipdpwpEbGdrc
N5Y29ycB4rvVjBnZwpEbGdrcN5Y29ycA"/>
        <owl:sameAs
rdf:resource="http://sw.opencyc.org/concept/Mx8Ngh4rvVipdpwpEbGdrcN5Y29ycB4rv
VjBnZwpEbGdrcN5Y29ycA"/>
    </owl:Class>
```

Figure 2.9. Description of the “apple (fruit)” using OWL in OpenCyc ontology.

2.4. ENTITY LINKING

Generating assignments of knowledge base entities to documents is called entity linking process. The entity linking studies propose a variety of techniques ranging from hand-coded rules to statistical machine learning techniques. The systems usually utilize NLP methods and knowledge bases (mostly Wikipedia) to detect spots, perform disambiguation and ranking. Figure 2.10 shows an example mapping of a piece of textual content to semantic tags defined in an ontology [51]. The ontological concept classes are represented as circles (i.e. City and Painter), while the concept instances are shown as rectangles (i.e. Philadelphia and Thomas Eakins).

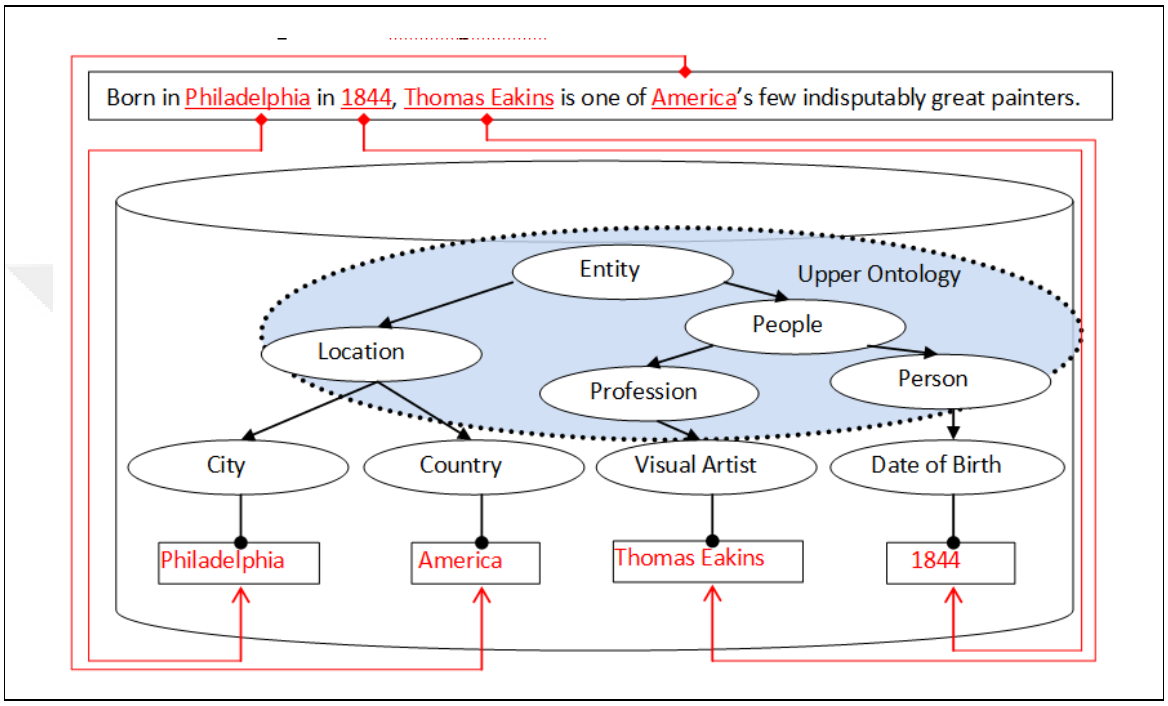


Figure 2.10. An example mapping of a piece of textual content to semantic tags defined in an ontology [6].

Spotting (entity detection) is an important step that may effect the performance of the whole system in terms of computational complexity and accuracy [52]. Thus, a good spotting performance is crucial for an entity linking system. The common approach for spotting is the extraction of noun phrases by using an NLP system and then searching for matching entities in a dictionary generated from a knowledge base. Noun phrases are extracted since entity mentions are typically nouns or noun phrases [53, 54].

Disambiguation is the most challenging step of entity linking. To handle ambiguous entity mentions, context dependent (global) and independent (local) features are used. The context independent features exploit the knowledge about that entity without considering the coherence among other entities in a given text content. The popularity (commonness) of a mention that refers to a particular entity in a utilized collection, is a widely used context independent feature in existing studies [55, 56]. In contrast, the context dependent features are extracted based on the context where the entity mention appears. The context dependent features aim to minimize the semantic distance among entities and optimize the coherence in a given text content. Most of the proposed context dependent features are based on the Wikipedia entity link graph. Ceccarelli et al. [57] evaluate several techniques to calculate relatedness of entities by leveraging the Wikipedia graph structure [58].

Ranking is the final step of entity linking where ranks are detected based on the popularity of the entities and relevancy with the input text. The relevance value is beneficial for especially information retrieval tasks, where the aim is to determine the ranking of the search results for a given query that contains an entity. Ranking process is similar to the problem of keyphrase extraction, which is the task of detecting significant terms that briefly describe the document's content. Three main approaches are utilized for the problem which are extraction based on statistics [59, 60], machine learning [61, 62] and shallow semantic analysis [63]. These approaches can be applied on entity ranking process, too. In this study, we used the statistical approaches such as term frequency to rank entities.

Several entity linking systems were proposed mostly for English: namely, AIDA [64], CMNS [65] and CSAW [66], Illinois Wikifier [67], DBpedia Spotlight [68], TagMe [55] and Wikipedia-miner [56].

AIDA [64] searches for entities using the Stanford NER Tagger and adopts the YAGO2 knowledge base [69] as the catalog of entities, including their semantic distance. Disambiguation comes in three variants: PriorOnly (each mention is bound to its most commonly linked entity in the knowledge base), LocalDisambiguation (each mention is disambiguated independently from others, according to a set of features which describe the mention and the entities), CocktailParty (YAGO2 is used to perform a collective disambiguation which aims maximizing the coherence among the selected annotations, via an iterative graph based approach). AIDA has been designed to deal with English documents of arbitrary length and it offers a publicly available API.

CMNS [65] generates a ranked list of candidate entities for all n-grams in the input text. The list is created through lexical matching and language modeling. The disambiguation is done with a method based on supervised machine learning that takes as input a set of (short) texts and, for each of them, a set of human annotations. CMNS has been designed to deal with very short texts only (mainly tweets).

CSAW [66] searches the input text for entities extracted from Wikipedia anchors and titles. It uses two scores for each annotation, one local and one global. The local score involves 12 features built upon the terms around the mention and the candidate entities. The global score involves all the other annotations detected for the input text and averages the relatedness among them. This was the first system that formulates the disambiguation problem as a quadratic programming optimization problem aiming for a global coherence among all mentions. CSAW has been designed to deal with English documents of arbitrary length, but is quite slow because of the quadratic programming approach.

Illinois Wikifier [67] searches the input text for mentions extracted from Wikipedia anchors and titles, using the Illinois NER system. Disambiguation is formulated as an optimization problem which aims at global coherence among all mentions. It uses a novel relatedness measure between Wikipedia pages. The relatedness measure is based on Normalized Google Distance (NGD) and point-wise mutual information.

DBpedia Spotlight [68] searches the input text for mentions extracted from Wikipedia anchors, titles and redirects. It then associates a set of candidate entities to each mention using the DBpedia Lexicalization data set. Given a spotted mention and a set of candidate entities, both the context of the mention and all contexts of each candidate entity are cast to a Vector-Space Model (using a BOW approach) and the candidate whose context has the highest cosine similarity is chosen. Note that no semantic coherence is estimated among the chosen entities.

TagMe [55] searches the input text for mentions defined by the set of Wikipedia page titles, anchors and redirects. Each mention is associated with a set of candidate entities. Disambiguation exploits the structure of the Wikipedia graph, according to the relatedness measure introduced in [70]. This measure takes into account the amount of common incoming links between two pages. TagMe's disambiguation is enriched with a voting scheme in which all possible meanings of mentions are scored. A proper mix of heuristics is eventually adopted to select the best meaning for each mention. TagMe has been designed to deal with short texts and it offers a publicly available API.

Wikipedia Miner [56] is one of the first approaches proposed to solve the entity annotation problem. This system is based on a machine learning approach that is trained with links and contents taken from Wikipedia pages. Three features are then used to train a classifier that selects valid annotations discarding irrelevant ones: (i) the prior probability that a mention refers to a specific entity, (ii) the relatedness with the context from which the entity is extracted. This is obtained by the non-ambiguous spotted mentions, and (iii) the context quality which takes into account the number of terms involved, the extent they relate to each other, and how often they are used as Wikipedia links.

Cornolti et al. [7] propose a benchmarking framework to compare publicly available entity annotation systems. Figure 2.11 shows performance results of the entity annotation systems for AIDA/CONLL dataset introduced in [7]. Their experimental results show that TagMe outperforms the other annotators in terms of $F1$ score and run-time duration.

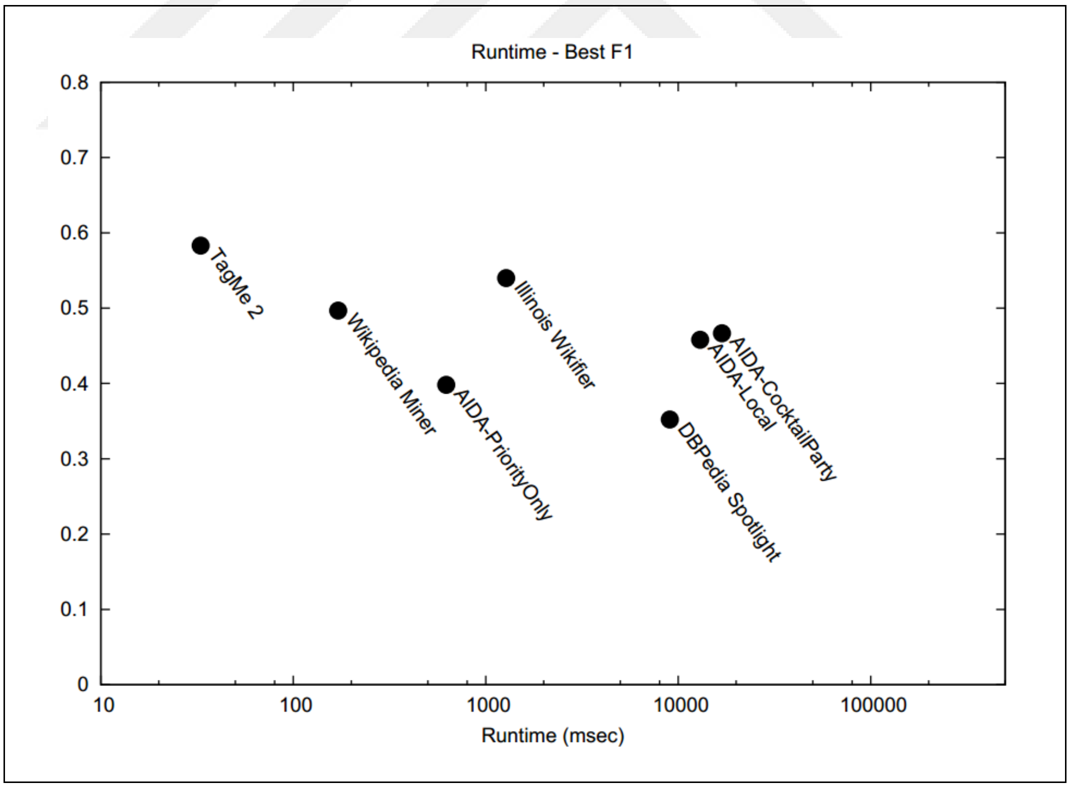


Figure 2.11. Average runtime (in log-scale) for dataset AIDA/CONLL and best achieved F1 measures (metrics based on weak annotation match) [7].

Deep learning is also applied for the entity linking task. Recently, Heck et al. [8] proposed a novel method to learn neural knowledge graph embeddings. The approach learns embeddings directly from structured knowledge representations by using a deep neural network model named Deep Structured Semantic Modeling (DSSM). The architecture for the DSSM is shown in Figure 2.12.

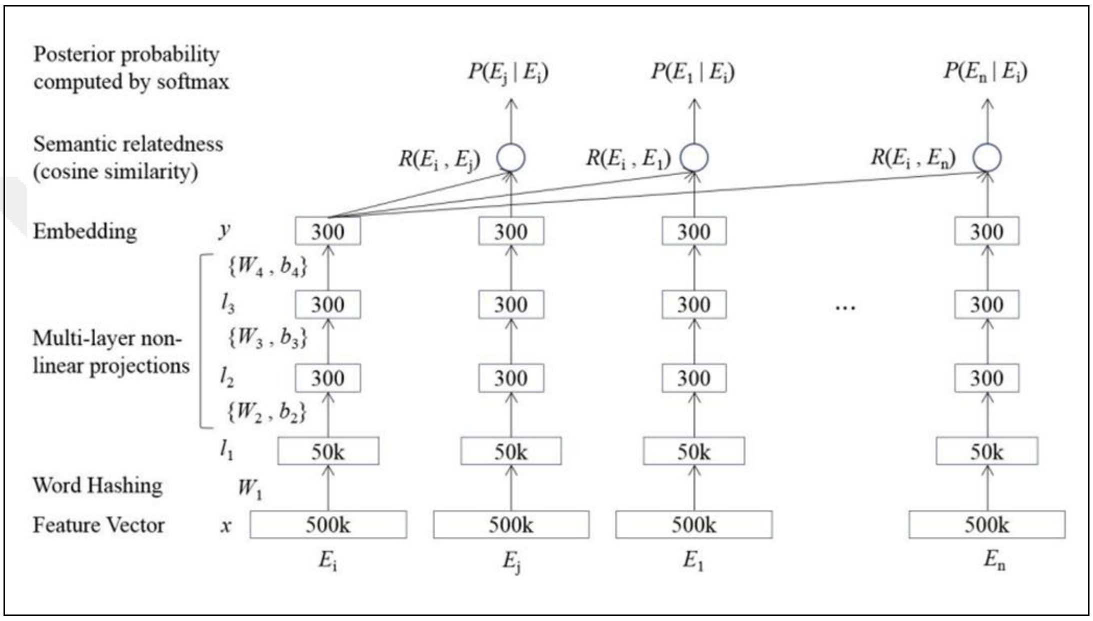


Figure 2.12. The DSSM architecture for learning neural knowledge graph embeddings [8].

Knowledge graph entities often consist of multiple words (e.g., "Yeditepe University", "Barack Obama", "Atlanta Hawks"). Previous extensions to word-based embeddings have typically represented multi-word entities as algebraic combinations (addition) of word-level embeddings. Even though this approach can work for some entities, it often introduces noise into the representation. For example adding the vector "Atlanta" (the city) to the vector "Hawks" (the bird) does not result in the vector for "Atlanta Hawks" (the basketball team). Therefore, the authors [8] use only a portion of the Freebase (entities, relations, and facts) as input features and generate an embedding for each entity. Entity relations are represented with bag-of-words term vectors and entity names are represented using a vector of letter n-grams where a hashing technique is utilized [8].

Word hashing aims to reduce the dimensionality of the bag-of-words term vectors. The specific approach utilized by the authors is based on letter n-grams. As shown in Figure

2.13, given a word (cat), they first add start and end marks to the word (e.g., #cat#). Then, they break the word into letter n-grams (e.g., letter tri-grams:#ca, cat, at#). Finally, the word is represented using a vector of letter n-grams.

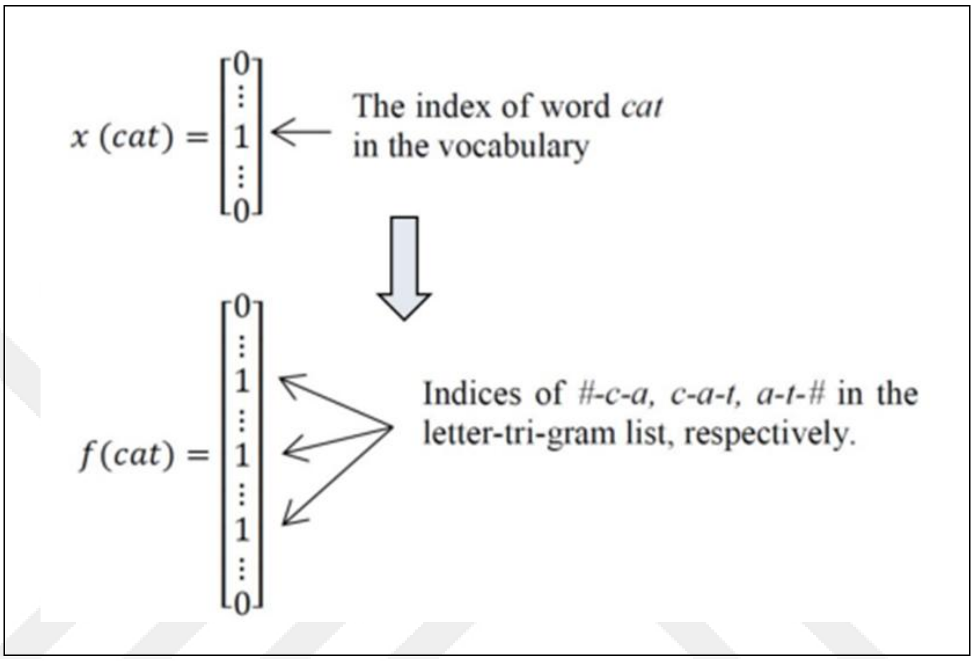

$x\,(cat) = \begin{bmatrix}0\\ \vdots\\ 1\\ \vdots\\ 0\end{bmatrix}$
The index of word *cat* in the vocabulary
$f(cat) = \begin{bmatrix}0\\ \vdots\\ 1\\ \vdots\\ 1\\ \vdots\\ 1\\ \vdots\\ 0\end{bmatrix}$
Indices of *#-c-a, c-a-t, a-t-#* in the letter-tri-gram list, respectively.

Figure 2.13. Word hashing with letter tri-grams [8].

By using word hashing, each entity could be represented by a 30*K* vector instead of a vector with size 500*K*. One potential drawback of word hashing approach is collision, i.e., two different words could have the same letter n-gram vector representation. The authors observe 22 collisions, which is a negligible collision rate of 0.0044.

After the word hashing step, a deep neural network is trained with semantically related (E_i and E_j) and unrelated entities (E_1, ...,E_n) in order to learn 300 dimensional embedding vectors for entities defined in Freebase. The semantic relatedness of two concepts is given by the knowledge graph as first-order related nodes (it can also be inferred from the co-occurrence of entities on a given Wikipedia page). The semantic relevance score between entities E_i and E_j is calculated as: $R(E_i, E_j) = cos(yE_i, yE_j)$, where yE_i and yE_j are the neural embeddings of the entities E_i and E_j, respectively. Given two semantically related concepts, the training procedure computes the posterior probability of entities E_j given E_i using a softmax function, as well as the probabilities for the unrelated concepts E_1, ...,E_n. The

DSSM is trained to maximize the likelihood of the related entities given the features created across the knowledge graph [8]. The generated embeddings are used for entity linking task. The authors compared their linking method with the current state-of-the-art TagMe system and observed better performance (23.6% error reduction).

In contrast to many successful applications for English, there is currently no publicly available entity linking system for Turkish. Although there are some approaches proposed to solve the disambiguation problem in entity linking, there is no complete system that can be considered as a Turkish entity linker. Note that Turkish is a morphologically rich language and it has free word order. Hence, standard approaches developed for English might fail for the Turkish language. Such properties make the language processing task more complex and difficult. Currently, studies for Turkish [22–24] are more focused on the Word Sense Disambiguation (WSD), which is similar to the entity linking task.

One of the WSD studies for Turkish is proposed by Mert et al. [22], which handles ambiguities such as polysemy, homonymy, categorical ambiguity and stemming ambiguity. The authors propose the Lesk-like methods [71,72] for the Turkish language. The methods are evaluated on a randomly selected set of 10 sentences from novels and newspapers that include the ambiguous Turkish word "çay (tea)". The authors observe a 68.57% success rate with the Lesk-like method, which is low compared to state-of-the-art results for English.

Our work can be distinguished from previous work in several ways. First of all, Thinker is the first proposed system for Turkish entity linking process. Secondly, unlike previous work for other languages, Thinker uses fusion of knowledge-based methods and supervised machine learning algorithms that utilize a rich set of features in order to link Turkish entities. Various methods and features that can handle the agglutinative and free word structure of the Turkish language are also proposed in this study. Lastly, a comprehensive Turkish knowledge base is generated by integrating Vikipedi and the Turkish dictionary in order to cover the majority of the Turkish entities.

2.5. FINE-GRAINED ENTITY RECOGNITION

Fine-grained entity recognition is the task of identifying semantic types of entities in the text. A key difference between named entity recognition (NER) is that more specific entity types are used in the fine-grained entity recognition process. For example, basketball player is one of the entity types that would be used for typing basketball players such as "Michael Jordan", which is a more informative type than person.

In contrast to coarse-grained NER [73], there are less fine-grained entity recognition studies [74–78] proposed in the literature. These studies utilize trained classifiers over a variety of linguistic features (i.e. part-of-speech tags, uni-grams, bi-grams) and contextual features (preceding and following words). Specifically, Ling et al. [75] propose FIGER that classifies entity mentions with 112 unique tags curated from Freebase [79] types. They trained a Conditional Random Field (CRF) by utilizing Wikipedia anchor links as training data. Yosef et al. [78] propose HYENA, which is a multi-label classifier based on hierarchical taxonomy of YAGO [80] knowledge base. In this study, a Support Vector Machine (SVM) based classifier is utilized on Wikipedia anchor links likewise of the work by Ling et al. [75].

Word embeddings approach is also applied on the fine-grained entity recognition task. Recently, Yogatama et al. [81] proposed a novel method to learn an embedding for each entity type and each feature. By this way, they can create feature vectors for entity mentions in order to classify entities. They compare their entity recognition method with FIGER and observe better performance (72.35% F1-score).

In contrast to many successful applications for English, there is currently no publicly available fine-grained entity recognition system for Turkish. Whereas, there are some studies [25, 26] proposed to solve the coarse-grained NER task.

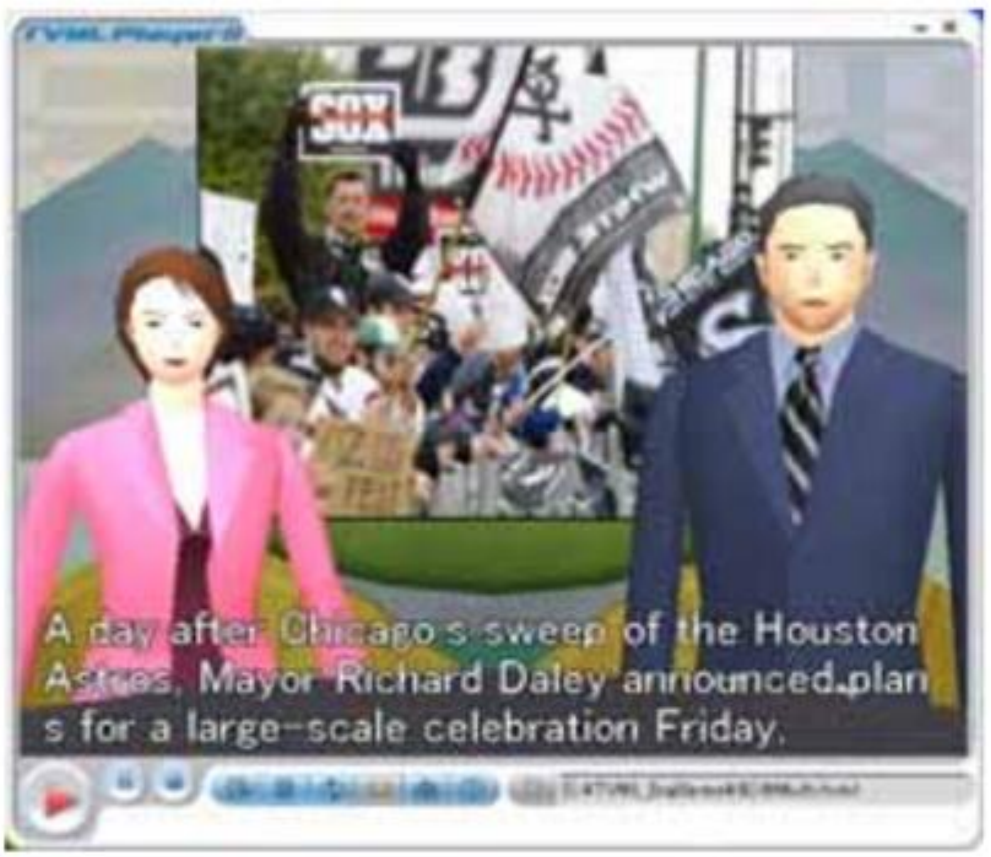
2.6. TEXT VISUALIZATION

Web pages consist of textual and audio-visual content which designate the user experience on websites. Although a Web page may include text, images, sounds, videos and animations, the majority of Web content is predominantly composed of text. There are

Image Generation and Image Retrieval based approaches in the literature to mitigate this situation by converting general text to visual representations.

Image generation based approaches create animations using Computer Graphics technologies for a given text content. Several image generation based studies were proposed mostly for English; namely, U-Pav [9], Web2Animation [10], e-Hon [82] and WordsEye [83].

U-Pav [9] is proposed by Tanaka, where the system reads out the entire text in Web content together with an image animation. The proposed system shows the title and Web content through a ticker and the images in the web page are animated at the same time. The tickers, the animations and the TTS output are synchronized. Figure 2.14 shows a screen-shot from U-Pav.



A day after Chicago's sweep of the Houston Astros, Mayor Richard Daley announced plans for a large-scale celebration Friday.

Figure 2.14. A screenshot from U-Pav - TV-style Web watching [9].

Web2Animation [10] is a system that analyzes the semantics of recipes on the Web and generates 3D animations for them. Figure 2.15 shows internal pipeline of Web2Animation. The recipe instructions are mapped to set of animation clips using a semantic analyzer and the clips are displayed.

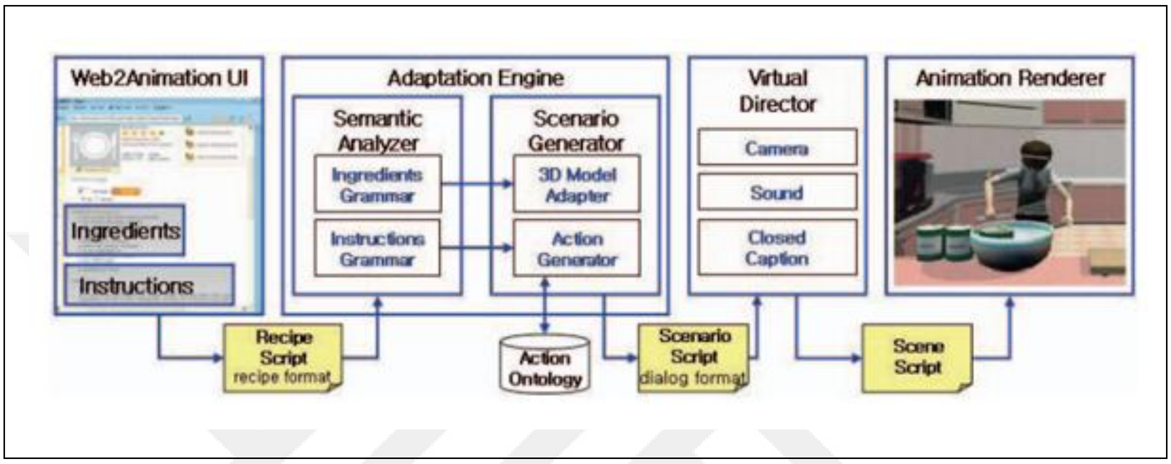


Figure 2.15. Web2Animation Internal Pipeline [10].

E-Hon [82] is a system that converts Web content into a storybook with dialogues and animation. It is especially designed to assist children to understand Web content by animating them. The e-Hon system utilizes semantic tags that are associated with the text on the web. To transform the web contents into dialogues, the system generates a list of subjects, objects, predicates, and modifiers from the text and connects them in a colloquial style. A subject is treated as a character and a predicate is treated as the action. An object is also treated as a character, and an associated predicate is treated as a passive action. Many characters and actions have been recorded in their database.

WordsEye [83] produces highly realistic 3D scenes by utilizing thousands of predefined 3D polyhedral object models with detailed manual tags and deep semantic representations of the text. As a result, WordsEye works best with definite descriptive sentences, e.g., "The cat is 20 feet front of John. The cat is 15 feet tall".

Image retrieval based approaches retrieve images by using image search techniques for a given text content. Such studies generally [84–86] apply NLP techniques to extract important words or phrases and Computer Vision techniques are used to find corresponding images from image databases. Finally, they use Computer Graphics techniques to render the retrieved images in a picture. With recent advancements in computer vision and NLP, there has been significant work in relating images to their sentence-based semantic descriptions [87]. Socher et al. [11] propose a model to map sentences and images into a common embedding space to retrieve one from the other. They introduce the dependency tree recursive neural network (DT-RNN) model which uses dependency trees to embed sentences into a vector space. Then these embeddings can be used in order to retrieve the images that are described by those sentences.

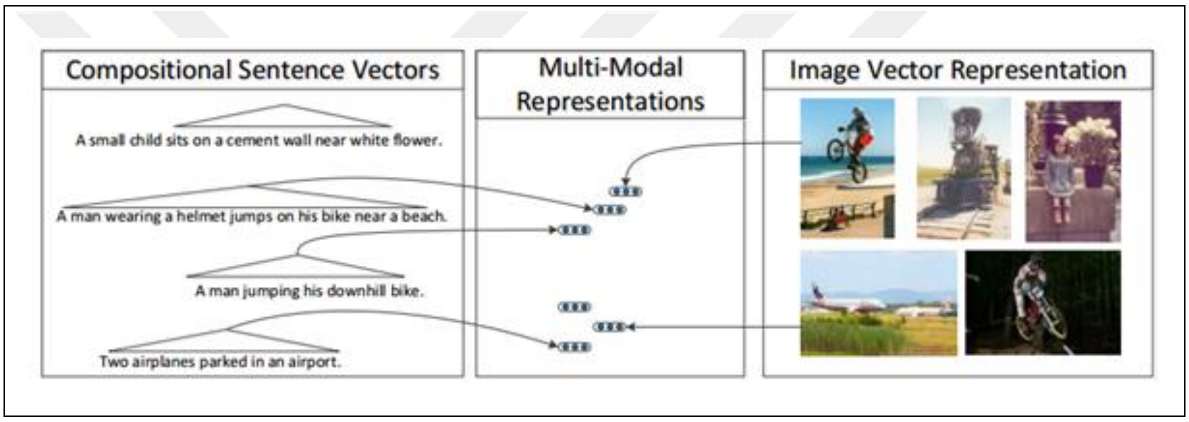

Compositional Sentence Vectors
Multi-Modal Representations
Image Vector Representation
A small child sits on a cement wall near white flower.
A man wearing a helmet jumps on his bike near a beach.
A man jumping his downhill bike.
Two airplanes parked in an airport.

Figure 2.16. The DT-RNN model to query images with a sentence and give sentence descriptions to images [11].

2.7. AUTOMATIC VIDEO GENERATION SOLUTIONS

There are several commercial automatic video generation solutions in the literature: namely, Stupeflix[8], SoMedia[9], Winston[10] and Wibbitz[11].

Stupeflix is a video editing product which provides an environment for semi automatically generating videos from photos, videos and music. Stupeflix uses a custom OpenGL[12] stack

[8] http://studio.stupeflix.com
[9] https://www.somedia.net
[10] http://getwinston.com/project/apptour/
[11] http://www.wibbitz.com/
[12] https://www.opengl.org/

as the video generation technology. Users could connect their social media accounts and Stupeflix generates automated videos from their photo albums. It also provides rich video editing functionality and easy development environment for developers. Figure 2.17 shows a screen-shot from a video, which is generated from a photo album.

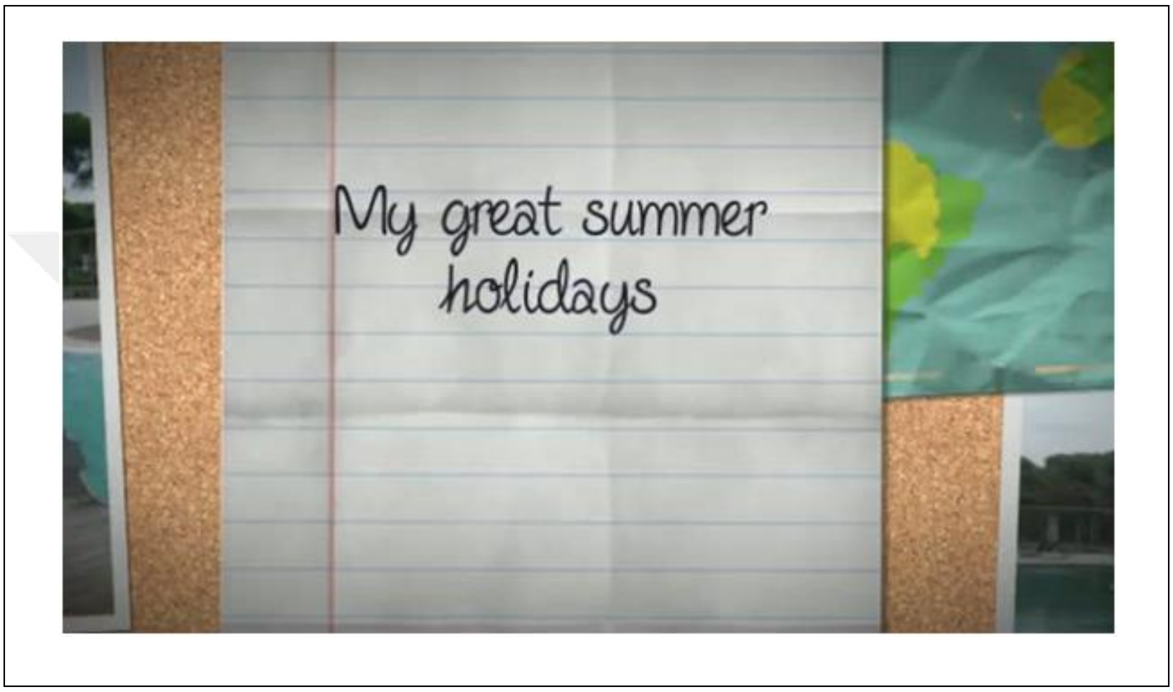


Figure 2.17. Stupeflix presents a photo album.

SoMedia delivers compelling, personalized, and individually targeted video communications. SoMedia has evolved automated video into a user driven video production platform. Using this platform, users can select animation styles, customize colors and music, choose scenes, and upload content. The platform allows users to create a totally customizable motion graphics driven video. Winston is a mobile application that turns social feeds and news interests into an audiovisual newscast. This product is only available for iPhone smart-phone users. Wibbitz re-packages textual content into rich and informative video summaries that can be watched conveniently on mobile screens. Wibbitz converts Web content into video using Artificial Intelligence (AI) and NLP technologies.

3. VIDEOLIZATION - KNOWLEDGE GRAPH BASED VISUAL INTERPRETATION SYSTEM

A notable difference between the contents of a Web page and a TV program is that the former is a document-based information media whereas the latter is a time-based continuous information media. This situation creates a difference for the information accessing methods that can be utilized. Conventional "Web browsing" is an active process for accessing information. On the other hand, conventional "TV watching" is a relatively passive way of accessing information. In order to consume Web content effectively on a TV, a media conversion product which enhances TV watching experience, is needed. This problem of consuming Web on TV motivated us for the development of a Videolization system.

Videolization is a knowledge graph based visual interpretation system that aims to automatically create TV program contents in a video format from Web content and provide passive consumption service for TV users. Via Videolization product, TV users can watch their favorite Web content instead of having to read it. TV program contents are generated in several categories based on the source and type of the Web content. Each category is presented in a separate TV channel to the TV audience. For instance, Social Networks are presented like TV channels and posts from those sources are played just like TV episodes. A few possible channels of a Videolization product are listed below:

- *Encyclopedia Channel*: Information about entities are presented to the user. For instance, when user asks for |*Yeditepe University*| entity, the channel would present a video about Yeditepe University with important facts about it such as foundation year, number of students, etc..
- *News Channel*: Online news documents are presented to the user. News documents are categorized into several categories such as politics, sports, etc. allowing users to watch news from different categories based on their preference.
- *Social Network Channel*: Posts from the Social Network platforms are presented to the user. By watching this channel, the user can get a brief report about his or her recent social media activities.

3.1. VIDEOLIZATION CHANNELS

TV program contents are generated in several categories based on the source and type of the Web content. Each category is presented in a separate TV channel to the audience. As a use case, Encyclopedia Channel is developed to present Wikipedia articles in video format. It is a Web application that allows the users to search Wikipedia article titles and watch the generated videos. User Interface of Encyclopedia Channel is developed by considering Human Computer Interaction (HCI) studies [88] and design principles for the usability. Figure 3.1 shows a screenshot of the Encyclopedia Channel.

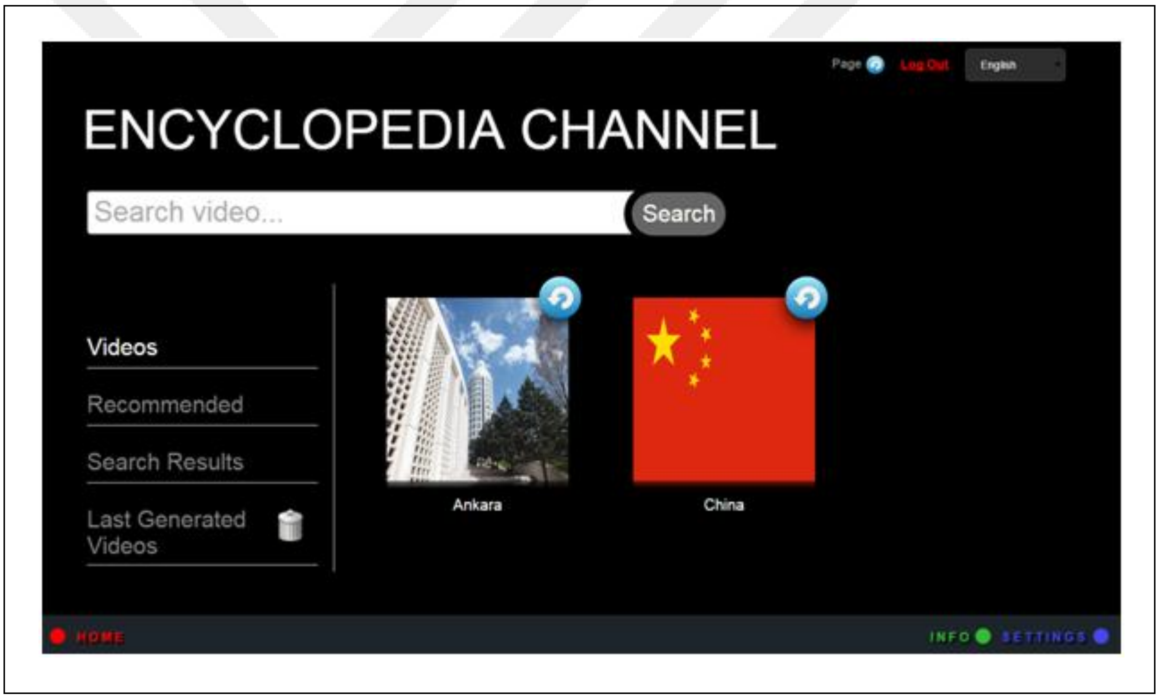


Figure 3.1. A screenshot of Encyclopedia Channel.

3.2. REPOSITORY

Repository module stores generated video files and utilized knowledge graph. MongoDB[13], a NoSQL database, is used for this purpose. In order to access and store the video data,

[13] https://www.mongodb.com/

Spring Data[14] is used for this. Key functional areas of Spring Data are a plain old Java object (POJO) centric model for interacting with a MongoDB records and easily writing a repository style data access layer. A POJO is an ordinary Java object, not bound by any special restriction and not requiring any class path. By this way, there is no need for entity–relationship model in order to store data in a database. Therefore modeling of only Java classes is adequate. Figure 3.2 shows our class diagram for storing the data and Figure 3.3 shows the list of video records stored in the repository.

[14] http://projects.spring.io/spring-data-mongodb/

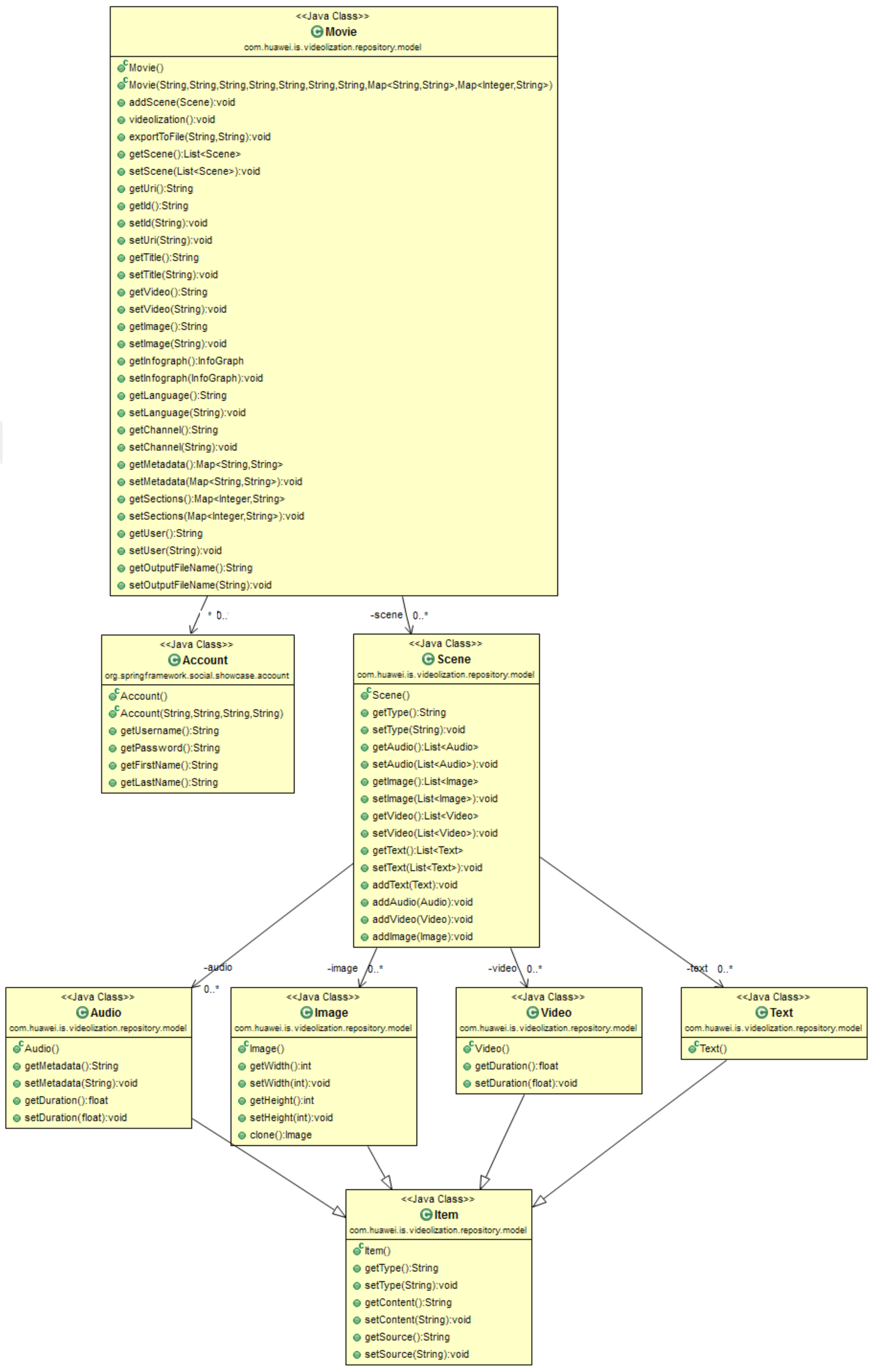
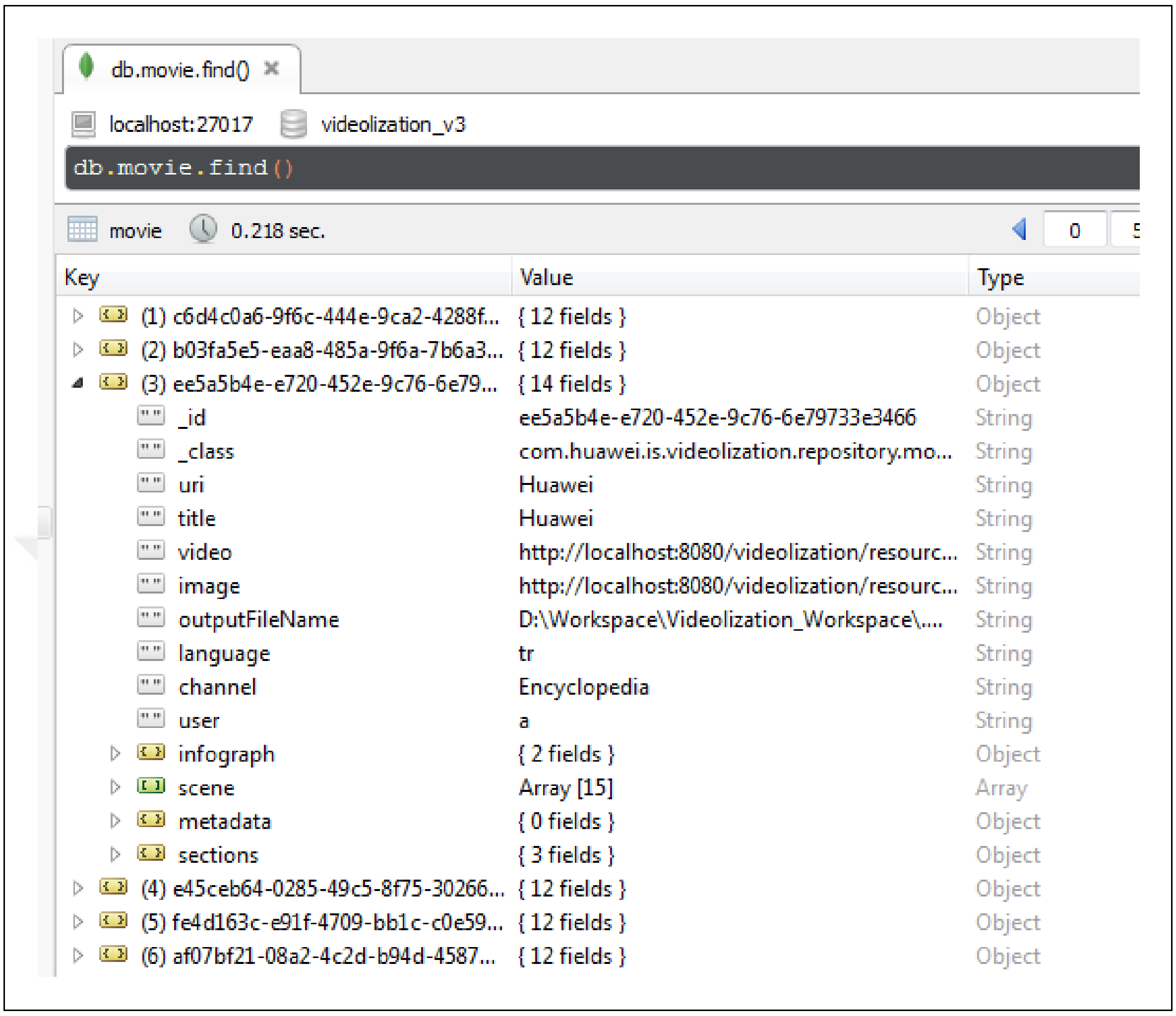


Figure 3.2. Videolization class diagram.



db.movie.find()

localhost:27017 videolization_v3

```
db.movie.find()
```

movie 0.218 sec. 0

Key	Value	Type
(1) c6d4c0a6-9f6c-444e-9ca2-4288f...	{ 12 fields }	Object
(2) b03fa5e5-eaa8-485a-9f6a-7b6a3...	{ 12 fields }	Object
(3) ee5a5b4e-e720-452e-9c76-6e79...	{ 14 fields }	Object
_id	ee5a5b4e-e720-452e-9c76-6e79733e3466	String
_class	com.huawei.is.videolization.repository.mo...	String
uri	Huawei	String
title	Huawei	String
video	http://localhost:8080/videolization/resourc...	String
image	http://localhost:8080/videolization/resourc...	String
outputFileName	D:\Workspace\Videolization_Workspace\....	String
language	tr	String
channel	Encyclopedia	String
user	a	String
infograph	{ 2 fields }	Object
scene	Array [15]	Array
metadata	{ 0 fields }	Object
sections	{ 3 fields }	Object
(4) e45ceb64-0285-49c5-8f75-30266...	{ 12 fields }	Object
(5) fe4d163c-e91f-4709-bb1c-c0e59...	{ 12 fields }	Object
(6) af07bf21-08a2-4c2d-b94d-4587...	{ 12 fields }	Object

Figure 3.3. An illustration of the list of video records stored in the repository.

In the context of this module, we also carried out generation of a simplified version of DBpedia knowledge graph. DBpedia contains metadata about entities. However, most of the information consists of details or intermediate information not suitable to be presented in the generated videos for the end users. Thus, we decided to identify most significant properties of entities and filter out the rest. For example, for a company entity, founder, foundation year, industry are most informative properties and worth to present in a video.

In order to achieve simplified version of DBpedia, we decided to manually determine the significant properties, since DBpedia schema barely changes and there are not so many defined property types. Firstly, we created histogram of DBpedia properties and this is resulted with a list of 1367 distinct properties. Then most frequent and informative 100 properties are manually identified (listed in Table 3.1).

Table 3.1. List of curated 100 DBpedia properties used in this study.

Team	City	Maximum Elevation (µ)	Ship Beam (µ)
Founder	Film Director	Religion	Architectural Style
Type	Language	Minimum Elevation (µ)	Year
Description	Home Town	Motto	Location City
Birth Place	Alma Mater	Located In Area	Series
Birth Year	Nationality	Department	Creator
Country	Founding Year	Ground	Number Of Episodes
Genre	Population Density	Headquarter	Origin
Death Year	Weight (g)	Builder	Number Of Pages
Location	Length (µ)	Opening Year	Number Of Employees
Family	Party	Known For	Coached Team
Starring	Publisher	Cinematography	Former Name
Population Total	Birth Name	Field	Broadcast Area
Occupation	Owner	Combatant	Affiliation
Death Place	Music Composer	Network	Programme Format
Class	Musical Artist	Strength	Family
Elevation (µ)	Author	Key Person	Route Start
Runtime (s)	Computing Platform	Military Command	Route End
Producer	Album	Band Member	Manufacturer
Position	Product	Literary Genre	River Mouth
Release Date	Commander	Spouse	Engine
Area Total (m2)	Industry	Developer	Batting Side
Height (µ)	Order In Office	Number Of Students	Draft Year
Writer	Year Of Construction	County	Country
Performer	League	Parent	Population As Of

3.3. CONTENT CURATION

Content Curation is the core module that provides the main functionality of the system. This module transforms acquired Web content such as social feeds, Web document, news RSS, etc. into structured visual representation format using Semantic Web technologies. It analyzes the collected Web content and firstly determines what to present to the individual TV audience. Then the module decides about how to present the selected content. This system is our main research focus and Semantic Web techniques are utilized for the design and implementation of the system.

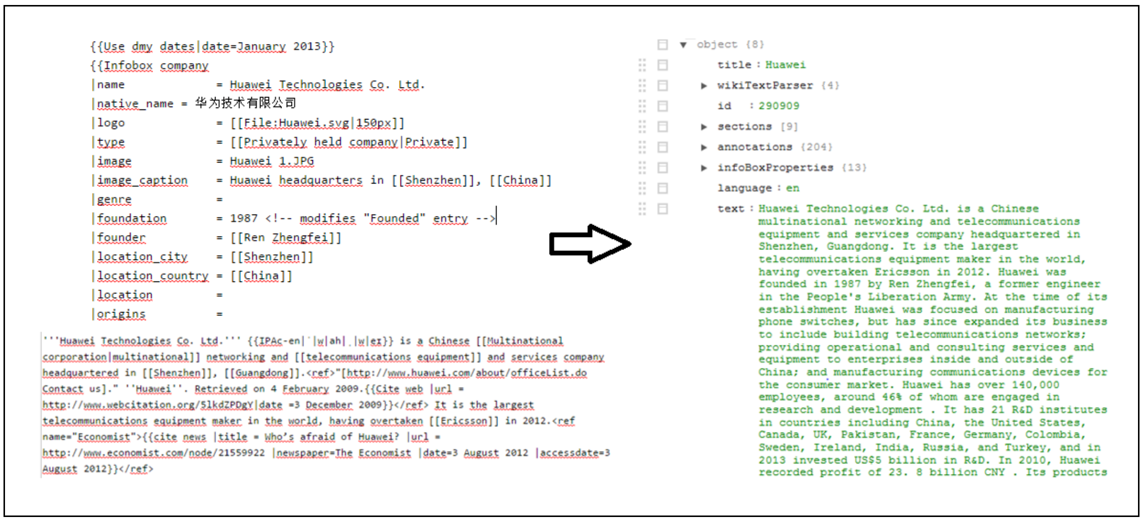
The final output of this module is an XML/JSON file, more specifically; a "Videolization Video Description Language" (VVDL) file that defines the content of the video that would be generated. By using a 3rd party video visual effects and compositing application such as Adobe After Effects[15], it is possible to generate videos from the VVDL file by using a specialized template and a parser for the VVDL.

Our methodology for the Content Curation has three main processing steps: Data Acquisition, Semantic Analysis and Information Presentation.

3.3.1. Data Acquisition

Data Acquisition deals with the Web data extraction and retrieval challenges. Various kinds (social media, news, product) and types of (xml, html, image, video) Web resources are collected in order to be presented on the Videolization channels. For this study, we realized Encyclopedia Channel by parsing dump of Wikipedia[16], which is provided in wikitext format and then the data is converted into JSON format. Figure 3.4 illustrates parsing of a given Wikipedia article. The left side of the figure shows the input wikitext and the right side shows the parsing result in JSON format.

[15] http://www.adobe.com/products/aftereffects.html
[16] https://dumps.wikimedia.org/enwiki/



```
{{Use dmy dates|date=January 2013}}
{{Infobox company
|name             = Huawei Technologies Co. Ltd.
|native_name = 华为技术有限公司
|logo             = [[File:Huawei.svg|150px]]
|type             = [[Privately held company|Private]]
|image            = Huawei 1.JPG
|image_caption    = Huawei headquarters in [[Shenzhen]], [[China]]
|genre            =
|foundation       = 1987 <!-- modifies "Founded" entry -->
|founder          = [[Ren Zhengfei]]
|location_city    = [[Shenzhen]]
|location_country = [[China]]
|location         =
|origins          =
'''Huawei Technologies Co. Ltd.''' {{IPAc-en|ˈ|w|ah|ˌ|w|eɪ}} is a Chinese [[Multinational
corporation|multinational]] networking and [[telecommunications equipment]] and services company
headquartered in [[Shenzhen]], [[Guangdong]].<ref>"[http://www.huawei.com/about/officeList.do
Contact us]." ''Huawei''. Retrieved on 4 February 2009.{{Cite web |url =
http://www.webcitation.org/5lkdZPDgY|date =3 December 2009}}</ref> It is the largest
telecommunications equipment maker in the world, having overtaken [[Ericsson]] in 2012.<ref
name="Economist">{{cite news |title = Who's afraid of Huawei? |url =
http://www.economist.com/node/21559922 |newspaper=The Economist |date=3 August 2012 |accessdate=3
August 2012}}</ref>
```

```
▼ object {8}
     title : Huawei
   ▶ wikiTextParser {4}
     id   : 290909
   ▶ sections [9]
   ▶ annotations {204}
   ▶ infoBoxProperties {13}
     language : en
     text : Huawei Technologies Co. Ltd. is a Chinese
            multinational networking and telecommunications
            equipment and services company headquartered in
            Shenzhen, Guangdong. It is the largest
            telecommunications equipment maker in the world,
            having overtaken Ericsson in 2012. Huawei was
            founded in 1987 by Ren Zhengfei, a former engineer
            in the People's Liberation Army. At the time of its
            establishment Huawei was focused on manufacturing
            phone switches, but has since expanded its business
            to include building telecommunications networks;
            providing operational and consulting services and
            equipment to enterprises inside and outside of
            China; and manufacturing communications devices for
            the consumer market. Huawei has over 140,000
            employees, around 46% of whom are engaged in
            research and development . It has 21 R&D institutes
            in countries including China, the United States,
            Canada, UK, Pakistan, France, Germany, Colombia,
            Sweden, Ireland, India, Russia, and Turkey, and in
            2013 invested US$5 billion in R&D. In 2010, Huawei
            recorded profit of 23. 8 billion CNY . Its products
```

Figure 3.4. Parsing illustration of a given Wikipedia article.

3.3.2. Entity Linking

Acquired Web content is first semantically analyzed to understand the context by performing entity linking (semantic annotation). In this study, we used TagMe for entity annotation of English documents and Thinker for entity annotation of Turkish documents. The details of TagMe system are introduced in Section 2.4 and the Thinker system is presented in detail in Chapter 4. These entity linking systems provide list of entities for a given text content with their significance values. Then we retrieve properties of these entities from the Videolization Knowledge Graph to enrich the video content.

3.3.3. Information Presentation

A template based approach is utilized to render the analyzed text content into video format. The template includes semantic and functional rules. In this study, we realized the Encyclopedia Channel by using our template based visualization approach. In the Encyclopedia Channel, first summarization function is utilized to create a more concise representation which will still retain the most important sentences. The videolization system utilizes an extraction based automatic text summarization sub-module [89]. After summarization, each sentence is videolized separately by forming audio and video components. For audio component, the TTS is generated and optionally background music is added. For visual part, semantic analysis is performed to map each sentence into one of the four videolization scene types; namely Entity Graph, Entity Video, Entity Image and Text representations. Figure 3.5 demonstrates the decision process and Figure 1.2 illustrates these scene types for a given Wikipedia article.

```
summary = summarization_service(document);
while sentence in summary do
    sentence.scene = Text;
    tts = tts_service(sentence);
    if tts.length < min_scene_duration then
        sentence.scene = IgnoredSentence;
        next sentence;
    end
    entities = semantic_linking_service(sentence);
    sort entities by importance;
    for entity in entities do
        count = document.count(entity);
        if used(entity) or count < min_occurrence then
            next entity
        else if has entity graph(entity) then
            sentence.scene = EntityGraph;
            break;
        else if has entity video(entity) then
            sentence.scene = EntityVideo;
            break;
        else
            sentence.scene = EntityImage;
            break;
        end
    end
end
```

Figure 3.5. Rule based scene selection algorithm for an input document

3.3.4. Scene Types

For all scene types we have two general rules related to entity selection:

(i) The entity must occur in the whole document at least twice.

(ii) If there are multiple candidates, the one with the highest TagMe or Thinker significance weight is chosen.

We enforce these rules in order to select more coherent entities in the document. Additional rules exist for each scene type and they are listed as follows:

Entity Graph: A sentence can be videolized as an Entity Graph scene if it has an entity that fulfills the following conditions:

(i) The candidate entity must have at least two properties from Table 3.1.

(ii) It must not have been presented as an Entity Graph in the previous sentences.

The call has_entity_graph in Figure 3.5 refers to these conditions. The visual component of the video is created using the selected entity's key properties and its image. Figure 3.6 shows our design for visualization of an entity graph type Wikipedia sentence ("London is the capital city of both England and the United Kingdom").

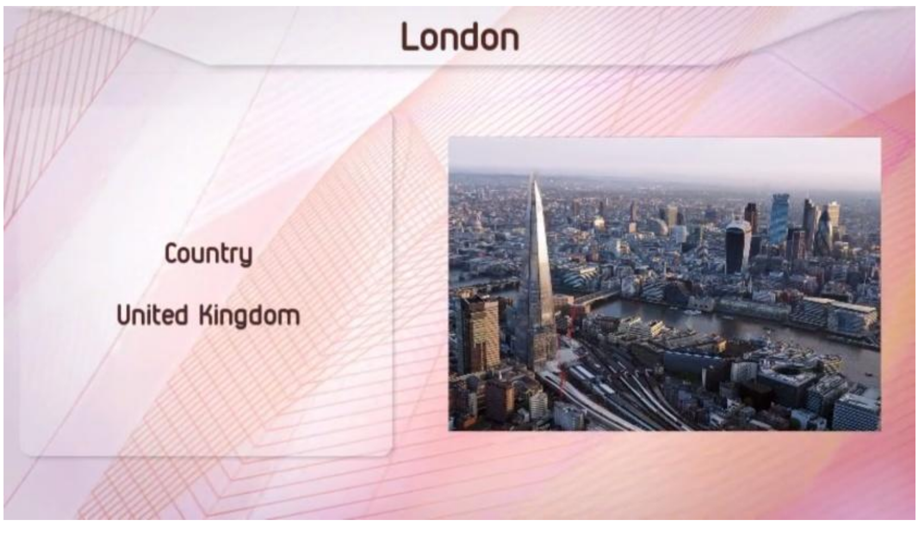


Figure 3.6. Visualization of a sentence, which is categorized as entity graph.

Entity Video: Similar to the entity graph type of scenes, most significant entity of a sentence is selected for visualization purposes. If the selected entity does not have any significant property value, system may show a video that represents the entity, if the following conditions are fulfilled:

(i) The candidate entity does not fulfill the conditions of Entity Graph.
(ii) It must not have been presented as an Entity Video in the previous sentences.
(iii) A video related to the entity can be found in available repositories.

The call has_entity_video in Figure 3.5 refers to these conditions. We utilize Shutterstock[17] website as a video repository. Along with Entity text, its type and document title are also used as search terms to handle disambiguation (Apple company or fruit) and context relevancy problems.

Additionally we filter the search results based on video duration. The minimum video duration is determined based on the duration of TTS audio file. The maximum video duration is fixed for 60 seconds. When the duration of the video found is longer than TTS duration, we apply a greedy algorithm to resolve audio-video synchronization issues. As long as total audio duration is shorter than the video duration, the following sentences in the text are concatenated to the current scene and their TTS are read out over the video. If all the remaining sentences are consumed and the video is still not finished, the video is cut with a transition effect.

Entity Image: A sentence can be videolized as an Entity Image scene if it has an entity that fulfills the following conditions:

(i) The candidate entity does not fulfill the conditions of Entity Graph or Video
(ii) It must not have been presented as an Entity Image in the previous sentences.

The call has_entity_image in Figure 3.5 refers to these conditions. We utilize images from Entity's Wikipedia page and the results from Google Image search. Wikipedia articles generally have at least one associated image and this representation is preferred by the Videolization System. If the Entity's Wikipedia page does not have an image, we perform a search using Entity text, its type and document title. The top result for this search query is used as the Entity Image.

[17] https://http://www.shutterstock.com/

Text: This scene type can be considered as a fallback visualization option. It is the least preferred option and it is only utilized when the other scene types' conditions are not fulfilled. Generally, this alternative comes into play when no entity in the sentence can be detected. The visual component of the video is created by simply depicting the sentence text in a TV friendly way. Figure 3.7 shows our design for visualization of this type of Wikipedia sentence.

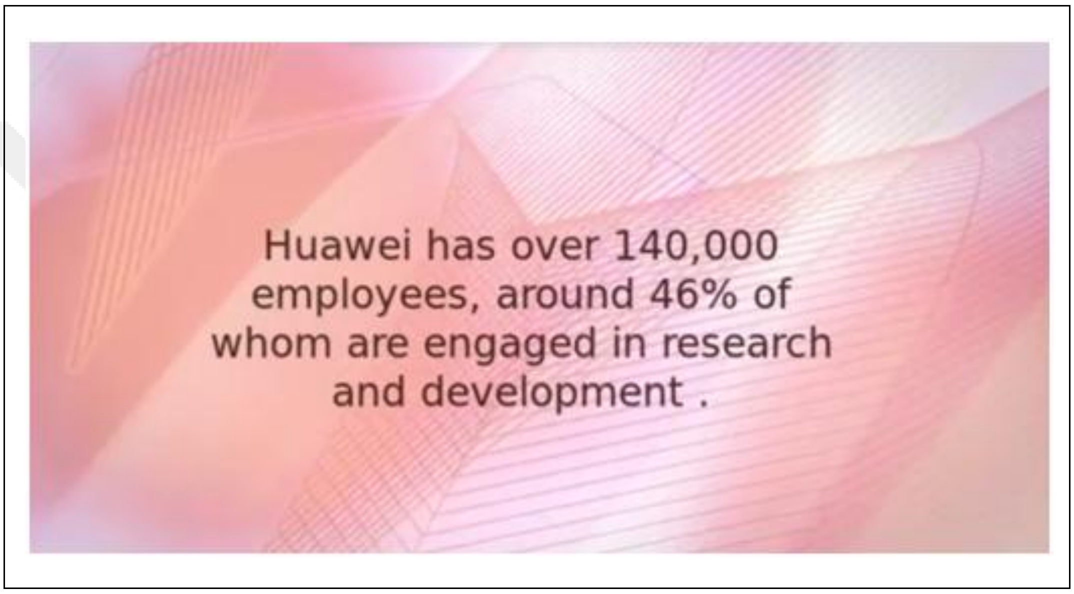

Huawei has over 140,000 employees, around 46% of whom are engaged in research and development .

Figure 3.7. Visualization of a Wikipedia sentence, which is categorized as text.

3.3.5. VVDL Output

The final output of the content curation module is an XML/JSON file, more specifically; a "Videolization Video Description Language" (VVDL) file that defines the content of the video that would be generated. A VVDL file includes scene elements and related configuration. Figure 3.8 shows an example of a VVDL file.

```
 1 [{
 2 "Videolization": {
 3   "id": unique_id,
 4   "title": the Title of Wikipedia page,
 5   "language": "en",
 6   "channel": "Encyclopedia",
 7   "duration": video duration in seconds,
 8   "Scenes":[{
 9     "t": "Entity Graph",
10     "start":scene start sec., "end":scene end sec.,
11     "audio": [ TTS information ],
12     "text": [{
13       "t": "title", "c": Entity (Subject) },{
14       "t": "eg", "c": Predicate_1:Object_1 },{
15       "t": "eg", "c": Predicate_2:Object_2 },{
16       }]
17     },{
18     "type":"Entity Image",
19     "start":scene start sec., "end":scene end sec.,
20     "audio": [ TTS information ],
21     "image": /path/to/image
22     "text": [{
23       "t": "sentence", "c": the sentence itself }
24       ],
25     }]
26   }
27 }]
```

Figure 3.8. An example of Videolization Video Description Language file for Encyclopedia channel is listed. This example has two scenes; an Entity graph scene (9-16) with three fields and an Entity Image scene (18-24) with one image. For some fields abbreviated versions are used; "t" stands for "type"; "c" stands for "content" and "eg" stands for "entity graph".

VVDL provides interdependence between content curation and video generation modules. Same content could be visualized differently with different visual templates. Alternatively, it is possible to generate videos using a 3rd party tool such as Adobe After Effects from the VVDL file by using a specialized template and a parser for the VVDL. Figure 3.9 presents different visualizations of same VVDL file.

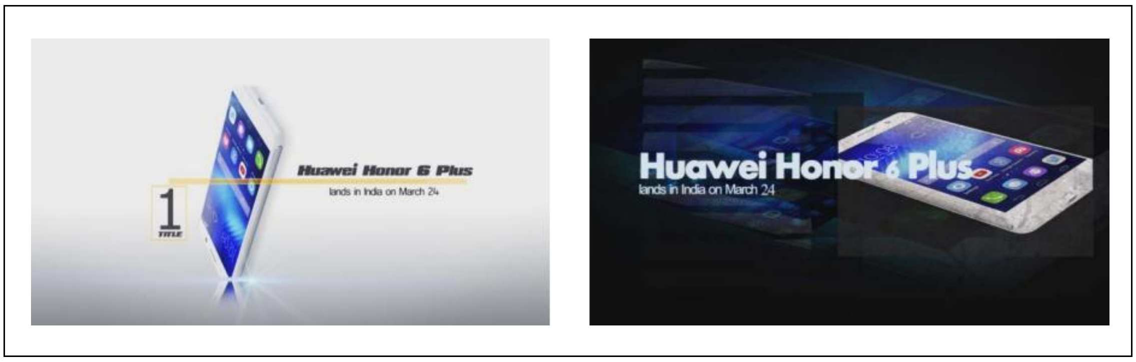


Figure 3.9. Sample scenes from a video produced through Adobe After Effects using a template and a parser for VVDL format. Generalized nature of VVDL format allows Content Curation output to be used in third party applications.

4. THINKER - ENTITY LINKING SYSTEM FOR TURKISH LANGUAGE

The amount of unstructured data has increased exponentially in recent years and Web resources form the vast part of it, including tweets, blogs, online news, comments, etc. Leveraging these resources by automatic processing is highly challenging due to the ambiguity of natural language [90]. The data needs to be transformed into a standard format that includes metadata in order to become beneficial for many information retrieval and extraction tasks such as semantic search, question answering and summarization systems.

Entity linking is one of the problems to be handled in order to process natural language and to enrich the existing unstructured text with metadata. The generation of assignments between knowledge base entities and lexical units is called entity linking. Entity linking process has to handle the ambiguity of the natural language since an entity mention in a text might have more than one corresponding entity defined in the utilized knowledge base.

Entity linking is similar to the problem of word sense disambiguation (WSD) [91]. WSD is the process of automatically mapping a polysemous word (i.e., a word having many meanings) to an appropriate sense (meaning) according to the context in which it is used. However, entity linking is a more complex task compared to WSD. In WSD process, the utilized lexical resource is complete by covering all senses. In contrast, there is no existing knowledge base that covers all entities. Hence an entity linker system is required to mark entity mentions without knowledge base entries as NIL (unknown entity). In addition, entity linking has a higher variation compared to lexical mentions in WSD [52].

State-of-the-art approaches usually utilize three steps for entity linking [58]. (i) Spotting of input text; that is finding mentions (fragments of text) and corresponding candidate entities defined in a knowledge base; (ii) Disambiguation of mentions; where each mention is linked to the correct entity (meaning) in that context; (iii) Ranking; where the detected entities are sorted based on their popularity and relevancy with the input text.

In this chapter, first we introduce a high performance entity linking system - Thinker - for Turkish that automatically maps entity mentions in a text with the corresponding real world entities defined in Vikipedi or the Turkish dictionary published by Turkish Language Association (TLA). A rich set of features for Turkish language have been utilized in this study; including extraction of entity embeddings by using unsupervised deep learning approaches. This approach forms the core part of the study. Then, we introduce a high performance entity discovery system for Turkish language. The system semi automatically detects entity mentions that are not defined in Vikipedi for a given Turkish text corpus and discovers semantic typing of detected unlinkable entity mentions.

As shown in Figure 4.1, Turkish Entity Linker has been realized through the design and implementation of three major modules: Linking User Interface, Knowledge Base and Entity Linking Pipeline.

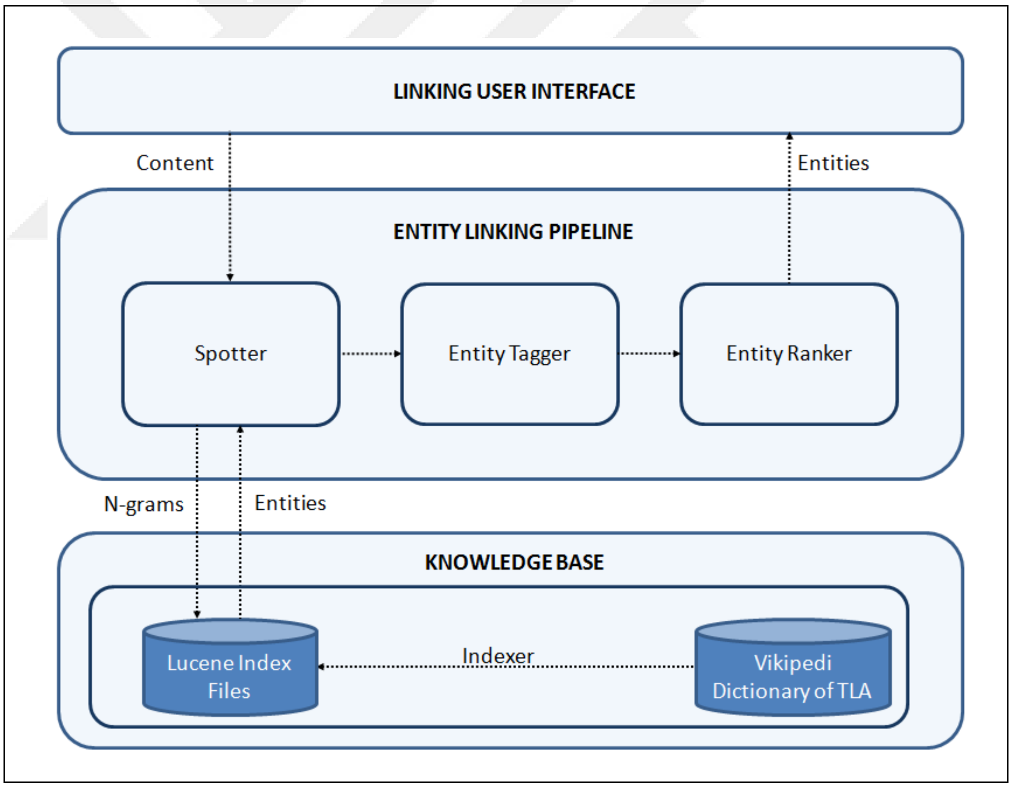
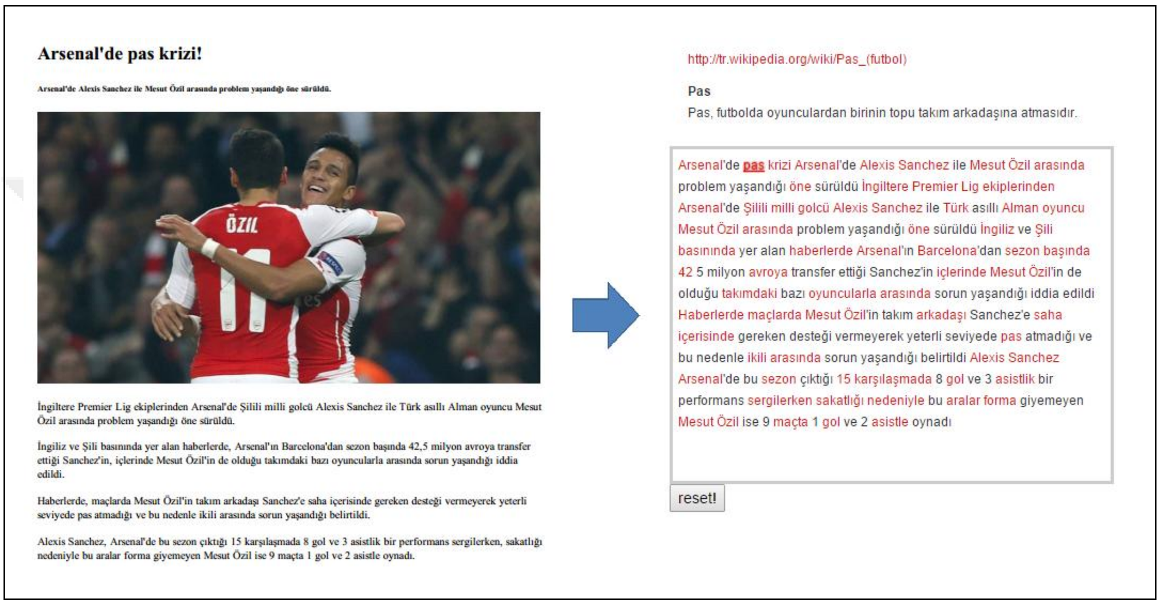


Figure 4.1. Thinker architecture.

4.1. LINKING USER INTERFACE

The Linking User Interface (UI) is used for presenting the identified entities. Turkish Entity Linker UI is a web application that users can input Turkish text content and see the linking results. Figure 4.2 shows an example of how a Turkish news article[18] content is mapped into corresponding entities defined in Vikipedi or the Turkish dictionary. Right side of the image is the Thinker User Interface and shows the linking result. The identified entities are represented in red color and the rest are in black. When a user places the mouse over an entity, she can see the details of the linked entity.



Arsenal'de pas krizi!

Arsenal'de Alexis Sanchez ile Mesut Özil arasında problem yaşandığı öne sürüldü.

İngiltere Premier Lig ekiplerinden Arsenal'de Şilili milli golcü Alexis Sanchez ile Türk asıllı Alman oyuncu Mesut Özil arasında problem yaşandığı öne sürüldü.

İngiliz ve Şili basınında yer alan haberlerde, Arsenal'ın Barcelona'dan sezon başında 42,5 milyon avroya transfer ettiği Sanchez'in, içlerinde Mesut Özil'in de olduğu takımdaki bazı oyuncularla arasında sorun yaşandığı iddia edildi.

Haberlerde, maçlarda Mesut Özil'in takım arkadaşı Sanchez'e saha içerisinde gereken desteği vermeyerek yeterli seviyede pas atmadığı ve bu nedenle ikili arasında sorun yaşandığı belirtildi.

Alexis Sanchez, Arsenal'de bu sezon çıktığı 15 karşılaşmada 8 gol ve 3 asistlik bir performans sergilerken, sakatlığı nedeniyle bu aralar forma giyemeyen Mesut Özil ise 9 maçta 1 gol ve 2 asistle oynadı.

http://tr.wikipedia.org/wiki/Pas_(futbol)

Pas

Pas, futbolda oyunculardan birinin topu takım arkadaşına atmasıdır.

Arsenal'de pas krizi Arsenal'de Alexis Sanchez ile Mesut Özil arasında problem yaşandığı öne sürüldü İngiltere Premier Lig ekiplerinden Arsenal'de Şilili milli golcü Alexis Sanchez ile Türk asıllı Alman oyuncu Mesut Özil arasında problem yaşandığı öne sürüldü İngiliz ve Şili basınında yer alan haberlerde Arsenal'ın Barcelona'dan sezon başında 42 5 milyon avroya transfer ettiği Sanchez'in içlerinde Mesut Özil'in de olduğu takımdaki bazı oyuncularla arasında sorun yaşandığı iddia edildi Haberlerde maçlarda Mesut Özil'in takım arkadaşı Sanchez'e saha içerisinde gereken desteği vermeyerek yeterli seviyede pas atmadığı ve bu nedenle ikili arasında sorun yaşandığı belirtildi Alexis Sanchez Arsenal'de bu sezon çıktığı 15 karşılaşmada 8 gol ve 3 asistlik bir performans sergilerken sakatlığı nedeniyle bu aralar forma giyemeyen Mesut Özil ise 9 maçta 1 gol ve 2 asistle oynadı

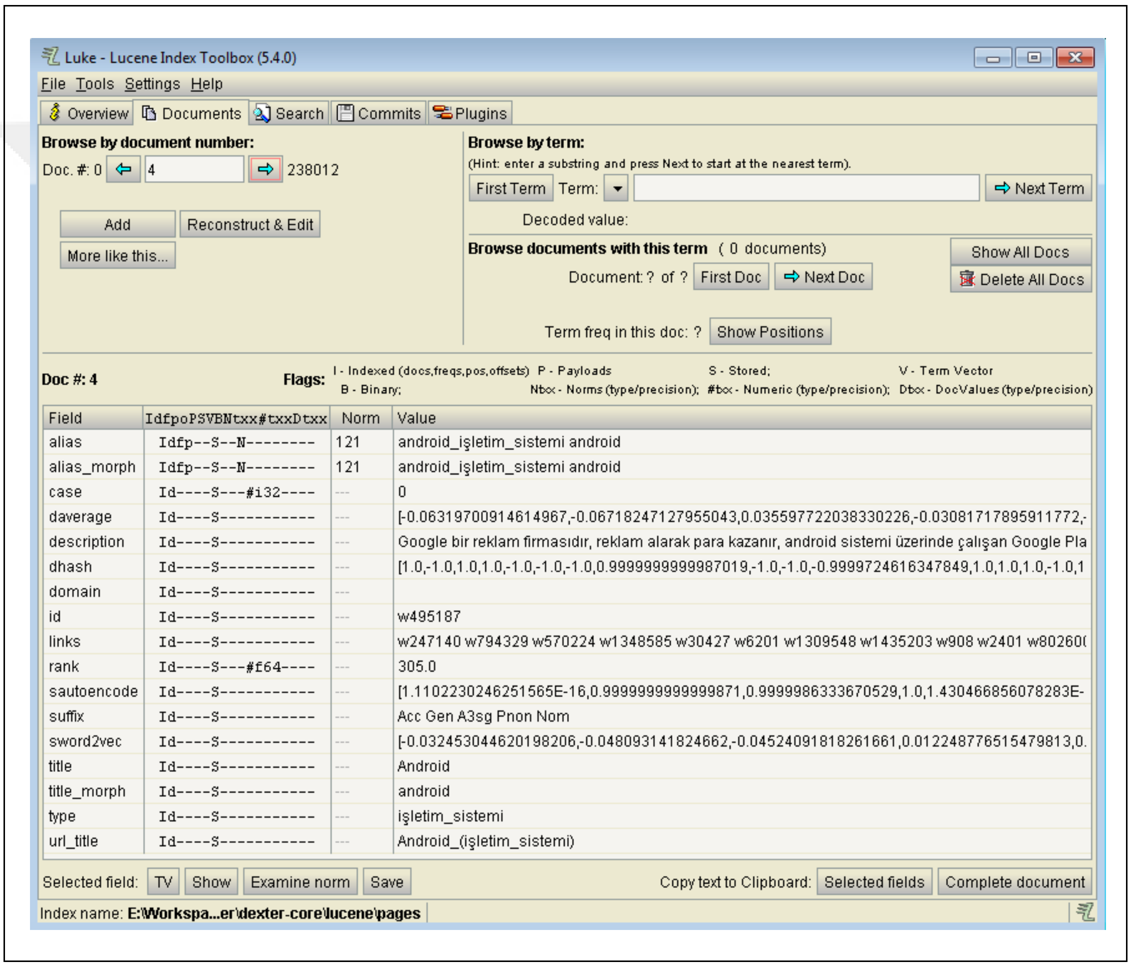
Figure 4.2. Linking result of a Turkish news article is shown through Thinker User Interface.

4.2. KNOWLEDGE BASE

The Knowledge Base module generates the knowledge base of Thinker by integrating the Turkish dictionary and Vikipedi. The Turkish dictionary covers the vast majority of Turkish concepts. On the other side, Vikipedi covers a large subset of these concepts. Hence, the knowledge base formed by combining these knowledge sources would be high quality, comprehensive, up-to-date and domain-independent. The knowledge base module takes the Turkish dictionary and Vikipedi dumps[19] in SQL and XML formats as its data

[18] http://www.trtspor.com.tr/haber/futbol/dunyadan-futbol/arsenalde-pas-krizi-83073.html
[19] https://dumps.wikimedia.org/trwiki/20150806/

sources and generates the knowledge base in Lucene[20] index format in order to improve the entity searching performance. Lucene is a powerful text search engine library, which comprises a comprehensive set of scalable, efficient and cross-platform algorithms written in Java. The index constructed in a way that every entity is represented as a Lucene document and the properties of the entity form the fields of the document. An illustration of an example Lucene document and its properties can be seen in Figure 4.3.


Luke - Lucene Index Toolbox (5.4.0)
File Tools Settings Help
Overview Documents Search Commits Plugins
Browse by document number:
Doc. #: 0 4 238012
Add Reconstruct & Edit
More like this...
Browse by term:
First Term Term: Next Term
Decoded value:
Browse documents with this term (0 documents)
Show All Docs
Delete All Docs
Document: ? of ? First Doc Next Doc
Term freq in this doc: ? Show Positions
Selected field: TV Show Examine norm Save
Copy text to Clipboard: Selected fields Complete document

(Hint: enter a substring and press Next to start at the nearest term).

Doc #: 4

Flags: I - Indexed (docs,freqs,pos,offsets) P - Payloads S - Stored; V - Term Vector B - Binary; Ntxx - Norms (type/precision); #txx - Numeric (type/precision); Dtxx - DocValues (type/precision)

Field	IdfpoPSVBNtxx#txxDtxx	Norm	Value
alias	Idfp--S--N--------	121	android_işletim_sistemi android
alias_morph	Idfp--S--N--------	121	android_işletim_sistemi android
case	Id----S---#i32----	---	0
daverage	Id----S-----------	---	[-0.06319700914614967,-0.06718247127955043,0.035597722038330226,-0.030817178959111772,-
description	Id----S-----------	---	Google bir reklam firmasıdır, reklam alarak para kazanır, android sistemi üzerinde çalışan Google Pla
dhash	Id----S-----------	---	[1.0,-1.0,1.0,1.0,-1.0,-1.0,-1.0,0.9999999999987019,-1.0,-1.0,-0.99997246163 47849,1.0,1.0,1.0,-1.0,1
domain	Id----S-----------	---	
id	Id----S-----------	---	w495187
links	Id----S-----------	---	w247140 w794329 w570224 w1348585 w30427 w6201 w1309548 w1435203 w908 w2401 w80260(
rank	Id----S---#f64----	---	305.0
sautoencode	Id----S-----------	---	[1.1102230246251565E-16,0.9999999999999871,0.9999986333670529,1.0,1.430466856078283E-
suffix	Id----S-----------	---	Acc Gen A3sg Pnon Nom
sword2vec	Id----S-----------	---	[-0.032453044620198206,-0.048093141824662,-0.04524091818261661,0.012248776515479813,0.
title	Id----S-----------	---	Android
title_morph	Id----S-----------	---	android
type	Id----S-----------	---	işletim_sistemi
url_title	Id----S-----------	---	Android_(işletim_sistemi)

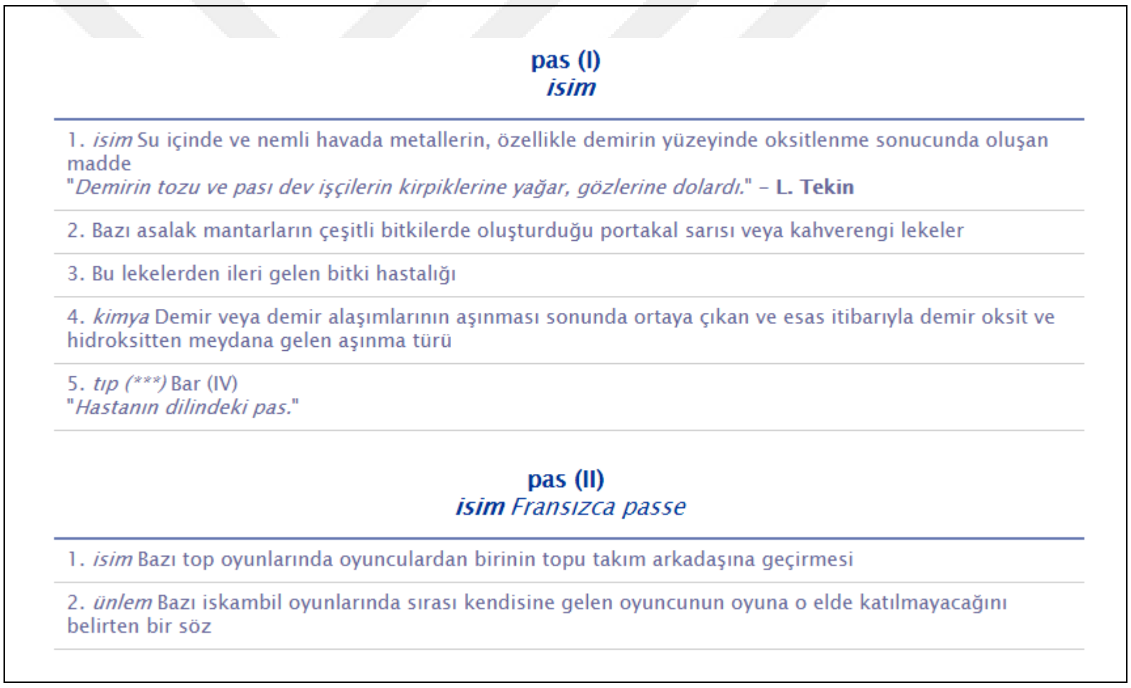
Index name: **E:\Workspa...er\dexter-core\lucene\pages**

Figure 4.3. An illustration of an example Lucene document and its properties.

The Turkish dictionary contains about 70,000 words and 110,801 fine-grained senses. For each sense, word id, title, short description (7 words in average), sense rank, part-of-speech (verb, noun, adjective, etc.) information is provided. However the dictionary only contains sample sentences for around 18,000 senses. Entities are either common nouns, which

[20] http://lucene.apache.org/

usually refer to a class of entities (person, company, city) or proper nouns, which are unique instances of certain classes such as “Barack Obama”, “Huawei” and “Istanbul”. Therefore, we filtered the Turkish dictionary and utilized only the senses, which have part-of-speech with the noun type. We also filtered sub-meanings of senses by selecting only the highest ranked coarse-grained meanings for each word. For example, "pas" word in the Turkish dictionary has two coarse-grained meanings and totally seven fine-grained meanings, which are listed in order by their rank in Figure 4.4. Besides, the highest ranked meanings which are passing in sports and rust, other five meanings are excluded due to the filtering utilized. As a result of the filtering, we obtained a dictionary with approximately 47,500 distinct words and 48,500 distinct senses.



pas (I)
isim

1. *isim* Su içinde ve nemli havada metallerin, özellikle demirin yüzeyinde oksitlenme sonucunda oluşan madde
"*Demirin tozu ve pası dev işçilerin kirpiklerine yağar, gözlerine dolardı.*" – **L. Tekin**

2. Bazı asalak mantarların çeşitli bitkilerde oluşturduğu portakal sarısı veya kahverengi lekeler

3. Bu lekelerden ileri gelen bitki hastalığı

4. *kimya* Demir veya demir alaşımlarının aşınması sonunda ortaya çıkan ve esas itibarıyla demir oksit ve hidroksitten meydana gelen aşınma türü

5. *tıp (***)* Bar (IV)
"*Hastanın dilindeki pas.*"

pas (II)
isim *Fransızca passe*

1. *isim* Bazı top oyunlarında oyunculardan birinin topu takım arkadaşına geçirmesi

2. *ünlem* Bazı iskambil oyunlarında sırası kendisine gelen oyuncunun oyuna o elde katılmayacağını belirten bir söz

Figure 4.4. Fine-grained senses of "pas" word in the Turkish dictionary.

Wikipedia is a popular and comprehensive online encyclopedia collaboratively created by volunteers. Each Wikipedia article has a unique title, which can be treated as named entities. Redirection links within an article can be considered as links to synonymous articles. Some articles contain infoboxes, which summarize the key information, such as “birth date” and “occupation of people”. Unlike in ontologies, Wikipedia articles do not have formally defined hierarchical relationships with each other. An article may be categorized in numerous ways. For example, the article on "Noam Chomsky" is categorized as 1928 births, 20th-century American writers, American linguists, Lecturers, etc. Such categories provide valuable information about the entity in the article. As mentioned previously, the Turkish Wikipedia (Vikipedi) [92] is utilized as the second knowledge source in this study. As of June 2016, Vikipedi contains approximately 274,000 articles[21]. Article titles, categories, links between pages are provided in relational tables and article content and infobox information are provided in an XML file in wiki-text format.

The Knowledge Base module queries the Turkish dictionary and Vikipedi tables and parses wiki-text content then creates a Lucene document for each entity containing its properties. The Knowledge Base module filters the disambiguation pages defined in Vikipedi during the indexing step. A disambiguation page lists references to entities that share the same name. For example, the disambiguation page for "pas"[22] lists eight associated entities having the same name "pas" including the rust and passing in football. These disambiguation pages are not used by the system, therefore these pages are filtered. Table 4.1 shows the number of entities in our final knowledge base.

Table 4.1. Turkish entity linker knowledge base characteristics.

# Total Entity	**# Vikipedi Entity**	**# Turkish Dictionary Entity**
216,550	168,050	48,500

Specifically, generated index documents have the following six attributes:

- *ID:* This field stores the unique id number assigned to each article in Vikipedi or each sense in the Turkish dictionary.
- *Title:* This field stores the title of an entity.
- *Alias:* This field stores titles of redirection links to a Vikipedi article. For example "Beşiktaş JK (Beşiktaş Gymnastics Club)" article has 21 redirection links such as

[21] https://tr.wikipedia.org/wiki/Özel:İstatistikler
[22] https://tr.wikipedia.org/wiki/Pas

"BJK, Besiktas, Besiktas Jimnastik Kulubu, etc". This field is empty for the entities extracted from the Turkish dictionary.

- *Links:* This field stores ids of outgoing links from a Vikipedi article. This field is empty for the entities extracted from the Turkish dictionary.
- *Type*: This field stores the infobox type or the phrase given in between parentheses in the title for Vikipedi entities. For example Java programming language article has the title Java_(programming_language). This way, we can extract the type value of this article as programming language. This field is empty for the entities extracted from the Turkish dictionary.
- *Description:* This field stores the entity description. It is directly provided for the Turkish dictionary. For the Vikipedi articles, first sentence of the article is used as the entity description.

4.3. ENTITY LINKING PIPELINE

The final module of Thinker is the Entity Linking Pipeline. It is the core module that links the entity mention in the input text with the knowledge base. Figure 4.5 shows flow chart of the Entity Linking Pipeline. This module is composed of three sub-modules: Spotter, Entity Tagger and Entity Ranker. In the following sections, these sub-modules will be analyzed comprehensively.

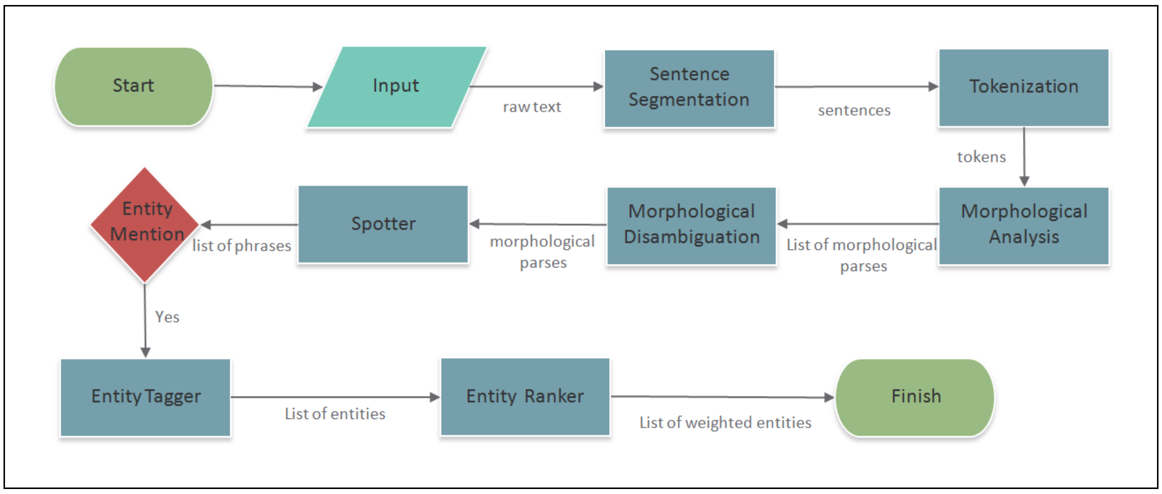


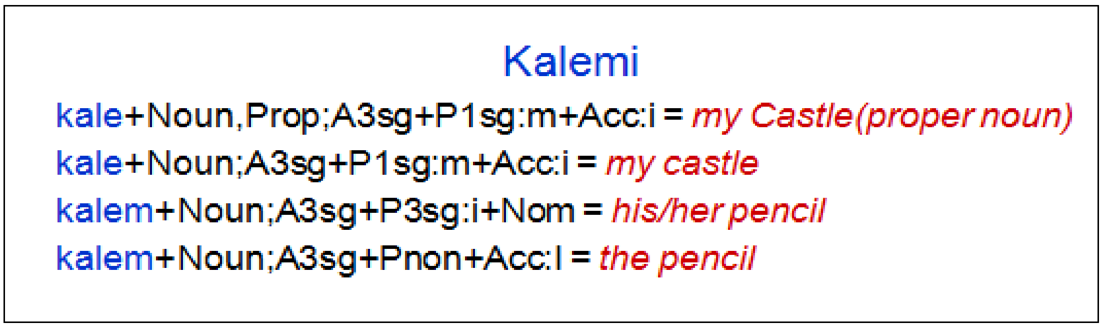
Figure 4.5. The Entity Linking Pipeline flow chart.

4.3.1. Spotter (Entity Detector)

The Spotter produces a list of possible spots (entity mentions) in a given document. Spot refers to small fragments of text, which may correspond to an entity in the knowledge base. The spotter produces all possible n-grams of terms, where n ranges from one to six in this study. Then spots are associated with a list of entity candidates (if any) by querying the knowledge base. Specifically, this task consists of; (i) sentence detection where input text content are split into sentences; (ii) morphological analysis and disambiguation where each word is analyzed to find its root, suffixes and part-of-speech tag; (iii) determining the n-grams where groups of successive words are identified; (iv) querying the knowledge base with these n-grams and finding candidate entities.

Sentence detection, morphological analysis and disambiguation are functions provided in Zemberek [1] NLP system. Zemberek is a popular open source NLP library for Turkish. Zemberek provides the most common NLP tasks, such as sentence detection, tokenization, morphological analysis and morphological disambiguation. The details of Zemberek NLP system are introduced in Section 2.1.

Sentence detection function simply splits the text into sentences. Morphological analysis function identifies morphemes and other linguistic units, such as root words, affixes, part-of-speech of the given input word. The morphological parser may output more than one possible analysis for a word due to ambiguity. For example, the parser returns four analyses for the Turkish word "kalemi" as shown in Figure 4.6. Zemberek also provides morphological disambiguation functionality to handle the morphological ambiguity.



```
Kalemi
kale+Noun,Prop;A3sg+P1sg:m+Acc:i = my Castle(proper noun)
kale+Noun;A3sg+P1sg:m+Acc:i = my castle
kalem+Noun;A3sg+P3sg:i+Nom = his/her pencil
kalem+Noun;A3sg+Pnon+Acc:I = the pencil
```

Figure 4.6. Morphological analysis of Turkish word "kalemi" using Zemberek NLP library.

We utilize the root word that is given by the morphological disambiguation operation in order to query the knowledge base. As noted above, N-gram function generates all possible n-grams of the terms, where *n* ranges from one to six in this study. An n-gram is a contiguous sequence of *n* items from a given sequence of text. The items can be phonemes, syllables, letters, words or base pairs according to the application. In this study, the items are words, n-grams may also be called shingles [93]. For example, Table 4.2 shows the generated shingles for a given example sentence.

Table 4.2. An illustration for the n-gram function of the spotter module.

Text	**Candidate Spots**
Arsenal'de pas krizi	arsenal
	arsenal pas
	arsenal pas kriz
	pas
	pas kriz
	kriz

The N-gram approach is computationally expensive compared to the common approach for entity detection, which is the extraction of noun phrases by using an NLP system. However, Turkish language has free word order, hence noun phrase detection performance of Turkish NLP systems are not satisfactory. Therefore, we preferred to apply the N-gram approach for entity detection.

4.3.2. Entity Tagger

The Entity Tagger is the key challenging component which accepts a list of spot matches produced by the Spotter and selects the best entity match for each spot by performing disambiguation whenever a spot has more than one candidate meaning. Disambiguation is achieved by extending previous methods [5,8,55,71,94]; (i) leveraging knowledge-based methods with context dependent and independent features [55], (ii) leveraging supervised methods with entity type information [94], and (iii) leveraging entity features to learn neural entity embeddings [5,8]. Using these methods, semantic relatedness between mention-entity pairs are computed and highest scored entity for each spot is selected. Specifically, each mention-entity pair is scored by weighting and combining score values with the formula below:

$$\begin{aligned} tagger(m,e) = {} & Wnss \times nss(m,e) + Wlcs \times lcs(e) \\ & + Wss \times ss(e) + Wts \times ts(e) + Wtcs \times tcs(e) \\ & + Wtc \times tc(e) + Wdw2vs \times dw2vs(e) + Wsdw2vs \times sdw2vs(e) \\ & + Wlw2vs \times lw2vs(e) + Wmes \times mes(e) + Wdes \times des(e) \\ & + Wls \times ls(e) + Wlesk \times lesk(e) + Wslesk \times slesk(e) \end{aligned} \tag{4.1}$$

Tagger is the function that returns the semantic relatedness value between a mention *m* given by the spotter and an entity *e* in the knowledge base. Each *W* in the equation denotes the weight associated with the corresponding metric and these weight values are set by using NSGA-II (Non-dominated Sorting Genetic Algorithm II) [95]. Name String Similarity (*nss*) and Letter Case Similarity (*lcs*) form the context independent and Suffix Similarity (*ss*), Type Similarity (*ts*), Type Content Similarity (*tcs*), Type Classifier Similarity (*tc*), Description Word2Vec Similarity (*dw*2*vs*), Simple Description Word2Vec Similarity (*sdw*2*vs*), Link Word2Vec Similarity (*lw*2*vs*), Metadata Embedding Similarity (*mes*), Description Embedding Similarity (*des*), Link Similarity (*ls*), Lesk (*lesk*) and Simplified Lesk (*slesk*) forms the context dependent similarity metrics that are used for entity disambiguation and scoring purposes, in this study.

Only a subset of the similarity metrics (*nss*, *lcs*, *ss*, *dw*2*vs*, *sdw*2*vs*, *des*, *lesk* and *slesk*) proposed above can be calculated for the Turkish dictionary entities because of lack of information (i.e. type, link and metadata). For the other similarity metrics, the Turkish dictionary entities are scored as zero. We formulated the tagger function in this way in order to favor the Vikipedi entities, since they provide richer information compared to the Turkish dictionary entities. However, this does not prevent the selection of Turkish dictionary. We have observed that the Turkish dictionary entities could also be linked by the system, when they are relevant to the given context in contrast to the Vikipedi entities.

All of the similarity metric scores are normalized between zero and one by dividing the scores with the highest scored entity given by the corresponding similarity metric. In the following sub-sections, these similarity metrics are introduced.

4.3.2.1. Name String Similarity

String similarity between detected spot and candidate entity title is the most direct feature that can be used for selecting the right entity for that spot [90]. We use Levenshtein distance for calculating the string similarity between a spot m and an entity e. The Levenshtein distance between two words is the minimum number of single character edits (i.e. insertions, deletions or substitutions) required to change one word into the other. Levenshtein distance is normalized with the length of the entity title. Since Turkish is an agglutinative language, most of the time entity mention and entity 64 name differ and necessity for *nss* occurs.

$$nss(m, e) = 1 - levenshtein(m, e) \tag{4.2}$$

4.3.2.2. Letter Case Similarity

Letter Case Similarity takes into account letter case match between detected spot and the corresponding entity. When cases of the detected spot and entity match, the entity gets letter case similarity score one.

There are three cases which are upper, lower and proper. If entity title occurs in all upper case in its description, then it is considered as upper case entity. If entity title occurs in all lower case, then it is considered as lower case entity. Otherwise the entity is considered as proper case entity.

4.3.2.3. Suffix Similarity

Suffix Similarity takes into account suffix match between detected spot and the description of the entity. Morphological analysis is done on the entity description and the list of suffixes are determined. Similarly, morphological analysis is carried out on the input text and the suffixes of the candidate entity mention are also identified. For each suffix match between the list of suffixes extracted from entity description and the input text, entity gets additional suffix similarity score one.

4.3.2.4. Type Similarity

Type Similarity takes into account the amount of common entity types (i.e. film, artist, language) in the context where the entity mention appears. The formula is defined as follows:

$$TS(e_j) = \sum_{i=1, i \neq j}^{n} \frac{type(e_j, e_i)}{n - 1} \qquad (4.3)$$

where *type*(e_j, e_i) returns 1, if types of entity e_j and e_i are same. N is again the number of candidate entities in the context. With this formula, entities sharing the same type in the input text, would have high scores and the entity having the most frequent type value would be favored.

4.3.2.5. Type Content Similarity

Type Content Similarity (*tcs*) checks whether the type value of the entity occurs in the context where the entity mention appears. If this is the case, the entity gets *tcs* score one, otherwise it is scored as zero. For example, for the given sentence "The film Titanic is directed by James Cameron", the mention Titanic would have two candidate entities; Titanic movie and Titanic Ship. In this case, the film entity would be favored; since the Titanic movie entity has a defined type value film and this entity exists separately in the given context.

4.3.2.6. Type Classifier Similarity

Type Classifier Similarity is used to predict semantic typing of entities for a given sentence. When predicted type of detected spot matches with the candidate entity type, the entity gets type classifier score one. The type classifier is realized in three steps: selection of the semantic types that will be utilized, creation of training data and development of a fast and accurate multi-class classification algorithm.

The first step in entity type classifier is defining the set of semantic types. There is no hierarchical relationship among Vikipedi types. Therefore, to achieve a high quality classifier, entity types are manually analyzed by human experts. They are sorted according to the number of Vikipedi entities, which are annotated with them. Most frequently occurring two hundred types are manually annotated with higher level types. Figure 4.7 shows the type taxonomy and sample types. We have defined 10 first level types such as person, organization, creative work, etc. and 200 second level types such as bird, kingdom, book, etc..

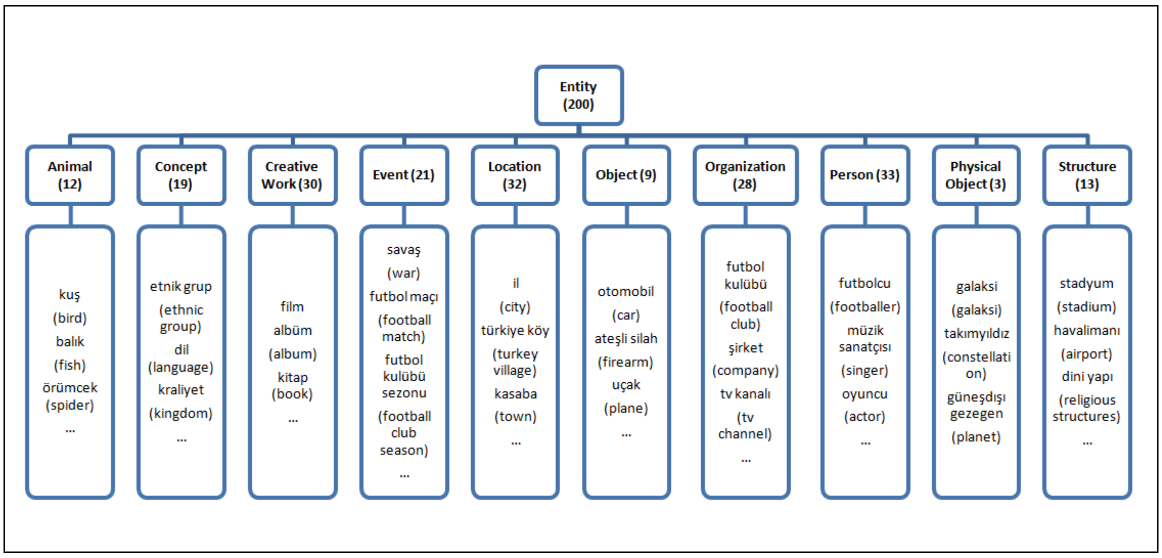
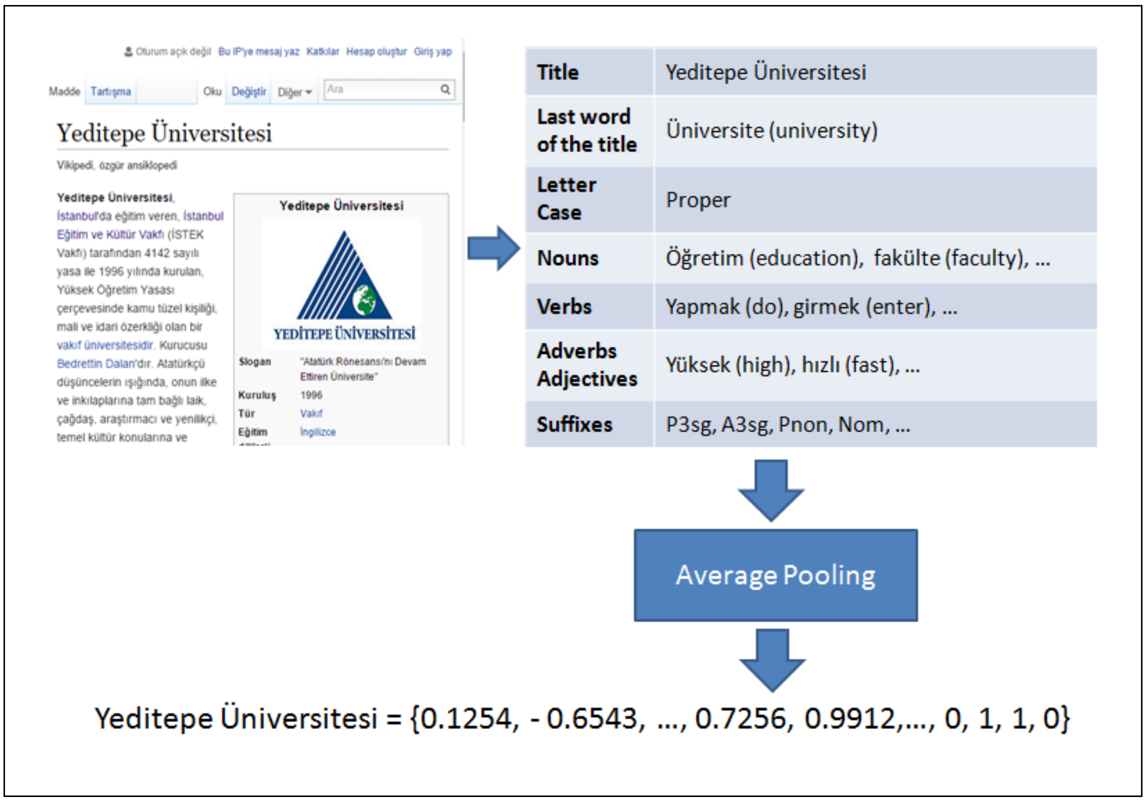

Entity (200)
Animal (12)
Concept (19)
Creative Work (30)
Event (21)
Location (32)
Object (9)
Organization (28)
Person (33)
Physical Object (3)
Structure (13)
kuş (bird) balık (fish) örümcek (spider) ...
etnik grup (ethnic group) dil (language) kraliyet (kingdom) ...
film albüm (album) kitap (book) ...
savaş (war) futbol maçı (football match) futbol kulübü sezonu (football club season) ...
il (city) türkiye köy (turkey village) kasaba (town) ...
otomobil (car) ateşli silah (firearm) uçak (plane) ...
futbol kulübü (football club) şirket (company) tv kanalı (tv channel) ...
futbolcu (footballer) müzik sanatçısı (singer) oyuncu (actor) ...
galaksi (galaksi) takımyıldız (constellation) güneşdışı gezegen (planet)
stadyum (stadium) havalimanı (airport) dini yapı (religious structures) ...

Figure 4.7. The Type Classifier taxonomy used in this study with sample types.

The second step, creating a training set for these tags, is achieved by utilizing content of Vikipedi articles, which are annotated with a type in our tag set. Eligible articles' contents are given to the Feature Extractor function and a list of vectors is created for each article (entity). Feature Extractor first detects the list of sentences in which the title of the entity exists. Then, for each sentence a feature vector is created that can be used for training. Feature Extractor extracts linguistic features (part-of-speech tags and suffixes) and contextual features by utilizing morphological analysis and morphological disambiguation functionalities of Zemberek.

Extracted part-of-speech tags are used to categorize and filter contextual information (surrounding words) of the entity mentions. Contextual words besides nouns, verbs, adverbs and adjectives are filtered since they are not discriminating features. As a result of this step, a set of words and suffixes are created for each entity mention. We also use letter case (upper, lower or proper) as a feature. The table in Figure 4.8 shows the generated features for a sentence of given Vikipedi article. Then, extracted feature sets (title, letter case, nouns, verbs, adverbs, adjectives and suffixes) are used for creating vectors that represent the entity mention. In order to achieve this goal, Word2Vec word embedding [5] and average pooling [96] algorithms are used.



Yeditepe Üniversitesi

Vikipedi, özgür ansiklopedi

Yeditepe Üniversitesi, İstanbul'da eğitim veren, İstanbul Eğitim ve Kültür Vakfı (İSTEK Vakfı) tarafından 4142 sayılı yasa ile 1996 yılında kurulan, Yüksek Öğretim Yasası çerçevesinde kamu tüzel kişiliği, mali ve idari özerkliği olan bir vakıf üniversitesidir. Kurucusu Bedrettin Dalan'dır. Atatürkçü düşüncelerin ışığında, onun ilke ve inkılaplarına tam bağlı laik, çağdaş, araştırmacı ve yenilikçi, temel kültür konularına ve

Yeditepe Üniversitesi

Slogan	"Atatürk Rönesansı'nı Devam Ettiren Üniversite"
Kuruluş	1996
Tür	Vakıf
Eğitim	İngilizce

Title	Yeditepe Üniversitesi
Last word of the title	Üniversite (university)
Letter Case	Proper
Nouns	Öğretim (education), fakülte (faculty), ...
Verbs	Yapmak (do), girmek (enter), ...
Adverbs Adjectives	Yüksek (high), hızlı (fast), ...
Suffixes	P3sg, A3sg, Pnon, Nom, ...

Yeditepe Üniversitesi = {0.1254, - 0.6543, ..., 0.7256, 0.9912,..., 0, 1, 1, 0}

Figure 4.8. An illustration of feature vector extraction for the Vikipedi entity "Yeditepe University".

Word2Vec word embedding algorithm is used to construct vector representations of words. Word2Vec was run using the skip-gram model with a window size of seven, hierarchical softmax training, and a frequent word sub-sampling threshold of 0.001 to create 100 dimensional vectors. We have used Milliyet corpus [97] as input data in order to cover as many words as possible. The corpus contains 408,305 documents; they are the news articles and columns of five years, 2001 to 2005, collected from the Turkish newspaper Milliyet[23]. The input data is also lemmatized to find root form of Turkish words before training. Essentially by lemmatization we make the input space denser and we prevent learning different vectors for inflectional forms of words. As a result, we constructed 100 length vectors for the words in the input corpus. After constructing word vectors, the extracted feature set for a candidate entity is converted into a set of vectors by using the corresponding word vectors obtained by Word2Vec algorithm. As a result, the feature set contains list of word vectors $W_{ei} = w_1, w_2, ..., w_n$ for each category which are: title, nouns, verbs, adverbs and adjectives.

Certainly, the number of features extracted for an entity would differ based on the number of words that exist in the context (Vikipedi article content) where this entity appears. The average pooling algorithm is applied in order to convert the set of word vectors into a fixed length vector that can be processed to classify entities by their types. Specifically, it is applied for each extracted feature sets (title, letter case, nouns, verbs, adverbs, adjectives and suffixes). The algorithm simply takes the average of the word vectors and computes a fixed length vector. The average pooling can be computed as in the following formula where c_i denotes the word vector of the n-th element in feature set i:

$$v_i = \frac{1}{N} \times \sum_{n=1}^{N} c_i, n \qquad (4.4)$$

[23] www.milliyet.com.tr

In contrast to other features, letter case and suffixes are not composed of full words. In order to represent suffixes in a vector format, bag of words method is utilized. As a result of this method, suffixes could be represented with a 64 length vector. For the letter case, we have represented lower case with zero, upper case with one and proper case with two. To derive the final entity mention vector, all feature vectors are unified as shown in Figure 4.8. The length of unified vector is 565.

In the final step, we use this labeled data in order to train linear classifier models for the entity type classifier. As linear classifiers, we build SVM, Logistic Regression and Softmax classifiers. SVM and Logistic Regression classifiers are realized using the software Liblinear [98] and Softmax classifier is realized using the software Encog [99]. We have set two parameters of Liblinear: cost of constraints violation = 2.0 and stopping criteria = 0.05. We trained 11 classifiers in total. One for predicting the first layer (e.g., person, animal, etc.), the other ten classifiers are used for predicting the final type of the input entity mention.

4.3.2.7. Description Word2Vec Similarity

Description Word2Vec Similarity (*dw*2*vs*) compares the possible descriptions of an ambiguous mention with the non-ambiguous mentions in the same context. The descriptions of entities are directly provided for the Turkish dictionary. For the Vikipedi articles, first sentence of the article is used as the entity description. The context is determined by sliding locality windows of a certain width. A candidate entity is compared with its surrounding entities in the same locality window. Firstly, as described in Type Classifier Similarity, description of entities are converted into vectors. The length of the vector is 300. First 100 dimensions are the averages of the nouns, the second 100 dimensions are the averages of the verbs and last 100 dimensions are the averages of the adverbs and adjectives in the same window. After the feature vector generation, the similarity between candidate entity and the surrounding non ambiguous entities are calculated by summing the cosine similarities of the generated vectors.

4.3.2.8. *Simple Description Word2Vec Similarity*

Simple Description Word2Vec Similarity (*sdw*2*vs*) compares the possible descriptions of an ambiguous mention with the input text. Like *dw*2*vs*, description of an entity and input text are converted into vectors. After the feature vector generation, the similarity between entity description and input text is calculated by measuring the cosine similarity of the generated vectors.

4.3.2.9. *Link Word2Vec Similarity*

Link Word2Vec Similarity (*lw*2*vs*) measures the link similarity of entities using the Word2Vec word embedding algorithm. Link vectors of entities are created by using the link information defined in the Vikipedi dump. For this, Continuous Bag-of-Words (CBOW) model proposed by Mikolov et al. [5] is utilized. The CBOW model is similar to the feed forward neural network language model, where the non-linear hidden layer is removed and the projection layer is shared for all words. Figure 4.9 shows the CBOW network model for our setting. Id values of entities that are linked to an entity are given as input to the network and id value of the entity is expected to be seen as output. Hence, the utilized framework is like a bi-gram model. As the dimension of output vectors increases, the quality of the resulting vectors increases as well as the complexity. For our task, the dimension is set to 150 and the model is trained with 11*M* entity pairs (Vikipedi links). After the generation of link vectors, the link similarity between candidate entity and the surrounding non ambiguous entities are calculated by summing the cosine similarities of the generated link vectors.

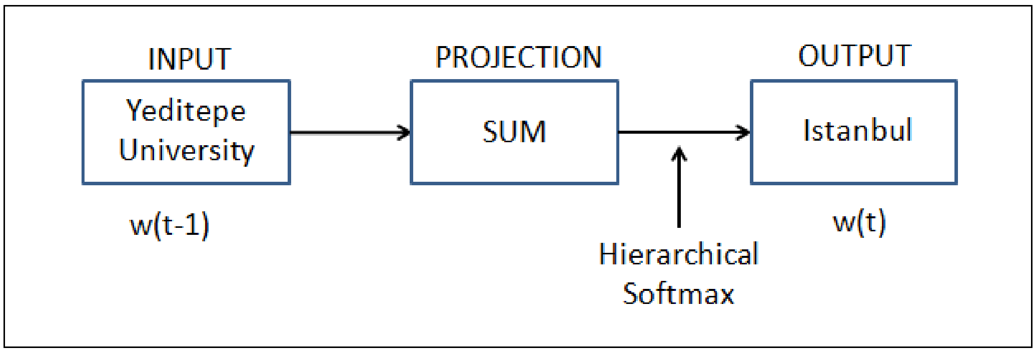
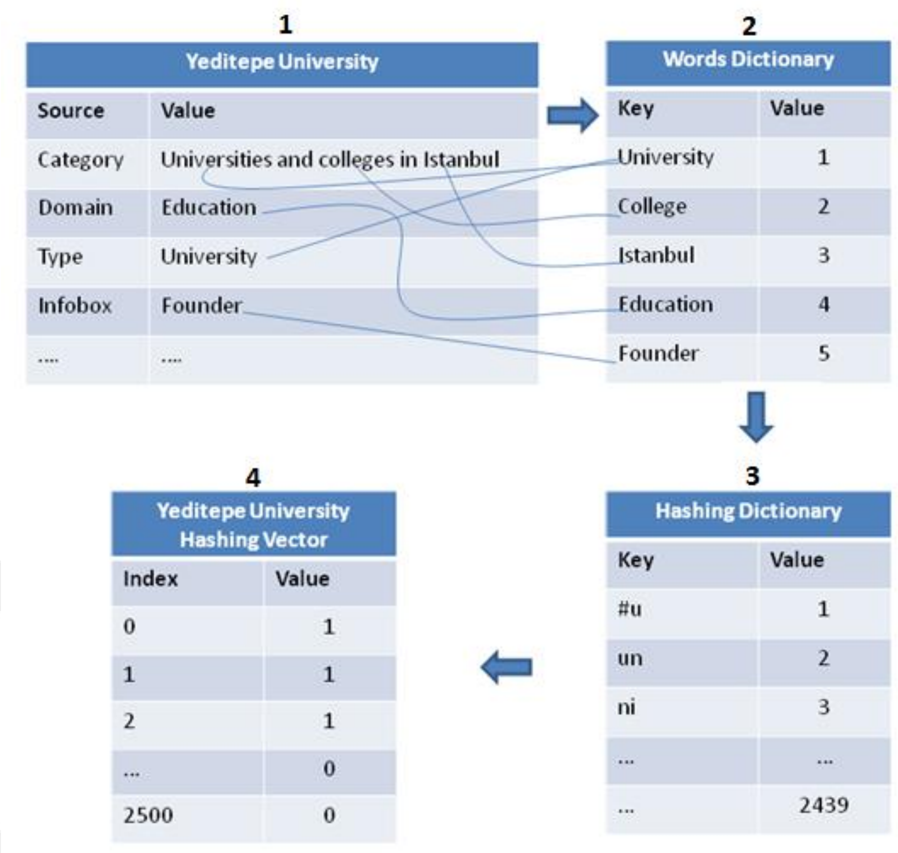


Figure 4.9. Simple CBOW model with only one word in the context.

4.3.2.10. Metadata Embedding Similarity

Metadata Embedding Similarity (*mes*) aims to measure the semantic similarity between entities. In order to achieve this goal, firstly the metadata (category, type, infobox) about Vikipedi entities is encoded in a vector format using bag of words method and afterwards a hashing technique is applied to reduce the dimensionality of vectors. Then, dense and continuous-valued semantic vector representations are created from the hashing vectors by using autoencoders. Finally, *mes* score of an ambiguous entity is calculated by summing the cosine similarity of its metadata vector representation with the non ambiguous entities' metadata vectors in the same context.

Figure 4.10 shows the illustration of our proposed entity hashing method. The aim is to encode the metadata that exists in the Vikipedi pages in a vector format. The first table in the figure lists the metadata for the entity "Yeditepe University". For all entities in Vikipedi, the string values of these metadata entries are firstly split into lower case words and the bag of words term vectors are generated (second table). After this step, hashing algorithm is applied on each word using a similar approach developed by Heck et al. [8]. Hashing helps to reduce the dimensionality of the bag of words vectors. As shown in the third table, for each word(e.g., university), start and end marks are added to the word (e.g., #university#). Then, the word is split into letter $2-grams$ (e.g., #u, un,...,y#) and the total word is represented as a vector of these letter $2-grams$. Finally, the union of the vectors (bitwise or operation) created for all of the words that exist in the metadata entries is formed. This final vector is the hashing vector of the corresponding entity (fourth table).



Yeditepe University	
Source	Value
Category	Universities and colleges in Istanbul
Domain	Education
Type	University
Infobox	Founder
....	....

Words Dictionary	
Key	Value
University	1
College	2
Istanbul	3
Education	4
Founder	5

Hashing Dictionary	
Key	Value
#u	1
un	2
ni	3
...	...
...	2439

Yeditepe University Hashing Vector	
Index	Value
0	1
1	1
2	1
...	0
2500	0

Figure 4.10. Illustration of metadata hashing with letter two-grams.

As shown in Table 4.3, by using bag of words vector representation, each Vikipedi entity could be represented by a 170*K vector*. Whereas by using *2 – grams*, *3 – grams* and *4 - grams* word hashing, each Vikipedi entity could be represented by a 2.5K, 23K and 132K vectors respectively, instead of a 170K vector. The Vikipedi corpus consists of 312 distinct symbols, including the digits, alphabetical letters together with a large number of various derivatives of Latin letters, such as letters with diacritics (e.g., “ç”, “ö”, “ü”, “ä”), and various symbols (e.g., “%”, “$”, “&”, “#”). Certainly, not all possible n-grams of these symbols exist in the corpus. For instance the number of different bigrams is 2.5K as noted above.

One potential problem of hashing approach is the collision of different words or entities that might have the same letter n-gram vector representation. 821 word and 61 entity collisions have been detected (e.g., words “sicili / silici” and entities “tatlı / tatlım”), when the 2 – grams hashing vectors are generated for the 168K Vikipedi entities defined in the knowledge base.

In order to reduce the collision rate, multiple length hashing approach has also been utilized. First, histogram of unique *n – grams* is calculated for the Vikipedi corpus. Then, most frequent *K* unique *n-grams* are represented with a distinct dimension in the generated vectors and the rest are split into *1 – grams* and they are represented by using *n* dimensions. For example, assume that "mur" *3 – gram* is not frequently occurring in the corpus, then it would be represented with "m", "u" and "r" *1 – grams*. By using this approach, we created vector representation of Vikipedi entities with varying *n – grams (2 and 3+1)* and vector sizes (2.5K, 5K and 10K). We also considered to use $3 - grams$ word hashing, which results $23K$ vectors. However, the computational complexity of the training process with the autoencoders increases exponentially with such huge vectors and it was not possible to couldn't train entity vectors with our hardware recourses. Then we evaluated performances of the generated entity vectors with the entity linking experiments and observed that *n-gram* length and collision rate do not have noticeable effect on accuracy of the system. Since satisfactory results are obtained with $2{-}grams$ word hashing, $3 - grams$ and (3+1) – grams vectors are not utilized in the experiments.

Table 4.3. Hashing statistics of Vikipedi articles' metadata information.

Hashing	**Vector Size**	**Number of Word Collision**	**Number of Entity Collison**
Bag of words	171,556	0	N/A
2 - grams	2,439	821	61
3 - grams	23,251	N/A	N/A
4 - grams	132,067	N/A	N/A
3+1 - grams	5,000	654	136
3+1 - grams	10,000	221	14
3+1 - grams	15,000	149	12
4+1 - grams	10,000	9,895	2,389

Figure 4.11 presents the architecture used for learning the entity embeddings. An autoencoder is a type of a neural network that is trained to reconstruct (decoding) its inputs. It is possible to obtain compressed and distributed representation (encoding) of the inputs with the training process in an autoencoder. The details of autoencoders are introduced in Section 2.2. The proposed autoencoder model has two encoding and decoding layers. The first layer is trained on the hashing vectors by encoding the input to 600 length vectors and then decoding these encoded vectors back to input hashing vectors. Similarly, the second layer gets as input the encoded 600 length vectors of the first layer and it is trained by encoding the input data to 300 length vectors. As a result, a 300 length compressed and distributed representation of an entity is obtained. The embeddings learned by using autoencoders carry category, type and infobox information of Vikipedi entities.

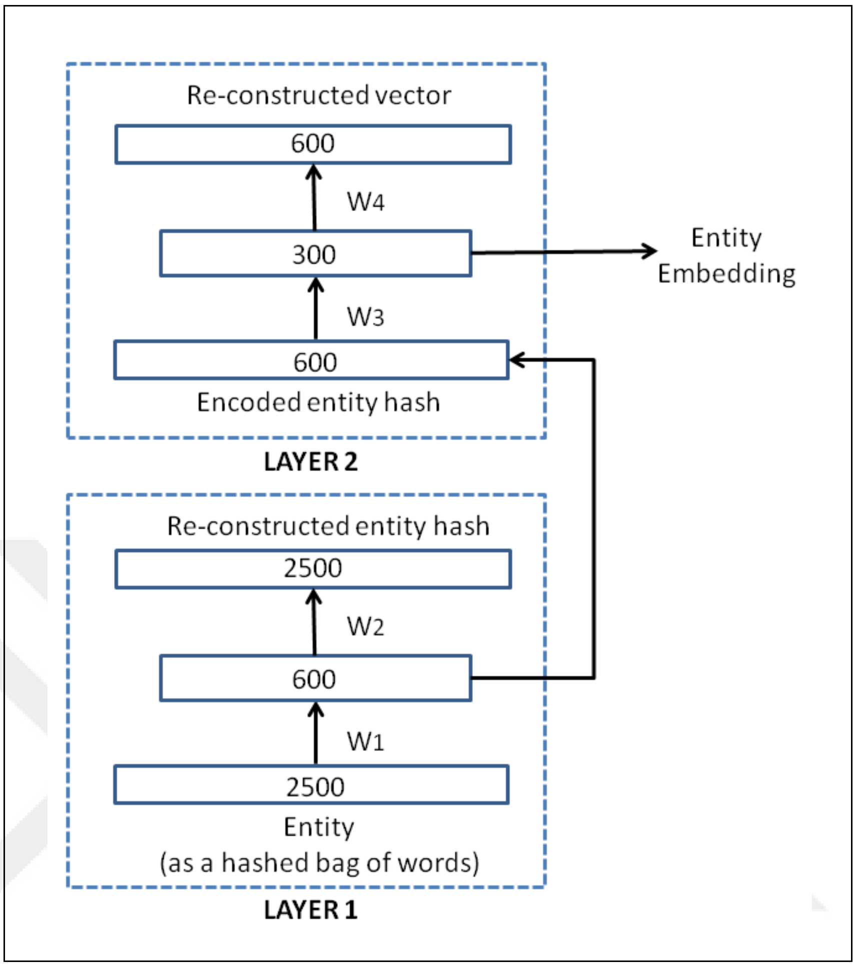


Figure 4.11. The Autoencoder architecture for learning embeddings from metadata.

The generated embedding vectors can be used for measuring entity similarities. Table 4.4 presents three Vikipedi entities and their ten closest neighbors by cosine similarity. The vectors utilized are the union of metadata and link embeddings. The cosine similarity values are also given in the table. The first words in columns one and two are the corresponding entities of ambiguous word "pas", which are passing in football and rust. We can observe that all of the 1st column entities are related with football and second one is related with chemistry. This shows that our proposed embedding learning algorithm distributes entities well. Finally, all of the entities in column three are universities in Turkey. Certainly, they are also semantically similar entities.

Table 4.4. Three Vikipedi entities and their ten closest neighbors by cosine similarity of their union of metadata and link embeddings.

Pas (futbol) / Passing	**Pas (kimya) / Rust**	**Yeditepe Üniv. / University**
Pas (futbol) / Passing = 1.00	Pas (kimya) / Rust = 1.00	Yeditepe Üniversitesi / University = 1.00
Averaj / Goal difference = 0.84	Ksenon tetraflorür / Xenon tetrafluoride = 0.88	Mersin Üniversitesi / University = 0.88
Jübile maçı / Testimonial match = 0.83	Ksenik asit / Xenic acid = 0.88	Kadir Has Üniversitesi / University = 0.88
Taraftar / Fan = 0.83	Ferrosen / Ferrocene = 0.88	Koç Üniversitesi / University = 0.88
Yaw Preko = 0.83	Klorit / Chlorite = 0.88	Sakarya Üniversitesi / University =0.88
Lesly Malouda = 0.82	Sülfit / Sulfite = 0.87	Cumhuriyet Üniversitesi / University = 0.87
Samuel Johnson = 0.82	Hidroflorik asit / Hydrofluoric acid = 0.86	Erzincan Üniversitesi / University = 0.87
Augustine Ahinful = 0.82	Sulfamik asit / Sulfamic acid = 0.86	Akdeniz Üniversitesi / University = 0.87
1958-59 İzmir P. Ligi / Professional League = 0.82	Demir oksit / Iron oxide = 0.86	Ege Üniversitesi / University = 0.87
Fernand Coulibaly = 0.82	Hidrojen sülfür / Hydrogen sulfide = 0.86	Trakya Üniversitesi / University = 0.87

The proposed Metadata Embedding Similarity is one of the contributions of this study. It differs in several directions from the method proposed in [8]. In this study, the hashing is applied only on the entity title and bag of words representation of its properties. The combination of these two vectors form the representation for an entity. In contrast, hashing is applied on both the entity properties (e.g., founder) and all related entity titles (e.g., Istanbul) in our study. Also, the whole Vikipedi data including Vikipedi categories are utilized. The generated Turkish entity embeddings are publicly accessible at the website[24].

[24] http://cse.yeditepe.edu.tr/ARTI/Projeler/embedding.htm

4.3.2.11. Description Embedding Similarity

Description Embedding Similarity (*des*) compares the possible descriptions of an ambiguous entity with the non ambiguous entities in the same context. Likewise *mes*, description of entities are converted into vectors again by using hashing and autoencoder algorithms. After the feature vector generation, *des* score of an entity is calculated by summing the cosine similarities of the surrounding non ambiguous entities' vectors.

4.3.2.12. Link Similarity

Previous studies for English denote that a link based similarity measure between entities is very effective for determining the coherence between entities [90]. A link based similarity is utilized in this study, too. The similarity function takes into account the amount of common outgoing links that exist in entities' content pages. Jaccard metric is utilized to calculate the similarity of an entity *ej* with a group of other entities as defined in the following formula:

$$LS(e_j) = \sum_{i=1, i \neq j}^{n} \frac{outgoingLinks(e_j) \cap outgoingLinks(e_i)}{outgoingLinks(e_j) \cup outgoingLinks(e_i)} \tag{4.5}$$

The link similarities are calculated only between an entity and the other non ambiguous entities in the same context. The context is determined again by sliding locality windows of a certain width. A candidate entity is compared with its surrounding entities in the same locality window. Certainly *n* is the number of surrounding entities in the formula.

4.3.2.13. Lesk and Simplified Lesk

Lesk [71] algorithm compares the possible descriptions of an ambiguous entity with the non ambiguous entities' definitions in the same context. It simply counts the amount of common words between the descriptions of entities. Simplified Lesk [72] algorithm compares the description of an ambiguous entity with the terms contained in the input text.

Thinker also utilizes Lesk and Simplified Lesk algorithms as similarity metrics. As in previous similarity metrics, the Lesk and Simplified Lesk similarity scores are normalized between zero and one.

4.3.3. Entity Ranker

Thinker uses two statistical features and a disambiguation score to rank and weigh the significance of linked entities. The utilized statistical features are Term Frequency (*tf*) and Web Popularity (wp). Tf feature favors entities with higher frequencies in a given context and wp feature favors entities with higher frequencies in a global context. In this study, we use entity frequencies in a collection of news articles published by one of the popular Turkish online newspaper HaberTurk[25] for calculating the global popularity of entities. The formula of wp is defined as follows:

$$wp(e) = \frac{\log_{10} hits(e) + 1}{\log_{10} M} \tag{4.6}$$

where *hits*(e) is the number of HaberTurk hits for the entity e, M is the total number of news articles published by HaberTurk. By weighting and combining score values of these two features, Thinker weighs the significance of an entity by using the below formula:

$$ranker(e) = W_{tagger} \times tagger(e) + W_{tf} \times tf(e) + W_{wp} \times wp(e) \tag{4.7}$$

4.4. ENTITY DISCOVERY

The success of an entity linking system clearly depends on the term coverage of the knowledge base utilized. In order to effectively process domain-independent documents a comprehensive, up-to-date, and evolving knowledge base is required. Wikipedia is one of

[25] http://www.haberturk.com/

the popular knowledge bases that satisfies these requirements and therefore it is widely used in entity linking systems. However, the long tail of entities is not popular enough to have their own Wikipedia articles. For example, "Yetenek Sizsiniz Türkiye" (Turkish version of the Got Talent series) television show exists in Vikipedi and it is typed as television show, however "İbo Show", another famous Turkish television show, is not defined as a Vikipedi article. In this study, an entity discovery system is also proposed for semi automatically detecting entity mentions that are not defined in Vikipedi. What is more, the system can discover the semantic typing of detected unlinkable entity mentions, too. For informative knowledge, we aim to type new entities in a fine-grained manner (e.g., basketball player, economist, airport, as opposed to generic types like person, organization, event) [74].

There are two main challenges for an entity discovery system: the detection of candidate entities and predicting their semantic types. We address the first challenge by an ngram based approach that detects frequent noun phrases in a given corpus as candidate entities. The second challenge is addressed with a fine-grained entity recognizer.

As shown in Figure 4.12, Turkish Entity Discovery system has been realized through the design and implementation of three major modules: Candidate Entity Detector, Feature Extractor and Fine-grained Entity Recognizer.

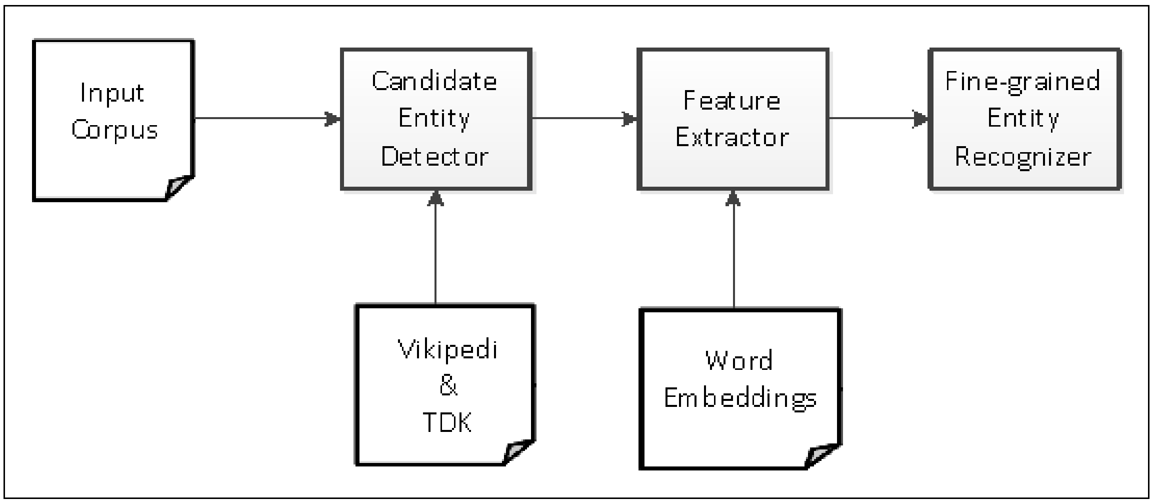


Figure 4.12. Architecture of Turkish Entity Discovery system.

The Candidate Entity Detector module produces a list of possible entity mentions for a given corpus. Entity mention refers to small fragments of text, which may correspond to an entity in a given knowledge base. This task is achieved in two steps. In the first step, Candidate Entity Detector produces all possible n-grams (where n can be between one and three) of successive nouns in a sentence. This task consists of (i) sentence detection where each document in the corpus are split into sentences; (ii) lemmatization and parts of speech detection where each word is analyzed to find its root and part-of-speech tag; and (iii) finding noun phrases where groups of successive nouns are identified. The idea behind this approach is that, multi-word entities, especially special names, location names etc. are usually consist of noun phrases. In the second step, Candidate Entity Detector identifies frequently occurring noun phrases as candidate entities and filters ones that are already defined in utilized knowledge base and dictionary. In this study, all phrases that occur in either Vikipedi or the Turkish Dictionary are filtered out since they are already known entities for Turkish.

Feature Extractor module takes the list of entity mentions and the list of sentences where the entities occur from Candidate Entity Detector module and produces an entity vector for each entity mention. The details of the Feature Extractor function is described in Section 4.3.2.6. Then, Fine-grained Entity Recognizer is used for detecting semantic typing of entities. We utilized the type classifier described in section 4.3.2.6 as fine-grained entity recognizer. The type classifier is a two level linear classifier. The first level has 10 distinct types such as person, organization, creative work, etc. and the second level has 200 distinct types such as bird, kingdom, book, etc..

5. EVALUATIONS AND EXPERIMENTS

5.1. VIDEOLIZATION EXPERIMENT AND USER STUDY

This section presents evaluations of the Videolization system. The visual interpretation of analyzed Web Content is converted into video by using video generation application of Huawei Turkey R&D Center. We evaluate the overall system from two different aspects; (i) we have assessed visual quality and effectiveness through a qualitative user study and (ii) we have measured performance and run-time characteristics through simulated experiment. Firstly, we will introduce our experimental setup.

5.1.1. Experimental Setup

Visualization of Web content has a subjective characteristics, that is why we have preferred to evaluate the system by a user study. Thus, the effectiveness of the proposed system is validated empirically over an opinion survey (questionnaire). A survey with eight questions is prepared for this purpose. Then we requested from 30 participants (19 male, 11 female engineers) to use our system and respond the survey questions. The participants' age average was 29.38 with a standard deviation of 2.83. The participants were given a basic training to familiarize them with the system and a sample scenario using the Web UI was illustrated to each participant. Then each participant were asked to use the system to generate a video on a topic selected by the participant. After watching the video, they were asked to evaluate the system by answering the given questionnaire.

The first six questions assess specific features of the system such as quality of video effects and TTS. The participants were asked to make a judgment for each question using a scale between one to four (four being the best). The latter two questions are yes/no questions utilized to assess the impression the system left on the participants.

5.1.2. Experiment and User Study

Our results from this study are visualized in Figure 5.1 and Figure 5.2. Quality wise the best performing aspect was "Audio-Visual synchronization" with $\mu = 3.40$, $std = 0.61$. For all the questions the performance was above average. The least performers were "background music" ($\mu = 2.40$, $std = 0.76$) and "visual quality" (animations, transition, etc...) ($\mu = 2.57$, $std = 0.62$) which are open to improvement.

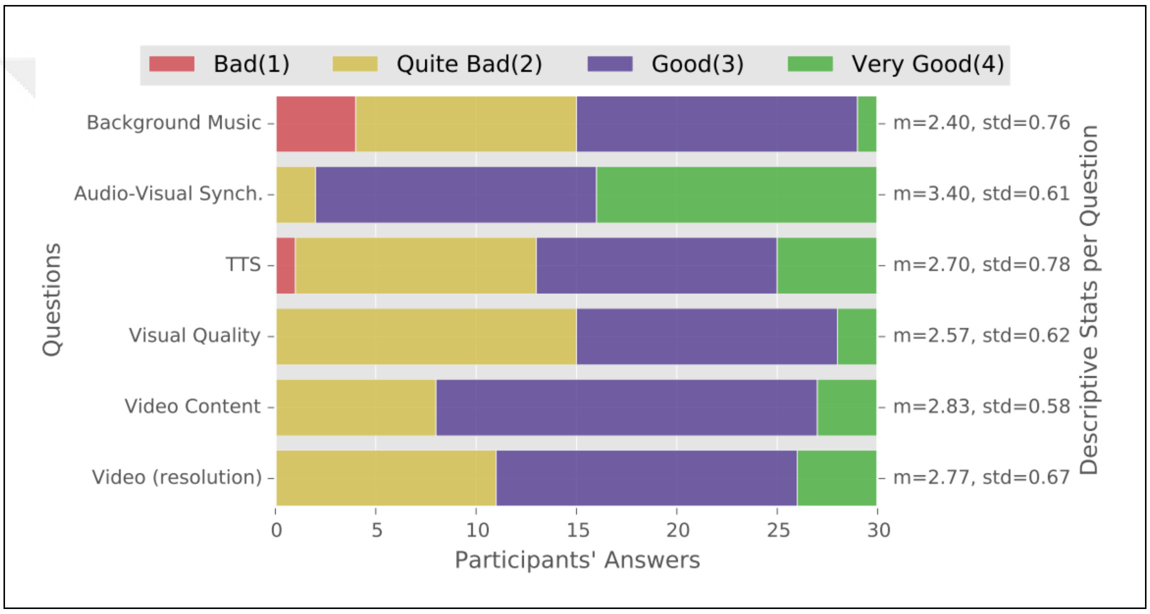


Figure 5.1. Opinion survey results for the first six questions are shown with a stacked histogram. Mean and standard deviation values per question are also reported. For these questions the participants were asked to evaluate the quality of the respective aspect with values from one to four.

The answers given to the last two questions show that 84% of participants indicate they found generated videos enjoyable and 70% of them would like to use our system to consume Web content on their TVs. These last two questions evaluate the system in terms of general performance.

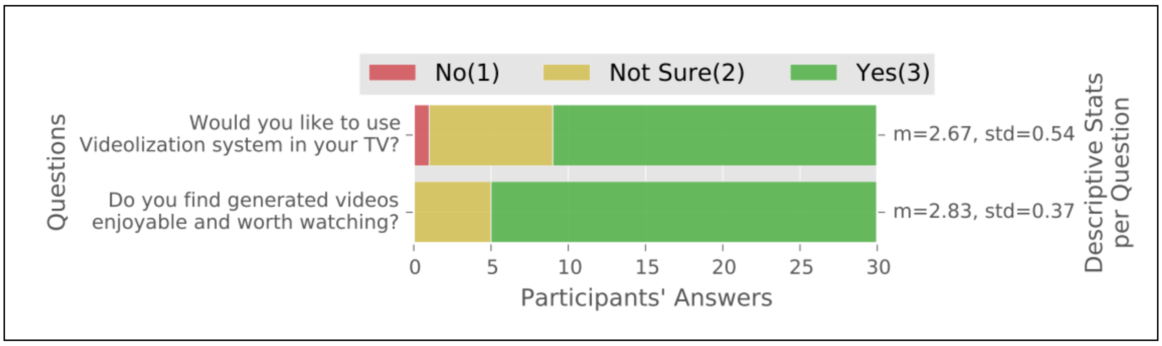


Figure 5.2. Opinion survey results for the last two questions are shown with a stacked histogram. Mean and standard deviation values per question are also reported. For these questions the participants were asked yes/no questions related to the general effectiveness and appeal of the Videolization system.

5.1.3. Runtime Efficiency

Runtime efficiency is another key performance indicator of video generation systems. The amount of Web data increases exponentially and a video generation system must be scalable to process the increasing amount of data. In order to measure runtime efficiency and scalability of our system, we experimented our system by creating videos for the most popular Wikipedia articles[26]. The list of Wikipedia articles are provided in Table A.1. The experiments resulted in 100 videos that were generated in a workstation with Intel Xeon E5 CPU @ 2.40 GHz and 16 GB of RAM. Figure 5.3 shows the performance results.

[26] https://goo.gl/c759nv

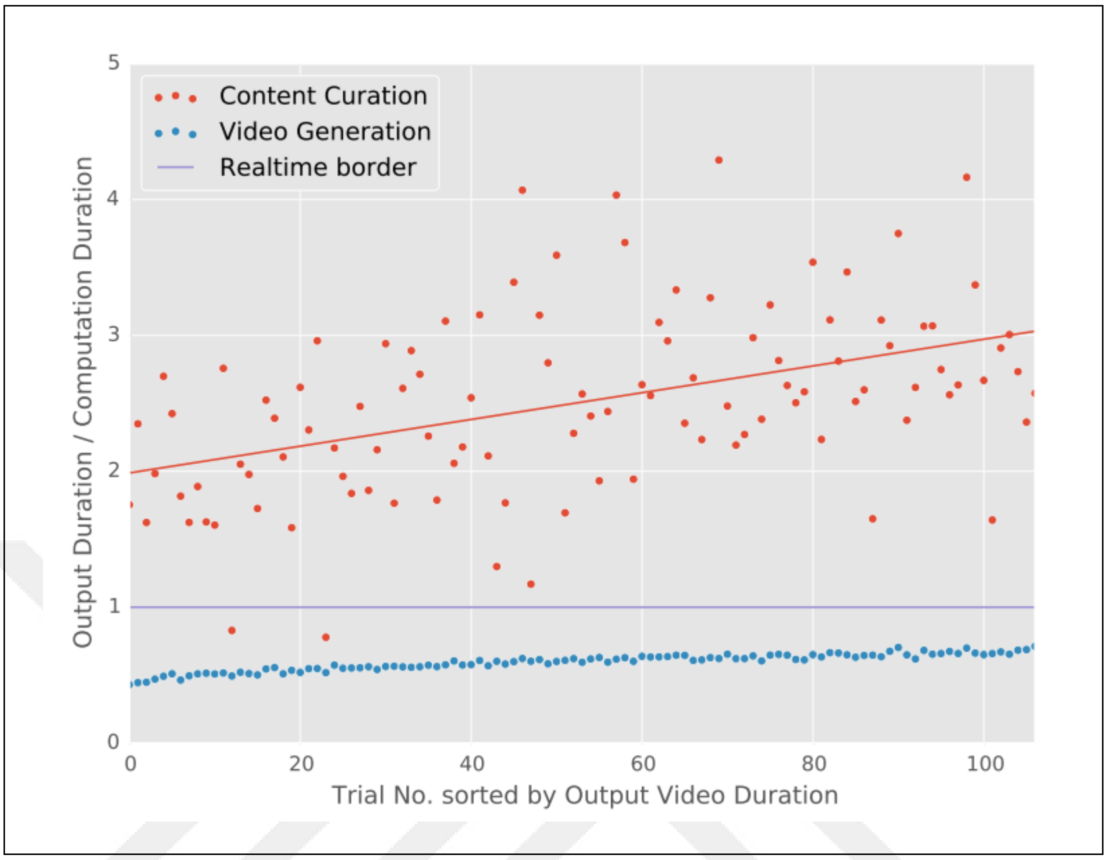


Figure 5.3. Content Curation and Video Generation phases are compared with respect to output video duration. 1.0 value denotes real-time computation, any value lower than this is slower than real-time and vice versa. Trial Values are sorted with respect to output video duration.

We observed that on the average, one minute video could be generated in approximately 140 seconds, which shows that our system can generate videos from Wikipedia articles almost in real-time. To further investigate our runtime efficiency we have analyzed the time consumption of our main modules, namely Content Curation and Video Generation. Although our Content Curation module is faster than realtime, the video generation task is around %70 slower. Moreover, as can be seen in Figure 5.3, the performance of video generation has a linear relation with respect to output video length. The longer the output video, the more closer to real-time video generation becomes. Content Curation has a less direct relation to output video length as it depends on the type and amount of content that is acquired. However a basic trend line analysis shows that in general Content Curation has a similar relation to video length as well.

We also analyzed scene type distributions in the 100 generated videos. Figure 5.4 shows the numbers per each scene type. We observed that most of the scenes are entity based. A small portion of the scenes are text based. Although we prioritize videos over images, there are ten times more Entity Image representations compared to Entity Video representation. We attribute this result to the lack of large enough video repository.

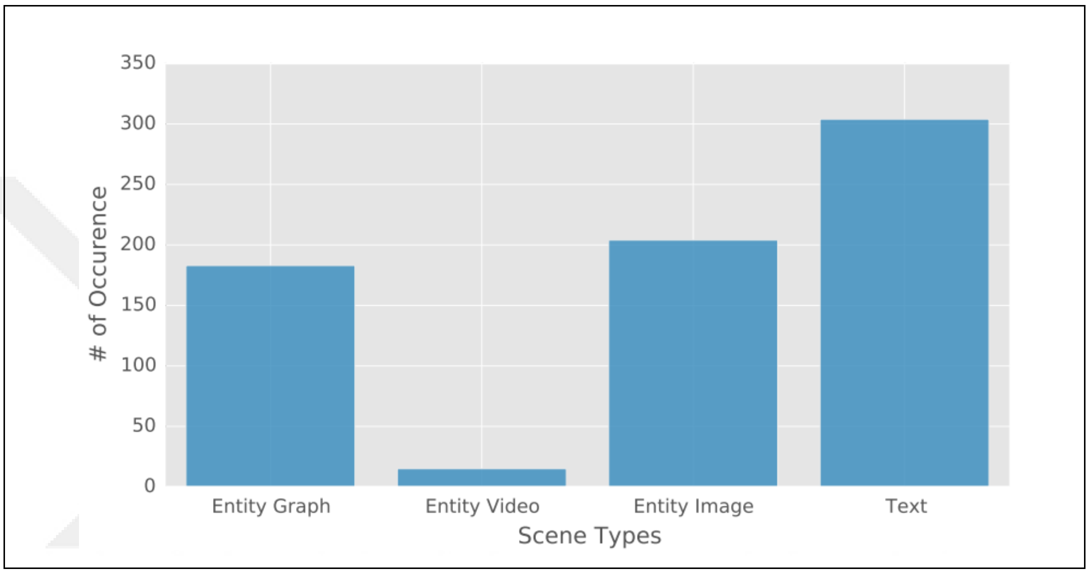


Figure 5.4. Visualization technique counts for the experiment are shown.

Samples videos produced by the Videolization system are publicly available[27].

5.1.4. Discussion

In our experiments we have shown that Videolization system is capable of providing videos in near real-time. Applying a distributed or parallel processing approach could easily improve the system to real-time performance as well. The Videolization system was designed to allow such improvements, since each sentence can be processed in parallel with minimal changes to the scene type rules. Hence, inter-sentence dependency could easily be removed. Then each of the videolized sentence could be stitched together to output the final video.

[27] https://goo.gl/K18mw9

Video and image selection process utilized to represent a given entity is also open to improvement. Currently we optimize this process by using the entity text, type and document title in the search query in order to acquire a related image or video. An image retrieval based approach [100] over an annotated video or image library can provide more suitable visual representations. An image retrieval approach can also be utilized to judge the quality of selected images and videos.

In the Videolization work-flow only one entity presentation is allowed per scene. This design choice was made to represent the most salient information clearly. However, in some cases allowing multiple entities can be desired. For instance consider the following sentence:

[... It has 21 R&D institutes in countries including China, the United States, Canada, the United Kingdom, Pakistan, France, Belgium, Germany, Colombia, Sweden, Ireland, India, Russia, and Turkey ...].

None of the entities in this sentence should have more weight compared to others. An additional scene type that allows multiple entities to be represented can be added to the system. For instance country flags could be used for this example. Quantized information can be found in many sentences with numeric values like ratios, counts and monetary values. Although these values are not technically entities, they are suitable for TV friendly visualization through specialized charts, scales and clip art style graphics. An NLP based approach [101] that can convert for instance the following sentence;

[... it currently serves 45 of the world's 50 largest telecoms ...]

into a an animated pie chart style visualization requires further investigation.

In the current Videolization system the Entity Graph is always preferred over other scene types. This is a design choice to maximize the amount of information presented to the user. Depending on the application, it is possible to come up with other orderings between scene types. Another approach to scene type selection is to utilize a global optimization that can maximize the amount of entities visualized and that can minimize the amount of Text representation types.

5.2. THINKER EVALUATIONS AND EXPERIMENTS

This section presents comparative evaluations of our entity linking and discovery algorithms. We analyze Thinker from three different aspects; (i) evaluate entity linking through a manually annotated news data set, (ii) measure the linking performance and runtime characteristics through a simulated experiment and (iii) evaluate entity discovery through a news corpus. Firstly, we will introduce our evaluation metrics.

5.2.1. Evaluation Metrics

The performances of the entity linking and discovery algorithms are calculated using the widely used measures Precision, Recall and F-measure.

Precision gives information about the correctness of given answers by the system. Precision is defined as follows:

$$Precision = \frac{\#\, correct\ answers\ provided}{\#\, answer\ provided} \tag{5.1}$$

Recall is another important measure that gives information about the coverage and the performance of a system. Recall is defined as follows:

$$Recall = \frac{\#\, correct\ answers\ provided}{\#\, all\ mappings} \tag{5.2}$$

F-measure is helpful to evaluate systems that have coverage lower than 100%. A system can achieve 100% precision without answering any queries. To asses overall performance of a system, F-measure is widely used. F-measure values can be calculated giving varying weights to precision and recall. The traditional F-measure (F_1 score) which gives equal weighs to precision and recall (harmonic mean) is used in this study. F_1 score is defined as follows:

$$F_1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (5.3)$$

5.2.2. Entity Linking Experimental Setup

To evaluate disambiguation performance of the proposed entity linking algorithm, a Turkish news data set is created with ambiguous Turkish phrases. Firstly, ambiguous phrases are detected and categorized by their types (e.g., film, singer, newspaper) in the data set. Then, 50 phrases with the highest frequency (number of incoming links to an article) in Vikipedi are selected manually for the experiments to cover most of the popular ambiguous phrases. Then distinct meanings of each phrase is determined. The meaning of a phrase could be defined both in Vikipedi and the Turkish dictionary. In such cases, the news articles are annotated with both of the entities. As a result, we have identified 112 different entities for the 50 ambiguous phrases. Table 5.1 shows the lists of the senses that exist in the experiment set, which are defined in the Turkish dictionary, Vikipedi or in both of the sources. Thirdly, for each meaning of the ambiguous phrases, three news articles are collected from Hurriyet[28] online news paper and the experimental disambiguation data set is formed. Finally, the data set is split into two as validation and test sets. One of the three articles is used as validation set in order to tune the parameters of Thinker and the other two articles are used as the test set to evaluate the final disambiguation performance. A sample from the news articles data set is provided in Table B.1.

[28] http://www.hurriyet.com.tr/

Table 5.1. Hurriyet news disambiguation data set characteristics.

Turkish Dictionary (11)
Akrep (hour hand), Al (red), Gözcü (watchman), Hortum (hose), Mali (financial), Pike (quilt), Pike (nose dive), Soket (socks), Tekir(fish), Ton (tone), Yaş (wet)
Vikipedi (50)
Alabanda (ancient city), Ateş (newspaper), Ayna (film), Ayna (music band), Ayna (TV show), Babil (film), Babil (newspaper), Babil (city), Bar (ballet), Ceylan (singer), Ceza (singer), Duvar (film), Fare (computer), Gözcü (newspaper), Gözcü (airplane), Güneş (newspaper), Havale (money order), Havale (illness), Hortum (proboscis), Hortum (tornado), Kafes (Ottoman), Kafes (game), Kale (chess), Kale (sport), Kanun (science), Kemer (arch), Klavye (computer), Mali (country), Milliyet (newspaper), Mısır (country), Nike (mythology), Nike (company), Nota (diplomatic note), Ordu (city), Pascal (unit), Pascal (programming language), Pas (illness), Penguen (magazine), Penguen (character), Petrol (song), Pul (flake), Sunucu (server), Tango (software), Tekir (cat), Tekir (software), Top (software), Türk (mechanical Turk), Ülker (company), Yumurta (film), Zaman (newspaper)
Both (51)
Akrep (scorpion), Akrep (horoscope), Alabanda (marine), Ateş (fire), Ateş (fever), Ayna (mirror), Bar (unit), Bar (folk), Bar (pub), Çay (stream), Çay (tea), Ceylan (gazelle), Ceza (penalty), Dil (language), Dil (tongue), Duvar (wall), Fare (mouse), Far (eyeshadow), Far (headlight), Güneş (sun), Kafes (cage), Kale (castle), Kanun (law), Kanun (music), Kemer (belt), Kivi (kiwi), Kivi (bird), Klavye (music), Lüks (unit), Lüks (luxury), Milliyet (nationality), Mısır (corn), Nota (music), Ordu (army), Pas (pass), Pas (rust), Penguen (penguin), Petrol (oil), Pul (stamp), Somun (nut), Somun ekmek (bread), Sunucu (presenter), Tango (dance), Ton (fish), Ton (tonne), Top (ball), Türk (Turk), Ülker (Pleiades), Yaş (age), Yumurta (egg), Zaman (time)

In order to evaluate general performance of the Entity Linking system, another Turkish news data set is created by collecting 20 news articles from latest news section[29] of Hurriyet online news paper. We collected five news articles from four different categories, which are Turkey, world, economy and sports. Then each article is automatically annotated with Thinker. Afterwards, each mapping is evaluated manually whether it is a correct

[29] http://www.hurriyet.com.tr/son-dakika/

mapping or not. If an entity is not detected by the system, it is marked as an undetected entity mention.

On the other side, in order to measure runtime efficiency and scalability of Thinker, we also experimented the efficiency of the system with randomly selected 500 news articles from the Milliyet corpus [97]. The selected news articles contain between 30 to 2000 words and the average document size is 400 words. The performance experiments were carried out on a laptop with Intel Core $i7 - 3537U$ CPU @ 2.50 GHz and 8 GB of RAM.

5.2.3. Entity Linking Results and Discussion

Thinker depends on several parameters therefore before evaluating the system performance, the system parameters are tuned on the evaluation data set. Firstly, hashing methodology of Metadata Embedding algorithm is determined. Table 5.2 shows the performance values for these experiments. The highest performance (47.71% F_1 score) is observed with the *2 - grams* 2,500 length and *3+1 - grams* 5,000 length vectors. We also observed that there is no significant disambiguation performance effect of hash vector sizes. However, length of vectors would effect the runtime performance. Therefore, the shortest length vectors which are generated with *2 - grams* hashing is used for further experiments.

Table 5.2. Comparison performance values of the Metadata Embedding algorithm with varying hash vector sizes on the validation data set.

Hashing Methodology	**Vector Size**	**Precision (%)**	**Recall (%)**	**F_1 (%)**
2 - grams	2,500	49.06	46.43	47.71
3+1 - grams	5,000	49.06	46.43	47.71
3+1 - grams	10,000	48.15	46.43	47.27

Then, the weight values of the similarity metrics and minimum confidence values are determined. The minimum confidence value is a threshold value that a candidate entity must have in order to be linked by the system. There are totally 15 parameters that need to be optimized. We have used NSGA-II in order to solve this parameter optimization problem by using the software MOEA Framework[30]. Although NSGA-II is widely used for multiobjective optimization problems, it is possible to run the software with a single objective, too. The objective function used in our experiments is the F_1 score of the system on the validation set. We used default parameters of the framework such as population size 100, crossover rate 0.1 and mutation rate 1/*N*, where *N* is 15 that is the number of decision variables for our problem. Then, the algorithm is run by using 20 randomly chosen seeds with 1.800 iterations. Figure 5.5 shows the change of the maximum F_1 score and the iteration number in the best run.

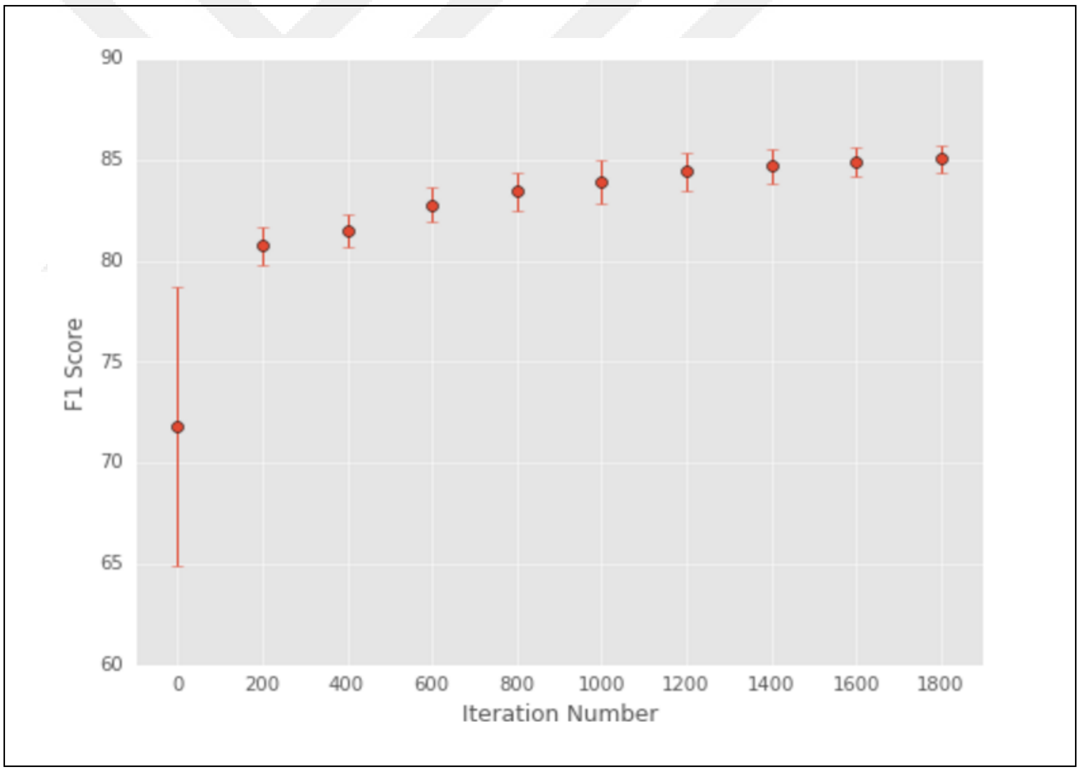


Figure 5.5. Improvement of the F_1 score during the parameter optimization of Thinker by using the NSGA-II algorithm.

We also evaluated each similarity metric individually and their combination with equal weights. Table 5.3 shows the performance values for these experiments. The highest

[30] http://moeaframework.org/

performance (85.71% F_1 score) is observed with the weights determined by using the NSGA-II. The second highest performance is achieved with the Simplified Lesk algorithm (73.73% F_1 score) and the third is when all the metrics have equal weights (71.17% F_1 score).

Table 5.3. Comparison disambiguation performance values of the entity linking algorithms on the validation data set.

Algorithm	**Precision (%)**	**Recall (%)**	**F_1 (%)**
Thinker	85.71	85.71	85.71
Simplified Lesk	76.19	71.43	73.73
All One	71.82	70.54	71.17
Link	75.00	66.96	70.75
Lesk	66.67	64.29	65.45
Link Word2Vec	58.65	54.46	56.48
Description Word2Vec	54.55	53.57	54.05
Description Embedding	53.64	52.68	53.15
Simple Description Word2Vec	51.82	50.89	51.35
Metadata Embedding	49.06	46.43	47.71
Type Content	93.75	26.79	41.67
Type	63.27	27.68	38.51
Letter Case	65.79	22.32	33.33
Suffix	72.00	16.07	26.28
Name String	83.33	8.93	16.13
Type Classifier	50.00	7.14	12.50

Precision values are also important indicators for individual performances of the similarity metrics. We observed that some of the similarity metrics (e.g., type content, suffix) performed well in terms of precision but performed poor in terms of F_1 score. Precision and F_1 scores differ because; (i) system does not link any entity to a mention when there is more than one highest scored entity; (ii) when an entity is scored lower than the minimum confidence value again it is not linked to the entity mention. The threshold is zero for the individual metrics and 1.35 for the whole system Thinker, which is determined using the NSGA-II with the optimum similarity weight values.

Table 5.4 presents the weight values obtained by using the NSGA-II. Description, word2vec and name string similarity metrics have the highest weights. We observed that most of the similarity features contributes to asses candidate entities; except suffix, type classifier and simple description word2vec metrics. The suffix similarity has low weight because most of the target entities in the experiment set probably do not have a suffix. The type classifier got lower weight because its precision is low and most of the entities (all of the Turkish dictionary) do not have predefined type values. Simple description word2vec also got lower weight since description word2vec, its modified version, performed better compared to this metric. On the other side, these weights are the output of a single GA run. There exists a number of alternative solutions that has similar performance with the solution provided above and it has been observed that even though the general picture does not change considerably, ranking of the weights might vary slightly in different solutions.

Table 5.4. Weights of the Turkish entity linker algorithms, which are determined by using the NSGA-II.

Description Word2Vec	0.99
Name String	0.98
Letter Case	0.82
Type	0.8
Link	0.73
Type Content	0.71
Simple Lesk	0.63
Link Word2Vec	0.56
Metadata Embedding	0.29
Description Embedding	0.28
Lesk	0.15
Suffix	0.04
Type Classifier	0.01
Simple Description Word2Vec	0

Thinker is also evaluated based on the width of the sliding locality windows. In the experiments, different sizes from 5 to 15 have been tested. Figure 5.6 shows the experiment results. Best result is obtained with the window size of 10 on the validation data set.

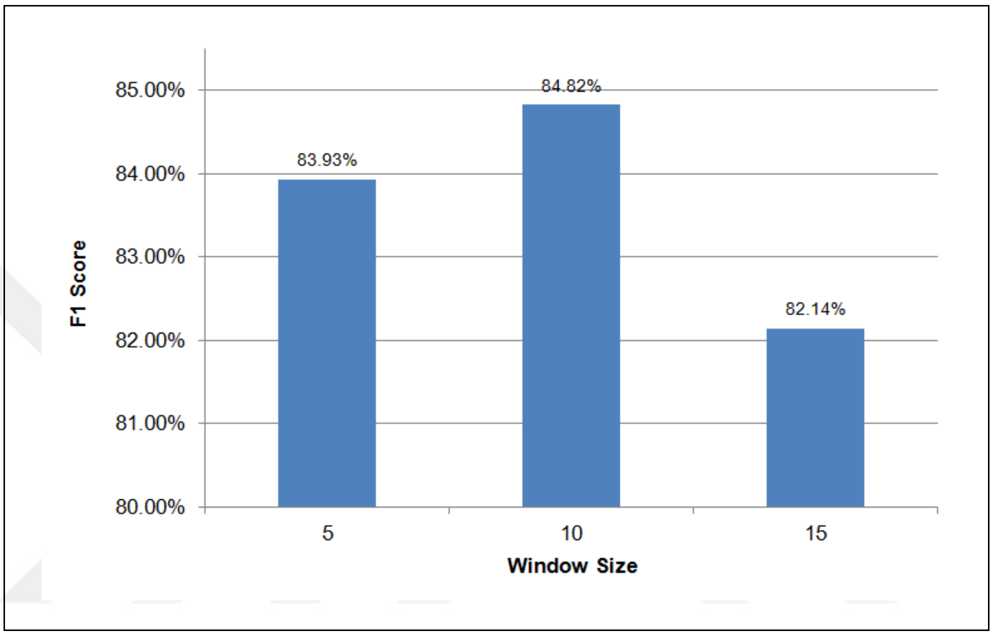


Figure 5.6. Evaluation of the entity linking algorithm with varying window sizes.

After tuning the parameters of Thinker, we used the testing data set to evaluate the disambiguation performance and perform comparative evaluation. Since, there is no Turkish entity linking system proposed in the literature, we compared Thinker with the Simplified Lesk algorithm and also individual algorithms utilized in Thinker. Table 5.5 shows the performance values on this data set. The experiment results show that our algorithm outperforms the Link and Simplified Lesk algorithms with approximately %4 and 5% better F_1 score, respectively. We observed 68.98% F_1 score for the Simplified Lesk algorithm. It is a very close to the result with the score (68.57%) reported by Mert et al. [22] using the Simplified Lesk algorithm for the Turkish WSD problem. We also observed that performance of Thinker is improved approximately 5% by using the NSGA-II compared to using equal weights (all one).

Table 5.5. Comparison disambiguation performance values of the entity linking algorithms on the test data set.

Algorithm	**Precision (%)**	**Recall (%)**	**F_1 (%)**
Thinker	73.99	73.66	73.83
Link	73.53	66.96	70.09
Simplified Lesk	71.63	66.52	68.98
All One	68.92	68.30	68.61
Link Word2Vec	62.20	58.04	60.05
Description Embedding	58.10	54.46	56.22
Lesk	59.69	52.23	55.71
Description Word2Vec	51.80	51.34	51.57
Type Content	95.00	33.93	50.00
Simple Description Word2Vec	47.66	45.54	46.58
Metadata Embedding	47.17	44.64	45.87
Type	59.32	31.25	40.94
Letter Case	68.54	27.23	38.98
Type Classifier	57.14	17.86	27.21
Suffix	60.00	10.71	18.18
Name String	80.00	5.36	10.04

The general entity linking performance of Thinker is evaluated manually by annotating the 20 news articles. Table 5.6 shows number of correct, wrong and undetected mappings for the each news article. Table 5.7 shows the general entity linking performance of Thinker, which has 71.79% F_1 score. There is no such a entity linking study for Turkish language to perform comparative evaluation. For English language, Cornolti et al. [7] compare publicly available entity annotation systems. They observe best entity linking performance with TagMe, which is 65.6% F_1 score on AIDA/CONLL dataset. The experimental results show that our system has a competitive performance in terms of accuracy, compared to the previous methods in the literature.

Table 5.6. General entity linking performance result of Thinker for the each news article.

Category	Article URL	Correct Mapping	Wrong Mapping	Undetected
Turkey	http://www.hurriyet.com.tr/yuksek-yargiya-yeni-duzen-geliyor-40116960	30	7	4
	http://www.hurriyet.com.tr/trtdeki-tepki-ceken-sozler-icin-bakanlar-kurulu-sonrasi-aciklama-40116953	37	7	4
	http://www.hurriyet.com.tr/derme-catma-sal-2-cana-mal-oldu-40116951	31	9	2
	http://www.hurriyet.com.tr/incirlik-ussundeki-feci-olumun-nedeni-belli-oldu-40116950	37	9	1
	http://www.hurriyet.com.tr/halk-otobusuyle-servis-minibusu-carpisti-21-yarali-40116940	25	15	1
World	http://www.hurriyet.com.tr/almanyadaki-turk-vekillerden-sert-aciklamalar-40116941	36	9	5
	http://www.hurriyet.com.tr/kahraman-cankurtaran-kucuk-kizin-hayatini-kurtardi-40116840	19	7	0
	http://www.hurriyet.com.tr/ab-turkiye-vize-muafiyetine-giderek-yaklasiyor-40116824	29	9	0
	http://www.hurriyet.com.tr/turkiye-dahil-tum-dunya-orlando-saldirisini-kinadi-40116822	36	11	0
	http://www.hurriyet.com.tr/ingiltere-kralicesi-dogum-gununu-sokak-partisiyle-kutladi-40116769	24	15	2
Economy	http://www.hurriyet.com.tr/gen-takside-sahte-plakayla-kiralama-40116963	28	7	0
	http://www.hurriyet.com.tr/52-yillik-uygulama-kalkiyor-tek-nushada-damga-vergisi-alinacak-40116942	18	9	1
Category	**Article URL**	**Correct Mapping**	**Wrong Mapping**	**Undetected**

Economy	http://www.hurriyet.com.tr/brent-petrol-50-dolarin-altina-geriledi-40116866	18	10	1
	http://www.hurriyet.com.tr/safiport-derinceye-deprem-raporu-40116888	17	6	0
	http://www.hurriyet.com.tr/bin-320-nufuslu-koyun-kiraz-ihracati-basladi-40116925	21	4	1
Sports	http://www.hurriyet.com.tr/riise-futbolu-birakti-40116959	28	16	3
	http://www.hurriyet.com.tr/wesley-sneijderin-kardesinden-ilginc-ozan-tufan-paylasimi-40116955	21	8	1
	http://www.hurriyet.com.tr/irlanda-1-1-isvec-macin-ozeti-40116933	19	8	4
	http://www.hurriyet.com.tr/potanin-perilerinin-ilk-rakibi-arjantin-40116914	15	5	1
	http://www.hurriyet.com.tr/8-metreden-dusen-taraftar-oldu-40116910	34	18	2

Table 5.7. Performance values of the Thinker on the general test data set.

Precision (%)	**Recall (%)**	**F_1 (%)**
73.46	70.20	71.79

5.2.4. Entity Linking Runtime Efficiency

Runtime efficiency is one of the key features of entity linking systems. The amount of unstructured data increases exponentially and an entity linking system must be scalable to process the increasing amount of data. In order to measure runtime efficiency and scalability of Thinker, we experimented Thinker with the data set introduced in Section 5.2.2. Figure 5.8 shows the performance values for this experiment. We observed that entity linking is performed appropriately in 5.5 milliseconds for per word. The average word size of the news data set is approximately 400 words that means linking of a news article takes approximately two seconds. We also observed that entity linking duration is incremented linearly with increasing input sizes. We could fit a line to the experiment results as seen in Figure 5.7.

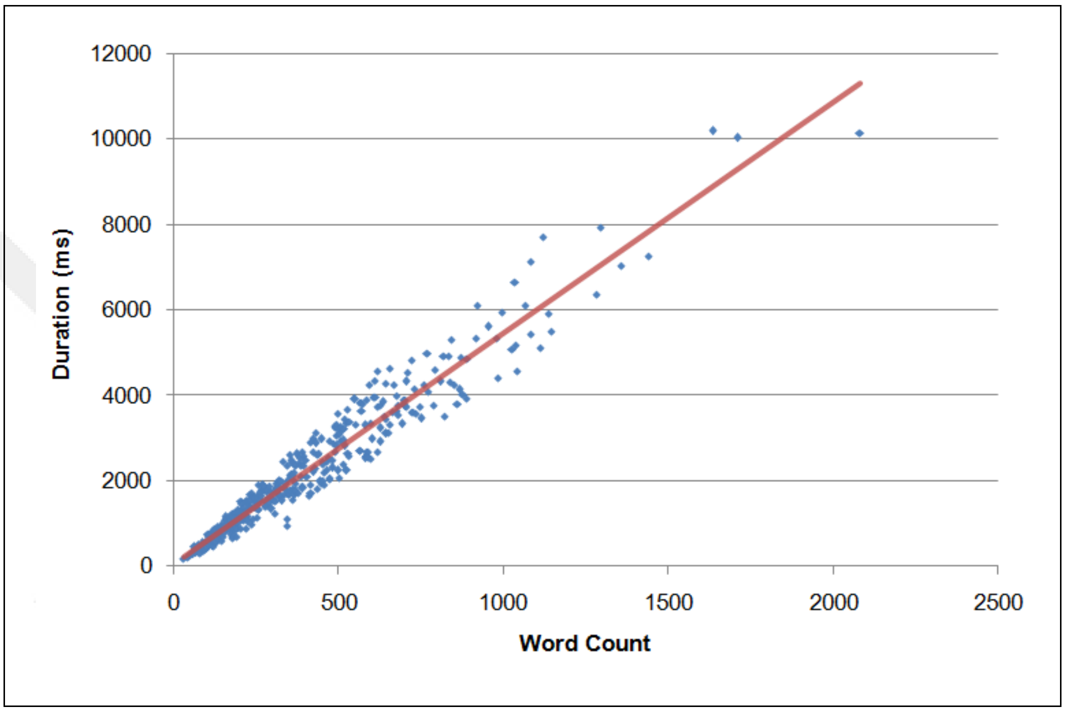


Figure 5.7. Runtime performances of Thinker with varying input sizes.

5.2.5. Entity Discovery Experimental Setup

In order to tune the parameters of the proposed Entity Discovery algorithm, content of Vikipedi articles are used. As described in Section 4.3.2.6, a set of labeled data with *36,245* instances is created by using Vikipedi articles. The labels (tags) are listed in Table 5.8. The most popular three labels are: “yerleşim (location)” with *2,917* instances, “film (film)” with *2,649* instances and “müzik sanatçısı (singer)” with *2,376* instances. The data set is split into two parts: 70 for training the classifiers, 30 for testing the classifiers.

Table 5.8. List of curated 200 tags curated from Vikipedi that is used in entity discovery.

abd eyalet, ada, albüm, almanya yerleşim yeri, amfibi, anatomi, antlaşma, arma, asker, askeri birim, askeri yapı, ateşli silah, avrupa yakası karakteri, azerbaycan rayon, bakteri, balık, baraj, basketbol, basketbol kulübü, basketbol ligi, basketbolcu, bayrak, bayram, bilim adamı, bisikletçi, bitki, biyoloji, bm, böcek, bulgaristan il, buz patencisi, cadde, cep telefonu, çin eyalet, çizgi roman karakteri, dağ, deprem, dergi, dil, dil ailesi, dini yapı, dizi, doctor who bölüm, doctor who karakteri, edebiyat, eski idari bölüm, eski ülke, etnik grup, eurovision, fakülte, festival, film, filozof, fizik, futbol kulübü, futbol kulübü sezonu, futbol ligi, futbol ligi sezonu, futbol maçı, futbol turnuvası, futbolcu, galaksi, gazete, gemi, göl, güneşdışı gezegen, güreşçi, hakem, hanedan, harf, harry potter karakteri, hastalık, havalimanı, havayolları, hazır gıda, hükümdar, hükümet kurumu, il, ilçe, insan, iran köy, islam, işletim sistemi, italya belde, italya il, italya komün, kanton, karakter, kimya, kişi, kitap, konser turnesi, köpek ırkı, köprü, koruma alanı, kral tv vmö, kraliyet, kurgusal karakter, kuruluş, kuş, lost, lost karakteri, makam sahibi, manken, marş, matematik, memeli, milli futbol cemiyeti, millî futbol takımı, mimar, mitoloji, müze, müzik, müzik grubu, müzik sanatçısı, müzik türü, nba draft, nba takımı, nehir, ödül, okul, olimpiyatlar, olimpiyatlarda etkinlik, opera, örümcek, otomobil, öykü, oyun, oyuncu, para, parlamento, plak şirketi, porno yıldızı, portekiz belediye, portekiz bucak, primat, prison break bölümü, programlama dili, radyo istasyonu, renk, roman, rusya federasyonu idari birimi, sanat eseri, sanatçı, sanatçı diskografisi, şarkı, savaş, sayı, seçim, şehir, seri, silahlı kuvvet, simpsonlar bölümü, single, şirket, sivil toplum kuruluşu, sivillere karşı saldırı, siyasi makam, siyasi parti, spor derbisi, spor ligi, sporcu, stadyum, sunucu, sure, sürüngen, takımyıldız, tekli, telekomünikasyon şirketi, televizyon, televizyon sezonu, tenis sporcu, tiyatro oyunu, türkiye, türkiye belde, türkiye il, türkiye ilçe, türkiye köy, türkiye mahalle, tv bölüm, tv kanalı, uçak, ukrayna idari birimi, ülke, uluslararası futbol turnuvası, üniversite, uydu, uzay aracı, video oyunu, voleybol kulübü, voleybolcu, website, yapı, yazar, yazılım, yerleşim, yılan, yıldız savaşları karakteri, yol, yüksek yapı

To evaluate performance of the proposed Entity Discovery algorithm, the corpus [97] created by Bilkent Information Retrieval Group[31] from Milliyet online newspaper is used. The corpus contains 408,305 documents; they are the news articles and columns of five years, 2001 to 2005, collected from the Turkish newspaper Milliyet. The experimental data set is created from Milliyet corpus in two steps.

[31] www.cs.bilkent.edu.tr/˜canf/bilir_web/

Firstly, Milliyet corpus is given as input to the Candidate Entity Detector module and a list of candidate entities are extracted. This process produces hundreds of 1-grams, 2-grams, 3-grams and 4-grams as candidate entities. However, these identified candidate entity mentions are noisy; not all of them are real entities. Concurrent entity mentions, typos and HTML tables are the main causes for this problem. Therefore, a human intervention is needed to finalize the candidate entity detection process. In the end, we manually selected and annotated 150 candidate entities as the test data. A sample from the data set is provided in Table 5.10.

5.2.6. Entity Discovery Results and Discussion

Turkish Entity Discovery system depends on several parameters therefore before evaluating the system performance, the system parameters are tuned on the evaluation data set. Therefore firstly, fine-grained entity recognizer is evaluated with different classifier algorithms and with varying word vector sizes on the Vikipedi data set. In the experiments, the linear classifiers SVM, Logistic Regression and Softmax have been utilized and the vector sizes (from 50 to 200) have been tested. Table 5.9 shows the performance values for this experiment. We observed better performance values with SVM classier and when larger vector sizes are used. Hence, the best performance result (78.69%) is obtained with SVM classifier and a vector size of 200. This setting is used for further experiments to assess the final performance of the system.

Table 5.9. Evaluation of the fine-grained entity recognizer algorithm with varying classifiers and word vector sizes.

Vector size	**SVM (%)**	**Logistic regression (%)**	**Softmax (%)**
50	74.16	73.15	73.30
100	77.08	75.90	74.28
200	78.69	77.51	74.75

We also evaluated our fine-grained entity recognition algorithm for the English language in order to prove that our approach is language independent. In a similar way, English Wikipedia articles are processed and again the most frequently occurring 100 entity types are determined and the experimental data set is formed. By using an English NLP tool[32] and the Glove [12] word vectors for English, the entity vectors are created for English Wikipedia articles, too. We observed 77.14% accuracy for English, which is a very close performance compared to the experimental results on Turkish.

In our last experiment, manually created Milliyet dataset is evaluated by using our fine-grained named entity recognizer. The experiment is resulted with 61.95% accuracy for strict typing of entities and 73.45% for relaxed typing of entities. In contrast to strict typing, classifying entities with a more general type is evaluated as a correct assignment in relaxed typing such as sportsman instead of footballer. Sample results are listed in Table 5.10; Column one contains entity titles and column 3 contains the manually assigned type information.

[32] https://opennlp.apache.org/

Table 5.10. Fine-grained entity recognition sample results of Milliyet test data.

Candidate Entity	**Prediction**	**Correct**
Albert Einstein	Bilim adamı (scientist)	Bilim adamı (scientist)
Arı (bee)	Böcek (bug)	Böcek (bug)
Avrupa Şampiyonlar Ligi	Futbol ligi (football league)	Futbol ligi (football league)
Dallas Mavericks	Nba takımı (NBA team)	Nba takımı (NBA team)
Doberman	Köpek ırkı (dog)	Köpek ırkı (dog)
Elma (apple)	Bitki (plant)	Bitki (plant)
Facebook	Websitesi (website)	Websitesi (website)
Financial Times	Albüm (album)	Gazete (newspaper)
Florya Metin Oktay Tesis	Futbolcu (footballer)	Yapı (construction)
Hamsi	Balık (fish)	Balık (fish)
Harun Doğan	Person	Sporcu (sportsman)
Hristiyan	Eski ülke (former country)	Din (religion)
İbo Show	Televizyon (TV)	Televizyon (TV)
İngilizce	Dil (language)	Dil (language)
Michael Schumacher	Oyuncu (actor)	Sürücü (driver)
Pablo Montoya	Otomobil (automobile)	Sürücü (driver)
Polat Renaissance Otel	Yapı (construction)	Yapı (construction)
Portakal (orange)	Bitki (plant)	Bitki (plant)
Real Madrid	Yerleşim (location)	Futbol kulübü (football club)
Robert Pearson	Makam sahibi (officeholder)	Makam sahibi (officeholder)
Sezen Aksu	Müzik sanatçısı (singer)	Müzik sanatçısı (singer)
Sunday Times	Gazete (newspaper)	Gazete (newspaper)
Suudi Arabistan	Ülke (country)	Ülke (country)
Tansiyon	Hastalık (disease)	Hastalık (disease)
Türk Hava Yolları	Havayolları (airlines)	Havayolları (airlines)
Van Gölü	Göl (lake)	Göl (lake)
Van Hooijdonk	Futbolcu (footballer)	Futbolcu (footballer)

Note that the accuracy of this second experiment is lower compared to the first one. In fact this is an expected result, since the classifier is trained by using Vikipedi pages and it is tested with different entities but again collected from Vikipedi in the first experiment. However, in the second experiment a general corpus collected from an online newspaper is utilized. Moreover, missing types and types from the same domain reduce the performance. For example, "Pablo Montoya" is classified as an automobile rather than a driver in the experiments. The reasons behind this wrong assignment is driver and automobile entities are related entities that share similar contextual features (words). Hence, such misclassifications occur in the system. In order to handle this kind of wrong assignments, more sophisticated features and algorithms are needed.

6. CONCLUSIONS AND FUTURE WORK

This study has presented Videolization, a knowledge graph based visual interpretation system that automatically interprets visually given Turkish or English textual Web content by using Semantic Web based technologies. As a use case of Videolization system, Wikipedia articles are automatically converted into videos. Visualization of text content is the key challenge in the proposed system. To address this problem, a template based visualization algorithm is proposed in this study, which leverages DBpedia as a knowledge graph.

This study has also presented the Thinker system that can be used for linking Turkish documents with Turkish Wikipedia and the Turkish dictionary. Entity Disambiguation is the key challenge in the proposed Turkish entity linking system. To address this problem, a fusion of knowledge-based methods and supervised machine learning algorithms is proposed, which leverages Turkish linguistic features, the deep learning models and Wikipedia graph structure. Moreover, a series of experiments is conducted to evaluate the performance of the system. Evaluations show that the proposed system has a satisfactory performance for the task. Turkish Entity Linker system outperformed the Simplified Lesk algorithm, which is a well-known and commonly used disambiguation method.

The effectiveness of Videolization is validated empirically over opinion surveys. Evaluations show that the proposed system has a satisfactory video generation performance. 70% of survey users indicated that they would like to use our system to consume Web content on their TVs.

Our research on Web content visualization differs from the previous studies mainly in two points. Firstly, we used Semantic Web technologies to visualize Web content. Secondly, our approach is domain independent and it can also be applied to most of the Web content such as news articles, blog posts, etc..

Our work on entity linking can be distinguished from previous work in several ways. First of all, Thinker is the first proposed system for Turkish entity linking process. Secondly, unlike previous work for other languages, Thinker uses fusion of knowledge based methods and supervised machine learning algorithms that utilize a rich set of features in order to link Turkish entities. Various methods and features that can handle the agglutinative and free word structure of the Turkish language are also proposed in this study. Lastly, a comprehensive Turkish knowledge base is generated by integrating Vikipedi and the Turkish dictionary in order to cover the majority of the Turkish entities.

In conclusion, this study has demonstrated the potential and promise of the knowledge graphs for the text visualization task and deep learning approaches for Turkish entity linking task. We believe that Videolization system would allow the users to “surf the Web” in completely different yet satisfying way. They would be able to "play Internet" rather than "browsing" the content. Integrating the system with a digital assistant and realizing dynamic video generation in real-time based on user interactions would be a breakthrough technology.

As future work, the proposed Videolization and Thinker systems could be improved in several directions.

- Text visualization algorithm can be improved in several directions. An image retrieval based approach over an annotated video or image library can provide more suitable visual representations to represent a given entity. Moreover, a sentence could be visualized better with the combination of multiple entities instead of using visual representation for only the most significant entity.

- Entity detection performance of Thinker can be improved by realization of a Turkish chunker (Shallow parsing) system that identifies constituent parts of sentences (nouns, verbs, adjectives, etc.) and then that links them to higher order units with discrete grammatical meanings (noun groups or phrases, verb groups, etc.).

- Entity disambiguation performance of Thinker can be improved by removing duplicate definitions of entities. Note that some entities are both defined in Turkish Wikipedia and the Turkish dictionary.

REFERENCES

1. A. A. Akin and M. D. Akin. Zemberek, An Open Source NLP Framework for Turkic Languages. *Structure*, 10:1-5, 2007.

2. G. Eryigit. Itu Turkish Nlp Web Service. *The Association for Computer Linguistics*, 1-4, 2014.

3. Y. B. Goodfellow and A. Courville. Deep Learning. *MIT Press*, 2015.

4. Stanford University, “Ufdl Tutorial - Autoencoders”, http://deeplearning.stanford.edu/ wiki/index.php/ ufldl_tutorial [retrieved 07 February 2016].

5. T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. *Arxiv*, 1301.3781, 2013.

6. M. Kalender and J. Dang. Skmt: A Semantic Knowledge Management Tool for Content Tagging, Search and Management. *International Conference on Semantics, Knowledge and Grids*, 112-119, 2012.

7. M. Cornolti, P. Ferragina, and M. Ciaramita. A Framework for Benchmarking Entity-Annotation Systems. *Proceedings of The 22nd International Conference on World Wide Web*, 249-260, 2013.

8. L. Heck and H. Huang. Deep Learning of Knowledge Graph Embeddings for Semantic Parsing of Twitter Dialogs. *Signal and Information Processing (GlobalSIP)*, 597-601, 2014.

9. K. Tanaka. Research on Fusion of the Web and TV Broadcasting. *Informatics Research for Development of Knowledge Society Infrastructure*, 129-136, 2007.

10. H. Shim, B. Kang, and K. Kwag. Web2Animation - Automatic Gene*ration of 3D Animation from the Web Text. Proceedings of the 2009 IEEE/WIC/ACM International*

Joint Conference on Web Intelligence and Intelligent Agent Technology, 01:596-601, 2009.

11. R. Socher, A. Karpathy, Q. V. Le, C. D. Manning, and A. Y. Ng. Grounded Compositional Semantics for Finding and Describing Images with Sentences. *Transactions of the Association for Computational Linguistics*, 2:207-218, 2014.

12. J. Pennington, R. Socher, and C. D. Manning. Glove: Global Vectors for Word Representation. *Empirical Methods in Natural Language Processing*, 1532-1543, 2014.

13. R. Collobert and J. Weston. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. *Proceedings of the 25th International Conference on Machine Learning*,160-167, 2008.

14. D. Yuret and F. Türe. Learning Morphological Disambiguation Rules for Turkish. *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, 328-334, Association for Computational Linguistics, 2006.

15. Wikipedia, The Free Encyclopedia, “Turkish Language”, https://en.wikipedia.org/wiki/Turkish_Language [retrieved 07 February 2016].

16. K. Oflazer. Error-Tolerant Finite-State Recognition with Applications to Morphological Analysis and Spelling Correction. *Computational Linguistics*, 22:73-89, 1996.

17. M. Şahin, U. Sulubacak, and G. Eryiğit. Redefinition of Turkish Morphology Using Flag Diacritics. *Proceedings of the Tenth Symposium on Natural Language Processing*, 2013.

18. H. Sak, T. Güngör, and M. Saraçlar. Turkish Language Resources: Morphological Parser, Morphological Disambiguator and Web Corpus. *Advances in Natural Language Processing*, 417-427, Springer Berlin Heidelberg, 2008.

19. C. Coltekin. A Freely Available Morphological Analyzer for Turkish. *Language Resources and Evaluation Conference*, European Language Resources Association, 2010.

20. H. Sak, T. Güngör, and M. Saraçlar. *Computational Linguistics and Intelligent Text Processing*, 107-118, Springer Berlin Heidelberg, 2007.

21. D. Z. Hakkani-Tür, K. Oflazer, and G. Tür. Statistical Morphological Disambiguation for Agglutinative Languages. *Computers and the Humanities,* 36:381-410, 2002.

22. E. Mert and G. Dalkiliç. Word Sense Disambiguation for Turkish. *Computer and Information Sciences*, 205-210, IEEE, 2009.

23. B. Ilgen, E. Adali, and A. C. Tantug. A Comparative Study To Determine the Effective Window Size of Turkish Word Sense Disambiguation Systems *Information Sciences and Systems*, 169-176, Springer, 2013.

24. Z. Orhan and Z. Altan. Word Sense Disambiguation for Semantic Applications. *GI*, 94:321-328, 2006.

25. G. A. Seker and G. Eryigit. Initial Explorations on Using CRFS for Turkish Named Entity Recognition. *24th International Conference on Computational Linguistics*, 2459-2474, 2012.

26. D. Küçük. Named Entity Recognition Experiments on Turkish Texts. *Flexible Query Answering Systems,* 524-535, Springer, 2009.

27. R. Socher. Recursive Deep Learning for Natural Language Processing and Computer Vision. *Doctoral Dissertation,* Stanford University, 2014.

28. Y. Bengio, A. C. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives *Pattern Analysis and Machine Intelligence,* 35:1798-1828, 2013.

29. D. E. Rumelhart, G. E. Hinton, and R. J. Wilson. Learning Representations by Back-

Propagating Errors. *Cognitive Modeling*, 323:533-536, 1988.

30. G. E. Hinton and R. R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science,* 313:504-7, 2006.

31. D. Erhan, Y. Bengio, A. C. Courville, P.A. Manzagol, P. Vincent, and S. Bengio. Why Does Unsupervised Pre-Training Help Deep Learning? *The Journal of Machine Learning Research*, 11:625-660, 2010.

32. P. Vincent, H. Larochelle, Y. Bengio, and P.A. Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. *Proceedings of the 25th International Conference on Machine Learning*, 1096-1103, ACM, 2008.

33. G. E. Dahl, M. Ranzato, A. Rahman Mohamed, and G. E. Hinton. Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine. *Advances in Neural Information Processing Systems*, 469-477, Curran Associates, Inc., 2010.

34. Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A Neural Probabilistic Language Model. *Innovations in Machine Learning*, 3:1137-1155, 2003.

35. T. Mikolov and G. Zweig. Context Dependent Recurrent Neural Network Language Model. *Spoken Language Technology Workshop*, 234-239, IEEE, 2012.

36. T. Luong, R. Socher, and C. D. Manning. Better Word Representations with Recursive Neural Networks for Morphology. *The Conference on Natural Language Learning*, 104-113, ACL, 2013.

37. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. *Advances in Neural Information Processing Systems,* 3111-3119, 2013.

38. Q. V. Le and T. Mikolov. Distributed Representations of Sentences and Documents. *Arxiv*, Preprint Arxiv:1405.4053, 2014.

39. A. M. Dai, C. Olah, Q. V. Le, and G. S. Corrado. Document Embedding with Paragraph Vectors, *Arxiv*, Preprint Arxiv:1507.07998, 2015.

40. L. Van Der Maaten and G. Hinton. Visualizing High-Dimensional Data using TSNE. *Journal of Machine Learning Research*, 1:2579-2605, 2008.

41. L. F. Rau. Extracting Company Names from Text. *Artificial Intelligence Applications,* 29-32, IEEE, 1991.

42. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *The Scientific American*, 17:28-37, 2001.

43. A. Mathes. Folksonomies - Cooperative Classification and Communication Through Shared Metadata, http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html [retrieved 07 February 2016].

44. M. Obitko. Semantic Web Architecture, http://www.obitko.com/tutorials/ontologies-semantic-web/semantic-web-architecture.html [retrieved 07 February 2016].

45. W3c. "Rdf - Semantic Web Standards", https://www.w3.org/rdf/ [retrieved 07 February 2016].

46. F. V. H. Grigoris Antoniou. A Semantic Web Primer. MIT Press, 2004.

47. W3c. "Owl Web Ontology Language", https://www.w3.org/tr/owl-features/ [retrieved 07 February 2016].

48. T. Gruber. Ontology (Computer Science) - Definition in Encyclopedia of Database Systems, http://tomgruber.org/writing/ontology-definition-2007.htm [retrieved 07 February 2016].

49. W3c. "Web Service Modeling Ontology (WSMO)", https://www.w3.org/submission/wsmo/ [retrieved 07 February 2016].

50. G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction To Wordnet: An On-Line Lexical Database. *International Journal of Lexicography*, 21:235-44, 1990.

51. L. Reeve and H. Han. Semantic Annotation for Semantic Social Networks using Community Resources. *The Sigmas Conference and Event Center*, 2:52-6, 2005.

52. B. Hachey, W. Radford, J. Nothman, M. Honnibal, and J. R. Curran. Evaluating Entity Linking with Wikipedia. *Artificial Intelligence*, 194:130-150, 2013.

53. A. Gattani, D. S. Lamba, N. Garera, M. Tiwari, X. Chai, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan, and A. Doan. Entity Extraction, Linking, Classification, and Tagging for Social Media: A Wikipedia-Based Approach. *The VLDB Conference*, 6:1126-1137, 2013.

54. M. Kalender, J. Dang, and S. Üsküdarli. Semantic Tagprint - Tagging and Indexing Content for Semantic Search and Content Management. *Semantic Computing,* 260-267, IEEE, 2010.

55. P. Ferragina and U. Scaiella. Tagme: On-The-Fly Annotation of Short Text Fragments (By Wikipedia Entities). *International Conference on Information and Knowledge Management*, 1625-1628, ACM, 2010.

56. D. Milne and I. H. Witten. Learning To Link with Wikipedia. *Information and Knowledge Management*, 509-518, ACM, 2008.

57. D. Ceccarelli, C. Lucchese, S. Orlando, R. Perego, and S. Trani. Learning Relatedness Measures for Entity Linking. *Conference on Information and Knowledge Management*, 139-148, ACM, 2013.

58. D. Ceccarelli, C. Lucchese, S. Orlando, R. Perego, and S. Trani. Dexter: An Open Source Framework for Entity Linking. *Proceedings of the Sixth International Workshop on Exploiting Semantic Annotations in Information Retrieval*, 17-20, ACM, 2013.

59. X. Hu and B. Wu. Automatic Keyword Extraction Using Linguistic Features. *The IEEE International Conference on Data Mining Series*, 19-23, IEEE Computer Society, 2006.

60. Y. Matsuo and M. Ishizuka. Keyword Extraction from A Single Document Using Word Co-Occurrence Statistical Information. *International Journal on Artificial Intelligence Tools*, 13:157-169, 2004.

61. P. D. Turney. Learning Algorithms for Keyphrase Extraction. *Information Retrieval*, 2:303-36, 2000.

62. J. Wang and H. Peng. Keyphrases Extraction from Web Document by the Least Squares Support Vector Machine. *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, 293-296, IEEE Computer Society, 2005.

63. X. Li, X. Wu, X. Hu, F. Xie, and Z. Jiang. Keyword Extraction Based on Lexical Chains and Word Co-Occurrence for Chinese News Web Pages. *Data Mining Workshops*, 744-751, IEEE Computer Society, 2008.

64. J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust Disambiguation of Named Entities in Text. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 782-792, Association for Computational Linguistics, 2011.

65. E. Meij, W. Weerkamp, and M. De Rijke. Adding Semantics To Microblog Posts. *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, 563-572, ACM, 2012.

66. S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. *Proceedings of the 15th ACM Sigkdd International Conference on Knowledge Discovery and Data Mining*, 457-466, ACM, 2009.

67. L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and Global Algorithms for

Disambiguation To Wikipedia. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 1:1375-1384, Association for Computational Linguistics, 2011.

68. P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. Dbpedia Spotlight: Shedding Light on the Web of Documents. *Proceedings of the 7th International Conference on Semantic Systems*, 1-8, ACM, 2011.

69. J. Hoffart, F. M. Suchanek, K. Berberich, E. Lewis-Kelham, G. De Melo, and G. Weikum. Yago2: Exploring and Querying World Knowledge in Time, Space, Context, and many Languages. Proceedings of the 20th International Conference Companion on World Wide Web, 229-232, ACM, 2011.

70. I. H. Witten and D. Milne. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy*, 25-30, AAAI Press, 2008.

71. M. Lesk. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell A Pine Cone from an Ice Cream Cone. *Proceedings of the 5th Annual International Conference on Systems Documentation*, 24-26, ACM, 1986.

72. E. Agirre and P. G. Edmonds, Word Sense Disambiguation: Algorithms and Applications. Springer Science and Business Media, Springer, 33, 2006.

73. D. Nadeau and S. Sekine. A Survey of Named Entity Recognition and Classification. *Linguisticae Investigationes*, 30:3-26, 2007.

74. N. Nakashole, T. Tylenda, and G. Weikum. Fine-Grained Semantic Typing of Emerging Entities. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 1:1488-1497, ACL, 2013.

75. X. Ling and D. S. Weld. Fine-Grained Entity Recognition. *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, 2012.

76. T. Lin and O. Etzioni. No Noun Phrase Left Behi*nd: Detecting and Typing Unlinkable Entities. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 893-903, ACL, 2012.

77. M. A. Ur Rahman and V. Ng. Inducing Fine-Grained Semantic Classes Via Hierarchical and Collective Classification. *Proceedings of the 23rd International Conference on Computational Linguistics*, 931-939, Tsinghua University Press, 2010.

78. M. A. Yosef, S. Bauer, J. Hoffart, M. Spaniol, and G. Weikum. Hyena: Hierarchical Type Classification for Entity Names. 2012.

79. K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. *Proceedings of the 2008 ACM Sigmod International Conference on Management of Data*, 1247-1250, ACM, 2008.

80. F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. *Proceedings of the 16th International Conference on World Wide Web*, 697-706 ACM Press, 2007.

81. D. Yogatama, D. Gillick, and N. Lazic. Embedding Methods for Fine Grained Entity Type Classification. *Proceedings of the 53rd Annual Meeting of The Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, 2:291-296, ACL, 2015.

82. K. Sumi and K. Tanaka. Transforming Web Contents Into a Storybook with Dialogues and Animations. *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, 1076-1077, ACM, 2005.

83. B. Coyne and R. Sproat. Wordseye: An Automatic Text-To-Scene Conversion System. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive*

Techniques, 487-496, ACM, 2001.

84. X. Zhu, A. B. Goldberg, M. Eldawy, C. R. Dyer, and B. Strock. A Text-To-Picture Synthesis System for Augmenting Communication. *Proceedings of the 22nd National Conference on Artificial Intelligence,* 2:1590-1595, AAAI Press, 2007.

85. R. Mihalcea and C. W. Leong. Toward Communicating Simple Sentences using Pictorial Representations. *Machine Translation,* 22:153-173, 2008.

86. A. Borman, R. Mihalcea, and P. Tarau. Picnet: Augmenting Semantic Resources with Pictorial Representations. *AAAI Spring Symposium: Knowledge Collection from Volunteer Contributors,* 1-7, AAAI, 2005.

87. C. L. Zitnick, D. Parikh, and L. Vanderwende. *Proceedings of the IEEE International Conference on Computer Vision,* 1681-1688, IEEE, 2013.

88. V. Hansen. Interactive Television Design - Designing for Interactive Television V 1.0 Bbci and Interactive TV Programmes, http://www.bbc.co.uk/guidelines/futuremedia/desed /itv/ itv_design_v1_2006.pdf [retrieved 07 February 2016].

89. A. Nenkova and K. Mckeown. A Survey of Text Summarization Techniques. *Mining Text Data*, 43-76, Springer, 2012.

90. W. Shen, J. Wang, and J. Han. Entity Linking with A Knowledge Base: Issues, Techniques, and Solutions. *Knowledge and Data Engineering, IEEE Transactions on* 1:443-60, 2015.

91. R. Navigli. Word Sense Disambiguation: A Survey. *ACM Computing Surveys*, 2, 2009.

92. Vikipedi, “The Turkish Wikipedia homepage”, https://tr.wikipedia.org/ [retrieved 07 February 2016].

93. A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic Clustering of

the Web. *Computer Networks and ISDN Systems*, 29:1157-1166, 1997.

94. M. A. Yosef, S. Bauer, J. Hoffart, M. Spaniol, and G. Weikum. Hyena: Hierarchical Type Classification for Entity Names. *Conference on Computational Linguistics*, 2012.

95. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: Nsga-II. *Evolutionary Computation. IEEE Transactions*, 6:182-9, 2002.

96. C. Xing, D. Wang, X. Zhang, and C. Liu. Document Classification with Distributions of Word Vectors. *Asia-Pacific Signal and Information Processing Association*, 1-5, IEEE, 2014.

97. F. Can, S. Kocberber, E. Balcik, C. Kaynak, H. C. Ocalan, and O. M. Vursavas. Information Retrieval on Turkish Texts. *Journal of the American Society for Information Science and Technology*, 1:407-21, 2008.

98. R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C. J. Lin. Liblinear: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871-1874, 2008.

99. J. Heaton. Encog: Library of Interchangeable Machine Learning Models for Java and C#. *Arxiv*, Preprint Arxiv:1506.04776, 2015.

100. Y. Liu, D. Zhang, G. Lu, and W.Y. Ma. A Survey of Content-Based Image Retrieval with High-Level Semantics. *Pattern Recognition*, 40:262-82, 2007.

101. N. Uzzaman, J. P. Bigham, and J. F. Allen. Multimodal Summarization of Complex Sentences. *Proceedings of the 16th International Conference on Intelligent User Interfaces*, 43-52, ACM, 2011.

APPENDIX A: VIDEOLIZATION TEST DATA SET

This section provides the details of the test data used in evaluating runtime performance of Videolization. Table A.1 shows the most accessed Wikipedia article titles and number of views respectively.

Table A.1. The most accessed 100 English Wikipedia articles between 20 December 2015 and 27 December 2015, which are used for measuring runtime efficiency and scalability of Videolization system.

Rank	Article Title	Views
1	Main Page	118,989,073
2	Star Wars: The Force Awakens	4,905,073
3	Star Wars	2,703,355
4	Augusto Pinochet	1,380,919
5	Dilwale (2015 film)	1,241,440
6	Daisy Ridley	1,227,591
7	Pia Wurtzbach	1,224,259
8	'Tis the Season	1,220,089
9	Miss Universe 2015	1,123,111
10	Bajirao I	1,036,734
11	Bajirao Mastani (film)	1,008,736
12	Web scraping	991,645
13	Java (programming language)	914,861
14	Mastani	880,929
15	Göran Kropp	874,143
16	The Revenant (2015 film)	850,458
17	Star Wars (film)	808,903
18	Boxing Day	806,356
19	Kylo Ren	776,676
20	Adam Driver	775,776
21	Carrie Fisher	745,081
22	Miss Universe	731,473

Rank	Article Title	Views
23	Harrison Ford	693,925
24	Christmas	677,216
25	Mark Hamill	676,622
26	Star Wars sequel trilogy	669,203
27	Steven Avery	662,435
28	Steve Harvey	660,099
29	Joy (film)	620,980
30	Rogue One: A Star Wars Story	594,168
31	Book of the Dead	593,062
32	The Hateful Eight	588,492
33	Macaulay Culkin	580,647
34	Justin Berfield	570,016
35	{[}compensation{]}	564,480
36	Return of the Jedi	543,815
37	Deaths in 2015	543,599
38	Star Wars Episode I: The Phantom Menace	541,764
39	List of Miss Universe titleholders	526,445
40	USS Enterprise (CVN-65)	525,253
41	Joy Mangano	509,112
42	Darth Vader	505,264
43	'Tis the Season (Vince Gill and Olivia Newton-John album)	484,781
44	John Boyega	484,403
45	Donald Trump	465,532
46	Adele	463,874
47	The Empire Strikes Back	453,489
48	Star Wars Episode III: Revenge of the Sith	448,752
49	List of highest-grossing films	446,750
50	Santa Claus	445,770
51	Oscar Isaac	445,739
52	Charles Woodson	430,634
53	Festivus	430,274
54	Kim Peek	423,323
55	One-Punch Man	409,690
56	Han Solo	405,909

Rank	Article Title	Views
57	Queen Sonja of Norway	403,053
58	Frodo Baggins	374,067
59	.rss	373,829
60	Hugh Glass	373,678
61	Star Wars Episode II: Attack of the Clones	365,766
62	The Twelve Days of Christmas (song)	357,127
63	Santa Claus's reindeer	354,057
64	Krampus	354,010
65	Elon Musk	353,882
66	J. J. Abrams	346,514
67	Luke Skywalker	338,516
68	George Lucas	338,191
69	A Christmas Story	333,259
70	Jessica Jones	326,276
71	Jessica Jones (TV series)	319,691
72	Pablo Escobar	316,274
73	Princess Leia	307,110
74	List of Bollywood films of 2015	306,412
75	Grand Moff Tarkin	304,574
76	Ddd	298,602
77	Bob, Agent of Hydra	291,870
78	2015 in film	289,672
79	Winter solstice	289,415
80	Hayden Christensen	289,120
81	Ariadna Gutiérrez	286,992
82	Olivia Jordan	286,487
83	2012 Delhi gang rape	281,580
84	Facebook	277,111
85	Creed (film)	269,647
86	Kwanzaa	264,970
87	Bing Crosby	264,653
88	It's a Wonderful Life	263,717
89	UFC on Fox: dos Anjos vs. Cerrone 2	262,190
90	Jon Jones	259,530

Rank	**Article Title**	**Views**
91	Rian Johnson	259,066
92	American Horror Story: Hotel	255,135
93	The Danish Girl (film)	249,787
94	DJ Khaled	248,293
95	United States	247,760
96	Odell Beckham Jr.	247,276
97	Lupita Nyong'o	244,112
98	Sicario (2015 film)	243,970
99	Jennifer Lawrence	243,793
100	Solo family	243,342

APPENDIX B: THINKER TEST DATA SET

This section provides a sample from our news articles data set, which are collected from Hurriyet online news paper in order to evaluate performance of the proposed entity linking algorithm. Table B.1 shows the target entity title, the addresses and content of the news articles, which are used for evaluating the entity disambiguation performance. Table B.2 shows a sample from general performance evaluation result of Thinker system.

Table B.1. A sample from the Hurriyet news articles data set with target entities, addresses and content.

Akrep(scorpion)
www.hurriyet.com.tr/gundem/23338821.asp
Mugla'nin Bodrum ilçesi Turgutreis beldesinde okulun bahçesinde akrep sokan ögrenci öldü. Turgutreis Anadolu Otelcilik ve Turizm Meslek Lisesi'nde okuyan ve staj yaptigi için okulda kalan Samet Çetin'i geçtigimiz cuma günü okul bahçesinde otururken akrep soktu. Okulda yapilan müdahalenin ardindan dinlenmeye çekilen Çetin, gece rahatsizlanmasi üzerine Bodrum Devlet Hastanesi'ne kaldirildi. Yapilan müdahalenin ardindan Bodrum Özel Hastanesi'ne sevk edilen ögrenci, durumunun agirlasmasi üzerine Ege Üniversitesi Hastanesi'ne sevk edildi. Çetin, burada yapilan tüm müdahalelere ragmen kurtarilamadi. Çetin'in cenazesi, ögle namazina müteakip kilinan cenaze namazinin ardindan Ortakent Mezarligi'nda topraga verildi. Jandarma akrep sokmasinin ardindan Samet Çetin'i bir an önce donanimli bir saglik kurulusuna ulastirmayan okul yöneticileriyle ilgili sorusturma baslatirken, Çetin'in ailesi de ogullarinin ölümünde okulun ihmali bulundugunu söyledi.
Akrep (horoscope)
www.hurriyet.com.tr/magazin/astroloji/burc.asp?cinsiyet=erkek&burc=akrep

Eğer bir Akrep erkeğine aşıksanız ve ihtiras sözcüğü sizi korkutuyorsa, ayakkabılarınızı ayağınıza geçirdiğiniz gibi kaçın. Listenin başında o olmasına karşın, ben romantik ihtirastan söz etmiyorum. Aynı zamanda politikaya, çalışmaya, dostluğa, dine, yiyeceğe, akrabalara, çocuklara, giyim kuşama, yaşama, ölüme ve düşünebileceğiniz herşeye karşı duyulan şiddetli ihtirastan söz ediyorum. Eğer duygusal aşırılıkları kabul etmeyen bir insansanız. Akrep erkeği kesinlikle sizin ruhunuzun ihtiyaç duyduğu biri değildir.Sakın arkanıza bakmayın. Hemen kaçın.   Eğer Akrep erkeği ile yeni tanıştıysanız, onun ne kadar sakin ve dengeli bir insan olduğunu düşünebilirsiniz. Böylesine açıkça kendini kontrol edebilen bir insan nasıl ihtiraslı, hem de tehlikeli şekilde ihtiraslı olabilir. Çünkü o, yüzeydeki serin görünüşüyle sadece blöf yapmaktadır. O, aldatıcı şekilde konrollü davranışlarının altında cızır cızır yanmaktadır. Sakın dokunmayın. Onunla oynarken dikkatli olun. Nereye ve kiminle gittiğinizden emin olun. Akrep erkeği ile kurduğu ilişkide kendini güvende sanan hanımlara gelince; bakalım şu hipnotik, delip geçici Akrep gözlerinin arkasında neyin gizli olduğunu görebilecek misiniz? Şurası kesin ki, o sizin üstünüzde nötr bir izlenim bırakmadı. Ya onun çocuksu ve tatlı olduğunu düşündünüz ya da yaramaz ve ihtiraslı. Ama o bunların hiçbiri değil ve asıl sorun da bu . Veya belki, her ikisi de olduğu söylenmeli.   Tek kelimeyle, bu adam yenilmek, yılmak nedir bilmeyen biridir. O buz gibi sessizliğin arkasında sürekli olarak fıkır fıkır kaynayan kocaman bir kap vardır. Şansınız varsa, kapağını ömür boyu sımsıkı kapalı tutar ama derin bir yara onu korkunç bir patlamayla havaya uçurabilir. Eğer tehlike çizgisi içinde değilseniz, seyretmek bayağı heyecan verici olabilir. Patlamanın yaklaştığını hissediyorsanız kenara çekilin ve sakın patlamaya neden olacak bir şeyi kendiniz yapmayın. Akrep, ikiz huyları olan ihtiras ve mantıkla sizi şaşkına çevirecektir. O, bunların ikisinin de uzmanıdır. Zeka ve duygular onu eşit şekilde yönetir. Akrep zeki olmanın da ötesindedir. Eğer çok gelişmiş biriyse, o aynı zamanda varoluşun sırlarıyla ilgilenen ve yanıtlarını bulmaya çok yaklaşan derin filozofça bilgiye sahip bir insandır.
Akrep (hour hand)
www.hurriyet.com.tr/akrep-ve-yelkovan-neden-saga-doner-28251059
Akrep ve Yelkovan neden sağa döner?Dünya'nın bir ülkesi hariç her yerinde saatler sağa doğru döner. Peki hangi ülke o? Akrep ve yelkovan neden sağa döner? İlk olarak eski Mısırlılar, güneşin her gün düzenli bir hareketle doğup, belirli zamanlarda gökyüzünün aynı noktalarında bulunup battığını gözlemlediler ve bunun bir günü zaman parçalarına ayırmada kullanılabileceğini keşfettiler.
Çay (stream)
www.hurriyet.com.tr/gundem/25860021.asp

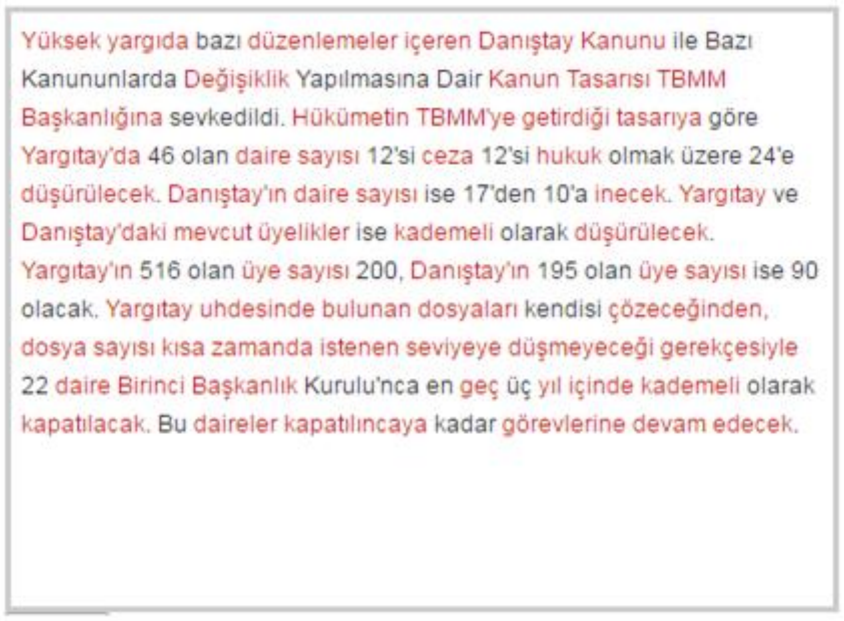
İSTANBUL'a su sağlayan Melen Çayı'nda kuraklığa bağlı olarak su seviyesinde düşme yaşanmaya başladı. Geçen yıllarda şubat ayı ile kıyaslandığında çaydaki su seviyesinde yaklaşık 80 santimlik düşüş olduğu belirtildi. Sakarya'nın Kocaali İlçesi'ne bağlı Ortaköy Beldesi'nde bulunan regülatör ile Melen Çayı'ndan İstanbul'a geçen yıl 159 milyon 170 bin metreküp su sağlandı. Bu yıl ise Melen Çayı'nın yüzde 80'inin bulunduğu Düzce'de kuraklık etkilerini göstermeye başladı. Melen Çayı'nda ve besleyen derelerde su seviyesi düştü. Geçen yıl şubat ayı ile kıyaslandığında çaydaki su seviyesinde yaklaşık 80 santimlik düşüş olduğu belirlendi. Düzce'nin Cumayeri İlçesi Dokuzdeğirmen Köyü'ndeki köprünün ayaklarında ve çay yatağında su seviyesindeki düşüşün izleri belli oluyor. Geçen yıllarda mayıs ayına kadar karların bulunduğu Kardüz Yaylası'nda ise çok az kar bulunması kuraklığın boyutunu da gösteriyor.
Çay (tea)
www.hurriyet.com.tr/saglik/20067686.asp
Trakya Üniversitesi Tıp Fakültesi Hastanesi Nefroloji Bilim Dalı öğretim üyesi Doç. Dr. Sedat Üstündağ, milli içecek haline gelen çayın kanser riskini azaltmasına karşın, sıcak içildiğinde mide hasarına ve kansızlığa neden olabileceğini söyledi. Koyu çayların demiri bağlayarak kansızlık yarattığını kaydeden Doç. Dr. Üstündağ, Sıcak çayın içerisinde demiri bağlayan bir takım maddeler var. Koyu çay çok içilirse, mide ve bağırsak sistemindeki demiri bağlar. Dolayısıyla anemi dediğimiz kansızlık hastalığının gelişmesini kolaylaştırır dedi. Türk Nefroloji Derneği Trakya sorumlusu da olan Doç. Dr. Sedat Üstündağ ve bazı dernek üyeleri, 8 Mart Dünya Böbrek Günü nedeniyle Trakya Birlik İlköğretim Okulu'ndaki öğrencileri ziyaret etti. Sınıfları gezerek bilgi veren Doç. Dr. Üstündağ, böbrek hastalarının her geçen gün arttığını belirterek, öğrencilere fazla tuz kullanmamalarını öğütledi. Tuzun adeta bir zehir olduğunu belirten Doç. Dr. Üstündağ şunları söyledi: Marketlerde alışveriş yaparken alacağınız ürünlerin sodyum (Na) oranına bakın. Sodyum oranı yüksek ürünler vücuda zarar verir. Tuz aslında zehir gibidir. Günlük tükettiğimiz besin maddelerinde tuz oranı var. Vücut, ihtiyacı olan tuz oranını bu besin maddelerinden alıyor. Üstüne bir de biz tuz kullandığımızda, başta böbreklerimiz olmak üzere birçok organımız hasara uğruyor. Dünya Sağlık Örgütü'ne göre bir insan günde 6 gram miktarında tuz tüketmeli, ancak ülkemizde bu oran 3 katı yüksekliğinde.
Ceylan (gazelle)
www.hurriyet.com.tr/seyahat/13179756.asp

Safari turlari için uçak hariç 2000-8000 Euro arasinda ücret ödeniyor. Tabii çok ultra lüks ve kisiye özel turlarla bu fiyatlar çok artiyor. Biz bu skalanin ortasinda, ancak epeyce konforlu bir tur seçtik. Ahsap çadirlarin bazilari 5 yildizli otelleri aratmiyordu. Lüks sayilabilecek kamp alanlarina lodge deniliyor. Bazi yemekler gurme restoranlarla yarisir. Kamp alanlarinda turistlerin kaldigi bölümler çok iyi aydinlatiliyor. Turumuzu özellikle çocuklu arkadaslarima anlata anlata bitiremedim. Ellerinde dürbünlerle hayvanlari izleyen çocuklari gördükçe içim gitti. Bu dünyanin bizden baska canlilarla ve hayatlarla dolu oldugunu anlatmanin daha iyi bir yolu olamaz. Safari boyunca kendinizi maceraperest saniyorsunuz. Zaman zaman kendimi bir belgeselin içinde buldum. Belgeselleri çekmek ve özel anlar yakalamak için ne kadar büyük bir çaba harcadiklarini daha iyi anladim. Turumuza Mayara Gölü'nden basladik. Burasi sodali bir göl, 400 kus türü yasiyor. Suyu çok yakici oldugu için insanlar giremiyor ama suaygirlarinin nesesine diyecek yok. Daha sonra sik sik karsilacagimiz bufalo, çita, antilop, benekli kirpi, akbaba, yabandomuzu, kertenkele, çakal, ceylan, zürafa, fil ve impalalari ilk kez burada gördük.
Ceylan (singer)
www.hurriyet.com.tr/ceylan-in-melodi-si-19207553
Türkücü Ceylan'ın ilk evliliğinden olan kızı Melodi Bozkurt, kendisine annesinden farklı bir kariyer yolu çizerek ‘iktisatçı' olmayı tercih etti. İlk sahne deneyimini 7 yaşındayken yaşayan ve sanat dünyasına çocuk yaşta girdiği için ‘Küçük Ceylan' olarak adlandırılan türkücü Ceylan'ın (37) ilk evliliğinden olan kızı Melodi Bozkurt'un fotoğrafları yıllar sonra ilk kez medyaya yansıdı. Vatan gazetesinden Zehra Çengil'in haberine göre Ceylan'ın ilk eşi Erhan Bozkurt'tan olan 21 yaşındaki Melodi'nin annesinden farklı bir yol izlediği ve "iktisatçı" olmaya karar verdiği öğrenildi. Samsun'da bulunan 19 Mayıs Üniversitesi İktisadi ve İdari Bilimler Fakültesi'nde okuyan Bozkurt, sosyal paylaşım sitesi twitter'daki hesabına ‘Geleceğin İktisatçısı' yazdı. Ceylan, ise kızıyla gurur duyduğunu belirterek "Kızım 19 Mayıs Üniversitesi İktisat bölümünü kazandığında çok sevindim. Devlet üniversitesi okumasından çok memnunum. Kendi hakkıyla ayaklarının üstünde durmaya çalışması bir anne olarak beni çok mutlu ediyor. Bu sene 1. sınıf öğrencisi. Müzisyen olmasından ziyade okumasını istedim. Her genç kız gibi altın bileziğinin elinde olması benim için önemliydi. Gelecekte borsayla ilgilenmeyi düşünüyor. Ekonomiyi yakından takip ediyor. Bölümüyle ilgili yüksek lisans hedefleri arasında. Onun büyüyüp bir iş kadını olmasını seyretmek heyecan verici" diye konuştu.
Duvar (wall)
www.hurriyet.com.tr/gundem/28194015.asp

Niğde Kalesi'ndeki mesire alanında kavga eden 2 genç, kale duvarından düşerek ağır yaralandı. Mesire alanına gelen bir grup genç henüz belirlenemeyen bir nedenle tartıştı. Tartışmanın büyümesi üzerine grupta bulunan Mustafa E. (19) ile Erkan D. (18) kavga etmeye başladı. Kale etrafını çevreleyen yaklaşık 10 metre yükseklikteki duvar üzerinde kavga eden gençler, dengelerini kaybedip önce park halindeki midibüsün üzerine ardından beton zemine düştü. 112 Acil Servis ekiplerince Niğde Devlet Hastanesine kaldırılan Mustafa E. ve Erkan D'nin sağlık durumlarının ciddiyetini koruduğu öğrenildi. Polis ekipleri olayla ilgili soruşturma başlattı. Görgü tanığı Yusuf Efe Er, İki kişi kucaklaşarak, birbirlerine vurarak, duvar kenarına geldi. Önce arabanın üzerine, sonra aşağı düştülerdedi.
Duvar (film)
www.hurriyet.com.tr/gundem/18460961.asp
Kültür ve Turizm Bakanı Ertuğrul Günay'ın talimatıyla hayata geçirilen çalışmada, Güney'in senaryosunu yazıp yönettiği Arkadaş, Umut, Aç Kurtlar, Duvar, Seyyithan, Ağıt ve Zavallılar ile senaryosunu yazdığı Yol, Düşman, Sürü ve Endişe DVD ortamına aktarıldı. Projeyle, bakanlık arşivinde bulunmayan 11 Güney filmi, bakanlık kayıtlarına girdi. Yaklaşık 1 yıldır proje üzerinde çalıştıklarını belirten Telif Hakları ve Sinema Genel Müdürü Çelik, şunları anlattı: Devletle barıştırma Bakanlık filmlerin DVD'ye aktarılmasında maddi ve manevi anlamda katkıda bulundu. Ayrıca filmler Bakanımız Ertuğrul Günay tarafından, yabancı heyetlere verilmek üzere Cumhurbaşkanlığı, Başbakanlık ve TBMM'ye de gönderildi. Yunus Emre Vakfı'nın yurtdışındaki bütün şubelerine de dağıtıldı. Çalışmanın Yılmaz Güney gibi bir sinemacıyla devleti barıştırma öyküsü olsun istedik.
Mısır (country)
www.hurriyet.com.tr/dunya/28343442.asp

BM Ortadoğu Barış Süreci Özel Koordinatörü Robert Serry, başta İsrail ve Mısır olmak üzere tüm taraflara başarısız olan Gazze politikalarını değiştirme çağrısı yaptı. Serry, Ortadoğu Barış Süreci Özel Koordinatörü sıfatıyla Gazze'ye son ziyaretini gerçekleştirdi. Serry, ziyaretin ardından yaptığı yazılı açıklamada, son yedi yılda Gazze'de 3 savaş yaşandığını ve sonuncusunun bölgeyi mahvettiğini kaydetti. Yıkımın ardından Gazze'nin yeniden inşası sürecinin başladığını dile getiren Serry, bunun için gerekli fonun sağlanamadığını ve Kahire Konferansı'nda söz verilen 5.4 milyar doların ancak küçük bir kısmının yerine getirildiğini belirterek bu durumun kabul edilemez olduğunu vurguladı. Gazze her zamankinden daha izole edilmiş bir durumda ifadelerini kullanan Serry, İsrail'den Gazze'ye geçişlerde halen büyük kısıtlamalar yaşandığını, Refah kapısının ise pratik olarak kapalı olduğunu bildirdi. BM olarak, Gazze'de sürdürülebilir bir yönetim, ekonomi ve istikrar için kuşatmanın kalkmasını hep savunduklarına işaret eden Serry, son ziyaretinde Gazze'deki muhataplarından da yer altında ve üstünde tüm askeri aktiviteleri bir kaç yıl dondurmalarını istediğini ve olumlu yanıtlar aldığını kaydetti.
Mısır (corn)
www.hurriyet.com.tr/ege/27354000.asp
MUĞLA'da, Gıda Tarım ve Hayvancılık İl Müdürlüğü tarafından 2014 yılında alternatif yem bitkisi üretim çalışmaları aralıksız sürüyor. Çalışmalar kapsamında dane sorgum, tirinova, caramba, caramba mix, yem bezelyesi, lenox, silajlık mısır, sorgum sudan otu melezi, yonca, hayvan pancarı, şalgam, fiğ, triticale, yulaf çeşitleri ve silaj kalitesinin arttırılması hedeflendi. Muğla Gıda Tarım ve Hayvancılık İl Müdürlüğü'nce Silaj Kalitesinin İyileştirilmesi Projesi kapsamında Menteşe İlçesi Doğanköy ve Yeniköy Mahalleleri'nde, topraksız mısır silajı yapımının gösterildiği Tarla Günü'ne ilçelerden ve çevre mahallelerden katılım büyük oldu. Üretici Şaban Başoğlan ile Saim Tezcan ın tarlalarında yapılan çalışmalar sonucunda, yüksek verimli mısır çeşitlerinin, silaj katkısı ve topraksız silaj örtüsü kullanımı ile silaj kalitesinin arttırılması ve bozulmalarının önüne geçilmesi için çiftçilere tanıtımı yapıldı. Tarla Günü'nde İl Müdürü Nazif Ekici, müdür yardımcısı Muhammed Sevinç, Bitkisel Üretim ve Bitki Sağlığı Şube Müdürü Resül Çoban, ilçe müdürleri, Ziraat Odası Başkanları, muhtarlar, teknik personel ile üreticiler katıldı.

Table B.2. A sample from general performance evaluation result of Thinker system.

Article URL
http://www.hurriyet.com.tr/yuksek-yargiya-yeni-duzen-geliyor-40116960
Entity Linking Result
 Yüksek yargıda bazı düzenlemeler içeren Danıştay Kanunu ile Bazı Kanununlarda Değişiklik Yapılmasına Dair Kanun Tasarısı TBMM Başkanlığına sevkedildi. Hükümetin TBMM'ye getirdiği tasarıya göre Yargıtay'da 46 olan daire sayısı 12'si ceza 12'si hukuk olmak üzere 24'e düşürülecek. Danıştay'ın daire sayısı ise 17'den 10'a inecek. Yargıtay ve Danıştay'daki mevcut üyelikler ise kademeli olarak düşürülecek. Yargıtay'ın 516 olan üye sayısı 200, Danıştay'ın 195 olan üye sayısı ise 90 olacak. Yargıtay uhdesinde bulunan dosyaları kendisi çözeceğinden, dosya sayısı kısa zamanda istenen seviyeye düşmeyeceği gerekçesiyle 22 daire Birinci Başkanlık Kurulu'nca en geç üç yıl içinde kademeli olarak kapatılacak. Bu daireler kapatılıncaya kadar görevlerine devam edecek.

Correct Mappings	Wrong Mappings	Undetected Entities
Birinci	Başkan	Başkanlık Kurulu
Bulunma	Daire	Danıştay Kanunu
Çozebilme	Düşme	Kanun
Danıştay_(Türkiye)	Geçme	Yüksek Yargı
Değişiklik	Içinde	
Devam	Mevcut	
Dosya	Üyelik	
Düşürülme		
Etme		
Gerekçe		
Getirme		
Görev		
Hükûmet		
Içermek		
Isteme		
Kademe		

Correct Mappings	Wrong Mappings	Undetected Entities
Kanun Tasarısı		
Kanun_(Hukuk)		
Kapatılma		
Kısa		
Sayı		
Seviye		
Tasarı		
Tbmm		
Uhde		
Üye		
Yargı		
Yargıtay_(Türkiye)		
Yüksek		
Zaman		