

University of West Bohemia
Faculty of Applied Sciences
Department of Computer Science and Engineering

Master Thesis

Visualization of Cost Data

Pilsen, 2013

Jan Rybák

Abstract

Our society became very information oriented in the recent two decades and every year the amount of digital data more than doubles. However, our ability to effectively process the data and extract useful and relevant information is limited.

In the scope of this thesis we are addressing two challenges, automated information extraction and its comprehensible visualization of numerical data. The first part is concerned with identification of possibly interesting financial data from newspaper articles and their extraction. We are proposing a new approach to recognize named entities using DBpedia, we are identifying financial relations and extracting monetary data. Next, to communicate obtained information clearly, a tool for visualizing the financial cost of recognized entities is introduced. As we developed an advanced system, which helps people to reinforce their finance-related knowledge, we believe that the thesis's objective was met.

Acknowledgements

This thesis would not have been possible without the support of many people. I would like to express my sincere thanks to Kjetil, Kristýna, Krisztian, Richard and Tárík. Finally, I am forever indebted to my parents for their endless love and support.

I hereby declare that this master thesis is completely my own work and that I used only the cited sources.

Pilsen, May 7, 2013

Jan Rybák,

Contents

Abstract	i
Acknowledgements	ii
Contents	iii
1 Introduction	1
1.1 Background	1
1.2 Motivation	2
1.3 Project Goals	3
1.4 Data Acquisition	4
1.5 Communicating information	4
1.6 Process	5
I Information Extraction	6
2 State-of-the-Art	7
2.1 Introduction	7
2.2 Historical Background	7
2.3 Definitions of Information Extraction	8
2.3.1 Information Extraction or Document Retrieval	8
2.4 Information Extraction Approaches	9
2.4.1 Information Units in IE	9
2.4.2 Division of IE Techniques	10
2.4.3 Rule-Based Methods	10
2.4.4 Probabilistic Approach	12
2.5 Named Entity Recognition	14
2.6 Relation Detection	15
	iii

2.6.1	Challenges	15
2.6.2	Semantic Role Labeling	16
2.7	Pre-processing	19
2.7.1	Tokenization	19
2.7.2	Sentence Segmentation	19
2.7.3	Part of Speech Tagging	20
2.8	Ontology	21
2.9	Linked Data	23
2.9.1	RDF	23
2.9.2	SPARQL	24
2.9.3	DBpedia	24
3	Realization	25
3.1	Goal of the Application	25
3.2	Technology	26
3.2.1	Programming Language and Development Environment	26
3.2.2	Overview of NLP Frameworks	26
3.3	Architecture	28
3.3.1	Pipeline Architecture	28
3.4	Data	30
3.4.1	New York Times corpus	30
3.4.2	Twitter Data	32
3.5	Monetary Information Recognition	33
3.5.1	Number Recognition	34
3.5.2	Currency Recognition	36
3.6	Relation Extraction	39
3.6.1	Ontology	39
3.7	Originator Recognition	42
3.7.1	DBpedia	42
3.7.2	Noun Phrase Chunking	48
3.7.3	Part of Speech Tagging (POS)	49
3.7.4	Extension of Noun Phrases using Regular Expressions	49
3.8	Information Identification and Extraction	51
3.8.1	Semantic Role Labeling	51
3.8.2	Extraction of Related Triples	53
3.9	Information Storage	55

3.9.1	RDF Triples	55
3.9.2	RDF Storage	56
3.9.3	RDF Triple uploader	56
4	Evaluation	58
4.1	Evaluation Data	58
4.2	Evaluation Metrics	58
4.3	Evaluation results	59
4.3.1	Precision	59
4.3.2	Recall	60
4.4	Evaluation Interface	61
II	Information Visualization	62
5	Background	63
5.1	Introduction	63
5.2	History	63
6	Visualization Methods	66
6.1	Process	67
6.2	Treemap	68
6.2.1	Layout Algorithms	68
7	Realization	72
7.1	Architecture	72
7.2	Technology	73
7.2.1	SPARQL	73
7.2.2	Data-Driven Documents (D3.js)	74
7.3	Interface	75
7.4	Data Selection	75
7.5	Visualization Methods	78
7.6	Interactivity	80
7.7	Data Format and Export	82
8	Conclusion and Future Work	84
8.1	Conclusion	84
8.2	Future Work	85

8.2.1	Information Extraction part	85
8.2.2	Visualization	85
List of Figures		87
List of Tables		89
A	Extraction XML File	90
B	User Manual	92
B.1	Visue Extraction Tool	92
B.2	Sesame RDF Uploader	93
B.3	Visualization	94
B.4	RDF storage	94
C	Digital Attachments	95
Bibliography		96

Chapter 1

Introduction

1.1 Background

During the recent decades our society became information-oriented. Enormous amount of data is being produced every single second. The amount is so huge that we are not able to process it all and use it effectively. In 2010 the size of the Internet was estimated to be approximately 500 billion gigabytes. To give you an idea, that is the equivalent of a stack of books stretching from the Earth to Pluto ten times¹. But not all the information is uploaded to the Internet. According to Bounie and Gille [6] the world generated 14.7 exabytes of new information in 2008, nearly triple the volume of information in 2003.

"We remember what we understand; we understand only what we pay attention to; we pay attention to what we want." - Edward Bolles -

For us, information consumers, the most difficult task is no longer to acquire the data, but to filter out all the irrelevant and unclear data that we don't need. In such situation it is very easy to lose our objectivity and general perspective. This phenomenon is called 'information overload' or 'info obesity' and the main causes are ²:

- a rapidly increasing rate of new information being produced,
- the ease of duplication and transmission of data across the Internet,
- an increase in the available channels of incoming information.

The major implications and challenges that the information overload brings are:

- contradictions and inaccuracies in available information,

¹<http://www.guardian.co.uk/business/2009/may/18/digital-content-expansion>

²Information overload: http://en.wikipedia.org/wiki/Information_overload

- a lack of methods for comparing and processing various kinds of information,
- the fact that the pieces of information are unrelated or do not have any overall structure to reveal their relationships.

Particularly, the last two points in the list above are crucial for this thesis. If there is no way to compare various pieces of information, it is impossible to realize its context. And without any context the information is meaningless.

Most of us are absorbing a huge amount of information every single day, it does not matter if the sources of it are newspapers, Internet, television or something else. We are used to hear news mentioning big financial numbers like billions or trillions of dollars but only a handful of us can really imagine such enormous sums of money. This is mostly caused by the fact that we are not dealing with that kind of money on a daily basis. It is difficult for us to relate quickly, for example, state's budget to the price of a new car. And even if we are masters in comparing numbers, there is a second aspect associated with big numbers - their memorability. Unfortunately due to their abstract nature, numbers are one of the hardest things to remember. After some time it is not that easy to remember correctly at least the order of magnitude of big sums.

It is nearly impossible to focus only on the important information that reaches our senses, filter out all the noise, and remember everything. On the other hand this is the only way how to gain undistorted knowledge. In the thesis we are proposing a solution how to extract financial data, let user pick only the important and relevant pieces of information and put them into comparison. The goal of our tool is to motivate users to explore seemingly unrelated financial data and see "the big picture" in it.

1.2 Motivation

The first impulse that made us think about the comparison of financial data from various domains was a talk called 'The beauty of data visualization' given by David McCandless at TED conference³ in 2010. He presents a treemap-style chart called 'The \$Billion Dollar o-Gram'⁴ where he compares various costs, subjects of these costs and motivation behind it. Even though the data he shows are publicly known, it is the comparison that makes all the magic. When you can visually compare different costs, big numbers are becoming less abstract.

³http://www.ted.com/talks/david_mccandless_the_beauty_of_data_visualization.html

⁴<http://www.informationisbeautiful.net/visualizations/the-billion-dollar-o-gram-2009/>

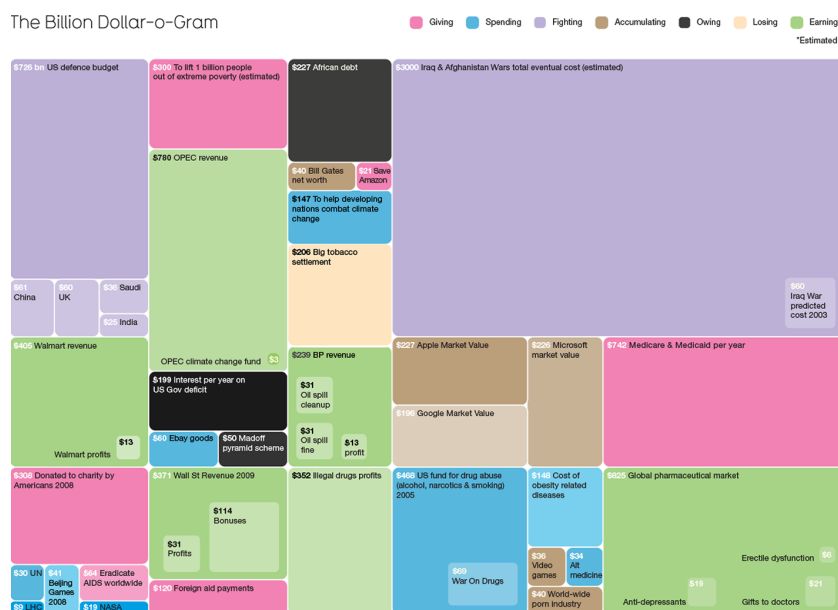


Figure 1.1: Billion Dollar o-Gram by David McCandless

David McCandless scraped all the data for his visualization manually. We are aware of the big potential in retrieving the data automatically or semiautomatically by means of natural language processing. Such an approach might bring new possibilities like manipulation with a vast amount of input data, interactivity, inclusion of temporal aspect of the data, etc.

1.3 Project Goals

In this thesis we would like to develop a system that will be able to analyze plain text written in natural language, to extract financial data from the unstructured sources and visualize them in the most appropriate form for the user. The application should help people to access financial data, to turn it into information and most importantly to offer a tool to gain knowledge from the information. The term *financial data* means a triplet of data in the following format:

SUBJECT + RELATION + COST

E.g.: Google Inc. + earned + \$ 37,000,000,000 USD

The subject, in other words the originator of financial information, might be e.g. a company, a person, a project or something else which is, somehow, connected with the cost. The interesting feature of the cost is its relation to the subject. To mention just a few examples, it can be revenue, loss, debt, expenses, share, etc.

1.4 Data Acquisition

Gathering the data is the first and foremost task which all the following steps directly depend on. To bring an interesting story to light, we first need relevant data. As mentioned in the introduction, the Internet offers nearly inexhaustible source of information, however, there is an disadvantage in its heterogeneity. This is the main reason why we have decided to extract the information from texts written in natural language. There is a whole field of science occupied with extracting useful information from large data sets or databases formally known as data mining [17].

Even though the system should be able to process any kind of plain text, we think that news articles are very suitable source of data. As we aim to develop an application for ordinary people, it is desirable to work with popular information channels such as The New York Times newspaper⁵ or similar.

1.5 Communicating information

"The visualization of information gains its importance for its ability to help us 'see' things not previously understood in abstract data." - Ben Fry -



Figure 1.2: Mock-up of our visualization

Once we turned the data into information (data triplets), all the effort will be concentrated on comprehensible communication of the results to the user. Even if people are given the right figures, it doesn't necessarily mean that they can realize their meaning. It is the comparison and the relation to the context that play the important role.

Information visualization is a flourishing discipline that has become very popular in the recent years [9]. Graphical presentation of the data shows relationships much more

⁵<http://www.nytimes.com/>

clearly, the user can find patterns in data very quickly and it is also a more attractive form of data presentation than plain tables.

James J. Thomas and Kristin A. Cook stated the following about information visualization: *"Visual representations and interaction techniques take advantage of the human eye's broad bandwidth pathway into the mind to allow users to see, explore, and understand large amounts of information at once. Information visualization focused on the creation of approaches for conveying abstract information in intuitive ways."*[47]

1.6 Process

This thesis is composed of two main parts - information extraction and information visualization. In this section we would like to briefly describe individual steps that are covered by these comprehensive terms.

1. Data acquisition

Feeding the system with input data - English plain text.

2. Filtering

Selection of relevant articles, i.e. those which contain only financial information (number+currency).

3. Subject extraction

Finding the originator of the financial information.

4. Relation extraction

Identification of the relation between the subject and the cost.

5. Storing

Storing the data triplets (Subject + Relation + Cost).

6. Correction / Selection

Verification of the output by a human and selection of the data to be presented.

7. Visualization

Graphical representation of the result.

8. Interaction

User interaction with the presented results (manipulation, additional information, ...).

Part I

Information Extraction

Chapter 2

State-of-the-Art

Before we start describing the process of realization, we will give a fundamental overview in the field of natural language processing while emphasizing the information extraction. In this chapter, we also provide description of Linked Data publication concept which is closely related to the way we store and visualize extracted data.

2.1 Introduction

Information extraction (IE) is a specific discipline of natural language processing (NLP), a subfield of artificial intelligence, which has been a rich subject of research. IE systems are used for transforming unstructured information (such as Web) into machine-readable structures (such as relational database). The main goal of IE is to extract predefined types of data and relations from a single document. Typical types of entities that we are searching for include names of organizations, people or locations, but generally IE does not necessarily have to be domainspecific [52]. Recognition of desired information is a non-trivial task and significant number of techniques have been developed over the last few decades.

2.2 Historical Background

The serious beginnings of IE are strongly related to the series of Message Understanding Conferences (MUCs) held from 1987 to 1998. The main task of these competition-oriented conferences was to fill predefined templates with very specific pieces of information such as details about terrorist events in Latin America. These information bits

were, after recognition, filled into exactly fitting slots of a prepared template (e.g. perpetrator, victim, date, location, etc.)[21].

Conferences such as MUCs set pattern matching approach of IE as a trend of the time. Basic, but very important concepts such as IE system evaluation techniques (recall, precision), named entity, word sense disambiguation, coreference, and many more, were introduced due to these conferences.

Another important challenge which influenced IE research was Automatic Content Extraction program [29]. The main objective of the program was identification of entities, relations, and the events in the natural language.

2.3 Definitions of Information Extraction

As the information extraction has been heavily researched for a long time, many definitions of the actual term 'Information Extraction' appeared in various papers. Formulations of these definitions are influenced mostly by the state-of-art techniques of the time. We can see the evolution in this particular field just by reading the following definitions.

"IE systems extract domain-specific information from natural language text. The domain and types of information to be extracted must be defined in advance. IE systems often focus on object identification, such as references to people, places, companies, and physical objects. [...] Domain-specific extraction patterns (or something similar) are used to identify relevant information." [38] - Riloff and Lorenzen -

"[...] isolates relevant text fragments, extracts relevant information from the fragments, and then pieces together the targeted information in a coherent framework. [...] The goal of information extraction research is to build systems that find and link relevant information while ignoring extraneous and irrelevant information." [13] - Cowie and Lehnert -

"Information extraction is the identification, and consequent or concurrent classification and structuring into semantic classes, of specific information found in unstructured data sources, such as natural language text, making the information more suitable for information processing tasks." [28] - Moens -

2.3.1 Information Extraction or Document Retrieval

The terms Information Extraction (IE) and Information Retrieval (IR) or even Document Retrieval (DR) are often confused. IR is the activity of identifying and obtaining possibly

relevant documents containing desired information from large document collections such as Web or a database. Document Retrieval is a specialized type of IR which is concerned with retrieval mainly from text-based documents. We can describe current Web search engines as Document Retrieval systems.

In their book [39], Russell and Norvig expressed the difference between information extraction and retrieval by the following sentence:

"Information extraction systems are mid-way between information retrieval systems and full-text parsers, in that they need to do more than consider a document as a bag of words, but less than completely analyze every sentence." [39]

2.4 Information Extraction Approaches

In this section we give thorough introduction to the main IE approaches. A nice survey of Web-focused IE systems is given in [8].

2.4.1 Information Units in IE

The first target elements of IE in the time of MUCs were names of people, organizations and locations together with temporal and numeric entries. Such mentions in a text are typically described as *entities* and, as expressed in [20], may take form of

- **names** ("Tim Berners-Lee"),
- **noun phrases headed by common nouns** ("the queen"),
- and **pronouns** ("it").

The task of *named entity recognition* is covered in Section 2.5.

The next step in IE is the recognition of relationship which is defined over two or more entities. Example of such relationship between two entities can be "is the father of" which is expressing the family relationship between a father and his son. Section 2.6 takes a closer look at Relation Extraction (RE) techniques. Higher-order structures such as *ontologies* or tables are also being extracted. Since we are using an ontology both for NER and RE purposes, there is Section 2.8 related to this topic.

2.4.2 Division of IE Techniques

According to Sarawagi [42] we can divide IE techniques into different groups by aspects stated below.

Hand-coded vs. Learning-based

The most straightforward approach for IE system implementation is to use handwritten rules. These rules used to extract a specific kind of information are usually composed by a domain expert (human) in the form of regular expressions. It is necessary that the person, who is preparing extraction rules, has a good knowledge of the IE system itself, so he or she is able to create robust rules. The preparation of such collection of rules is an expensive and tedious process, particularly for broad range domains. Knowledge-engineering approach was adopted mostly in the early IE systems because of its easier implementation.

More recent systems take advantage of machine learning model with limited amount of human input needed (supervised learning). Domain specific parts of training text must be manually annotated by a domain expert. Then, these data serve as a seed for training algorithm and the result is used to extract specific information from different data sets. We can see the process of learning as an attempt to obtain the rules automatically from the text itself. The majority of data is changing dynamically in time and retraining IE system seems to be easier than managing rule-based system up to date.

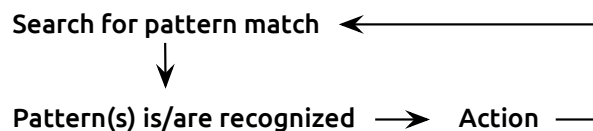
Rule-based vs. Probabilistic

Another aspect of IE systems division is whether extraction method is based on hard predicates (rules) or statistical analysis. Statistical learning methods are more robust to noise and variation in unstructured text [10]. Its advantage is also considerably lower need for human interaction when developing IE system for broad-range domains.

2.4.3 Rule-Based Methods

Rule-based approach, sometimes called *Language Engineering*, is a model relying on the theory of grammars and finite-state automata. In the rule-based systems we have to specify a pattern (i.e. rule, template) for every kind of information we want to extract. In the simplest cases, these patterns can be described by regular expressions, in more advanced cases some other tools might be used (e.g. JAPE rule language in GATE).

Behavior of rule-based systems is driven by the following formula:



Patterns can be recognized by lexical features of the matching token itself or by its context. Features of the given token might be for example:

- Actual string of the token.
- Part-of-speech category of the token or other specific feature of the token.
- Annotations from previous steps of NLP (e.g. category assigned by Gazetteers, NER).

Listing 2.1: Pattern example in JAPE format (using POS of the token)

```

Rule: AdjectiveTagger
(
  {Token.category == "JJ"}
) : Adjective
  
```

If it is the context that identifies the token, than we are usually using a prefix and/or suffix to describe the target (see Listing 2.2).

Listing 2.2: Pattern example in JAPE format (using currency suffix to identify the token)

```

Rule: MyCurrencyCategory
Priority: 100
(
  {Number}
  ({SpaceToken})*
  ( {Lookup.majorType == "currency_unit",
    Lookup.minorType == "post_amount"}
  ):cur
): label
-->
:label.MONEY= {type="Money", value= :label.Number.value, currency= :
  cur@string}
  
```

Information Extraction systems usually comprise of many patterns and it may happen that more than one pattern fits the rule, such situation is called a conflict. That is why we

need to define some policy that chooses the most suitable pattern in case of a conflict. Classical approaches to resolve conflicts are:

- **Implicit** - e.g., longest matching pattern
- **Explicit** - assigning each pattern an explicit priority (Example: Listing 2.3)

These methods can be combined for the case of two patterns covering text of the same length; then the rule with higher priority is then applied.

Listing 2.3: Use of priority attribute.

```
Rule: PrefixCurrencyTagger
Priority: 100
```

Pattern Learning

Manual creation of patterns is a tedious and error-prone process, not speaking of the need for domain experts. This situation has encouraged many researchers to explore the problem of pattern learning methods. This specific application of *Machine Learning* usually requires input in the form of manually annotated data which is called a *training data set*. In the following section we describe the probabilistic approach to IE which is the core theory behind the pattern learning.

2.4.4 Probabilistic Approach

In case we need to extract information from noisy or not well-structured data, the rule-based systems do not perform very well. The main disadvantage of the finite-state systems is their inability to capture more complex language patterns in an easy way. Probabilistic approach is seeking *language models*, i.e., models describing the probability distribution of language expressions (e.g. words).

Classifiers

The classification problem is addressing the task of deciding if a data item $x \in X$ on the input belongs to one of predefined classes $c \in C$. For example, given the word "perfectly" we have to tell to which syntactic category in $C = \{adjective, noun, verb, \dots\}$ it belongs with the highest probability. We can also see the process of classification as assigning labels of different classes to data provided on the input. The unit which performs this labeling is denoted as the *classifier*. We distinguish *binary* and *n-ary classifiers*, depending on how many labels they are working with.

A trainable classifier is able to learn classification patterns and operate on any input, including previously unseen data. In NLP a technique called *supervised learning* is very popular. As the name suggests, the training algorithm requires manually labeled data for learning.

A typical classification task depends on a set of *features* which are describing the data item. For word-like units called *tokens* such features can be organized in the following groups:

- **Lexical Features** - e.g., word's surface form, etc.
- **Part-of-Speech Features** - lexical category of the word
- **Orthographic Features** - such as its capitalization form, the presence of special symbol, etc.
- **Dictionary Lookup Features** - if the word matches with a word in dictionary.

Learning Models

Many different learning algorithms are available for classifiers. The main difference among them is whether the input values are classified as conditionally independent. One of the most widely used models in NLP which is based on conditional probability, is the **Maximum Entropy Model**. The probability of assigning label c to data item x is given by

$$p(c|x) = \frac{1}{Z} \exp \sum_{j=0}^N w_j h_j \quad (2.1)$$

where Z is a normalizing constant, h_j is the value of i^{th} *indicator function*, and w_j is the weight assigned to indicator function j by the training process. Each indicator function is a combination of a particular class label c and a particular feature of data item x . [20]

Other popular models, in detail explained in [24], are

- **Naïve Bayes Model** - input values are considered to be conditionally independent in this model. It is modeling the joint probability $p(c, \vec{x})$ of the input values \vec{x} and the class variable c .
- **Hidden Markov Models** - HMM is extension to NBM for sequential data.

- **Conditional Random Fields** - discriminative (conditional) model suitable for sequential data. It builds on the top of Maximum Entropy Model and also models the conditional probability $p(c|x)$.

2.5 Named Entity Recognition

Named entity recognition (NER) is the task of detection and classification of *proper nouns* such as names of people, organizations or locations. During the MUCs [1], it became clear that the ability to identify names is essential for information extraction. Nowadays, NER is an essential part for many NLP systems and from the initial set of 3 defined named entities introduced by MUC-6 [1] it has grown up to couple of hundreds [45].

Example

Johnie Walker was named CTO of IBM Inc. in 1999. Two years later he started working for Yahoo.

↓ *annotation*

[Johnie Walker]^{person} was named CTO of [IBM Inc.]^{org} in 1999. Two years later he started working for [Yahoo]^{org}.

NER Approaches

- List Lookup Approach (Gazetteers)
- Supervised Methods
- Semi-supervised Methods
- Active Learning

2.6 Relation Detection

The aim of our system and many others is, besides other things, to detect semantic relation between extracted entities because the relationship may not be known in advance. The ability to identify relations in NLP is an essential step for language understanding. Let us first define the term *relation*:

Definition of relation:

Let X and Y be nonempty sets. A binary relation (or just relation) from X to Y is a subset

$$R \subseteq X \times Y.$$

If $(x,y) \in R$, we say that x is R -related to y , denoted xRy .

Expressed by basic language, a binary relation in NLP context can be seen as an information describing some action between two entities. We can describe such relation by *triples* - relation type and two arguments.

Most of the systems focus on binary relations, however, there are some which are addressing the problem of n -ary relations. One of the proposed methods to detect n -ary relations is to decompose them into a set of binary relations [25].

Example of relations:

Company A was acquired by B. \rightarrow *acquiredBy(Company A, B)*

Person X paid \$40. \rightarrow *paid(Person X, \$40)*

John is from Prague. \rightarrow *IsFrom(John, Prague)*

"Whereas entities refer to a sequence of words in the source and can be expressed as annotations on the source, relationships are not annotations on a subset of words. Instead they express the associations between two separate text snippets representing the entities."[42]

2.6.1 Challenges

Sunita Sarawagi is describing two scenarios of relationship extraction:

- predicting the relationship between a given entity pair,
- extracting entity pairs given a relationship type,

and in his paper [42] gives thorough overview of various RE approaches including *Feature-based Methods* and *Kernel-based Methods*.

2.6.2 Semantic Role Labeling

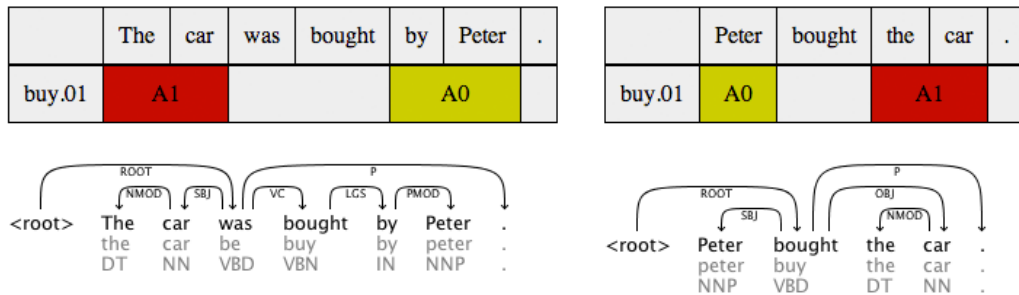


Figure 2.1: SRL representation of two sentences.

Semantic Role Labeling (SRL) is addressing the problem of automatic detection of *semantic roles* (see Section 2.6.2). Its goal is to identify semantic arguments of the predicate and assign roles that indicate for example the *Agent*, *Patient*, *Instrument*, or more specific roles such as the *Buyer*, *Seller*, etc. SRL is a very useful method for information extraction systems because of its ability to extract semantic roles independently on the sentence structure. For instance, these two sentences are equal in terms of semantic roles:

The car was bought by Peter.

Peter bought the car.

Semantic Roles

Semantic roles (also known as thematic roles) describe the relationship between the main verb in a clause and its arguments (participants). There is an important difference between semantic roles (agent, instrument, etc.) and grammatical relations (subject, object, etc.). While the grammatical relations are relations between words in a sentence, the semantic roles describe the actual relations that the participants play in a situation described by the sentence.

Table 2.1: Difference between grammatical relations and semantic roles.

<i>Sentence</i>	<i>Subject</i>	<i>Object</i>	<i>Agent</i>	<i>Recipient</i>
Google bought YouTube for \$1.65 billion.	Google	Youtube	Google	Youtube
YouTube was bought by Google for \$1.65 billion.	Youtube	Google	Google	Youtube

Typical semantic roles¹

- **Agent** The 'doer' or instigator of the action denoted by the predicate.
- **Patient** The 'undergoer' of the action or event denoted by the predicate.
- **Theme** The entity that is moved by the action or event denoted by the predicate.
- **Experiencer** The living entity that experiences the action or event denoted by the predicate.
- **Goal** The location or entity in the direction of which something moves.
- **Benefactive** The entity that benefits from the action or event denoted by the predicate.
- **Source** The location or entity from which something moves.
- **Instrument** The medium by which the action or event denoted by the predicate is carried out.
- **Locative** The specification of the place where the action or event denoted by the predicate is situated.

Process of Semantic Role Labeling

Semantic Role Labeling is the process of annotating sentence constituents with their semantic roles. Semantic roles are usually annotated with labels provided by a role set (for example see Fig. 2.2).

Frame for *change*, role set 1:

*arg*₀: the entity that initiates the change

*arg*₁: the entity that is changed

*arg*₂: the initial state of the changed entity

*arg*₃: the final state of the changed entity

Figure 2.2: Example of a role set from NomBank.

¹Source: <http://elies.rediris.es/elies11/cap5111.htm>

Example sentence:

Google bought YouTube in October for \$1.65 billion.

Example annotation:

[_{A0} Google] [_{Predicate} bought] [_{A1} YouTube] [_{AM-TMP} in October] [_{A3} for \$1.65 billion].

PropBank

PropBank² is a corpus in which the arguments of each predicate are annotated with their semantic roles in relation to the predicate [32]. Argument structure of every single verb is described in the so-called *semantic frame*. One particular predicate can have distinct meanings in different contexts. PropBank differentiate predicates with the same surface form but different semantic role by suffixing the predicate name with a unique number.

List of all semantic frames in PropBank can be found at <http://verbs.colorado.edu/propbank/framesets-english/>.

Predicate: *acquire*

Roleset id: *acquire.01*, *get*, *acquire*, **vncls:** [13.5.2-1](#), **framnet:** Getting

acquire.01: Based on big corpus, comparison with 'gain' and 'buy'. Member of Vncls obtain-13.5.2-1.

Roles:

Arg0: *agent, entity acquiring something* (vnrole: 13.5.2-1-Agent)

Arg1: *thing acquired* (vnrole: 13.5.2-1-Theme)

Arg2: *seller* (vnrole: 13.5.2-1-Source)

Arg3: *price paid*

Arg4: *benefactive*

Figure 2.3: Example of an entry in PropBank for predicate *acquire.01*.

²<http://verbs.colorado.edu/propbank/>

2.7 Pre-processing

The result of information extraction crucially depends on several operations which are usually described as pre-processing steps. Considering that our application's input is raw text in natural language, there is a need to recognize individual linguistic units such as words and sentences, to annotate named entities and filter out only the relevant content for further phases of IE.

Each part of pre-processing has to be as reliable and precise as possible because errors could cascade since the earliest stage and degrade overall results.

2.7.1 Tokenization

Tokenization is a process of splitting up input text into so called tokens. Usually these tokens denote words, numbers, symbols (currency) and punctuation. Tokenization can be also referred to as word segmentation.

Difficulty of tokenization depends on analyzed language. To give an example, European languages (e.g. English, Norwegian, Czech, etc.) have clear word boundaries given by space mark delimiters and therefore the task of segmentation is not as difficult as for some other languages, Chinese for instance [49].

2.7.2 Sentence Segmentation

Sentence segmentation can be described as a task of sentence boundary detection. In the majority of languages sentences are divided by punctuation marks. This observation can lead to the premature idea that plain identification of these marks is a sufficient technique of sentence boundaries recognition. In fact, the problem of sentence segmentation is much more complex. Exactly the same marks denoting the end of a sentence could be part of, for example, abbreviation or name (13 a.m., Prof.,). According to Liberman and Church [11], 47 % of periods in the Wall Street Journal delimit abbreviations.

Comprehensive overview of tokenization and segmentation problems is given by David D. Palmer in *Text Pre-processing* chapter of *Handbook of Natural Language Processing* book [31].

2.7.3 Part of Speech Tagging

Words exist in many different syntactic variations. Part of speech tagger assigns a grammatical category to every single processed word. A fixed set of part-of-speech tags (such as noun, verb, adjective, etc.) was established in well-known Brown and Penn treebank corpora. These part-of-speech tag sets contain conventional tags, such as "verb", as well as more detailed variations capturing, for example, different tenses of verbs (see Table 2.2). For information, the Brown tag set contains 179 tags.

Table 2.2: Example of Brown corpus part-of-speech tags [2].

<i>Part-of-speech</i>	<i>Tag</i>	<i>Part-of-speech</i>	<i>Tag</i>
adjective	ADJ	pronoun	PRO
adverb	ADV	preposition	P
conjunction	CNJ	the word <i>to</i>	TO
determiner	DET	interjection	UH
existential	EX	verb	V
modal verb	MOD	past tense	VD
noun	N	present participle	VG
proper noun	NP	past participle	VN
number	NUM	wh determiner	WH

2.8 Ontology

An ontology has been described by Noy & McGuinness [30] with the following words:

"Ontology is a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them."

As mentioned above, ontologies are usually domain-specific. Pieces of information are organized into categories and subcategories (classes). This taxonomic hierarchy should follow the real concept of things and should capture relations among them. Objects are usually expressed by nouns, relations by verbs.

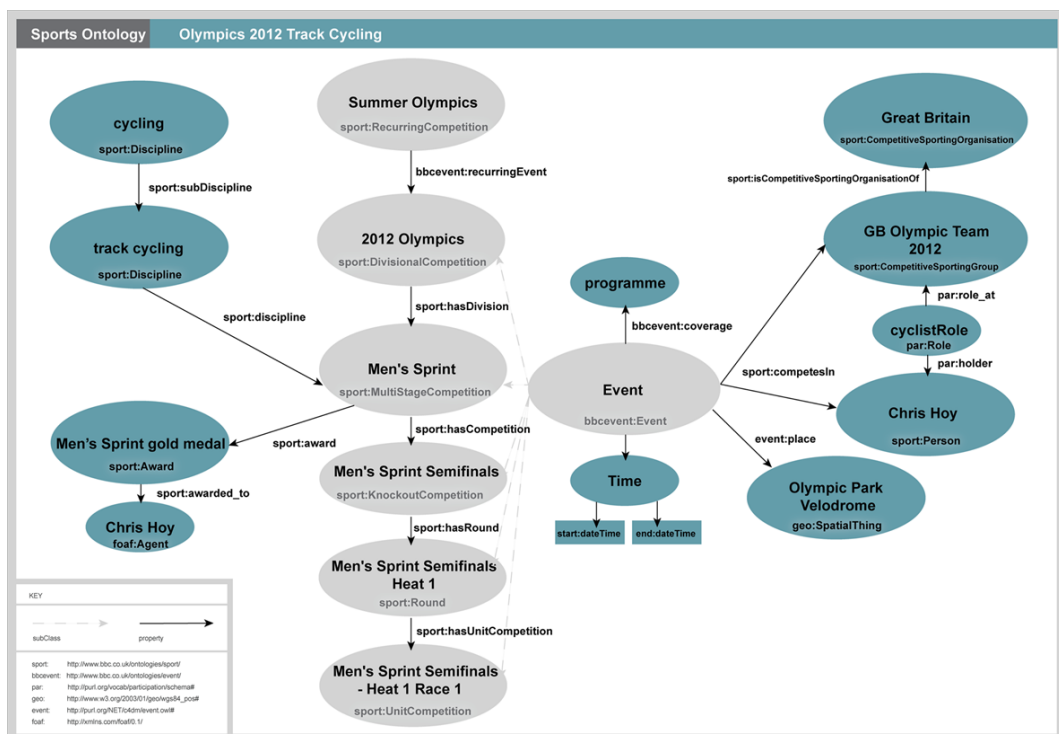


Figure 2.4: Example of an ontology - BBC Sports Ontology

Classes

Classes describe groups of items with the same qualities.

Development Approaches

Ontology can be visualized as a graph where the root node is the most general class and leaves are the most specific items. According to [30], there are three approaches in defining the class hierarchy:

Top-Down development

First we try to define the most general concepts, such as 'Car' if we want to create an ontology for cars.

Bottom-Up development

We start from the very specific class instances, such as 'BMW E61 M5 Touring' and later we create a superclass that covers several leaves in our graph, e.g. 'BMW', 'European cars', etc.

Combination of these two approaches

We are generalizing and specializing on the fly. First we create the most important classes and later we add up their instances or superclasses.

Ontology-based information extraction

Recently, new methods of information extraction using ontologies [50] have emerged and they are gaining even more attention nowadays when huge knowledge bases such as DBpedia or FactForge³ exists. The approach called Ontology-based information extraction (OBIE) makes use of ontologies for the document tagging process. Definition of OBIE given by Wimalasuriya and Dou [50] states the following sentence: *"A system that processes unstructured or semi-structured natural language text through a mechanism guided by ontologies to extract certain types of information and presents the output using ontologies."*

There are OBIE systems generating the ontology autonomously [51]. The ontology to be used in our system should be defined or at least revised manually by domain experts to ensure its precision.

³<http://ldsr.ontotext.com/>

2.9 Linked Data

Linked Data is a concept whose objective is to interconnect related data published on the Web. In [3] this term was described with the following words: “*Linked Data is simply about using the Web to create typed links between data from different sources*”. This approach brings more possibilities to the current Web by providing data with URI identifiers, which makes them accessible and sharable. Linked Data structure can be seen as a graph. Each information is represented by RDF statement described below.

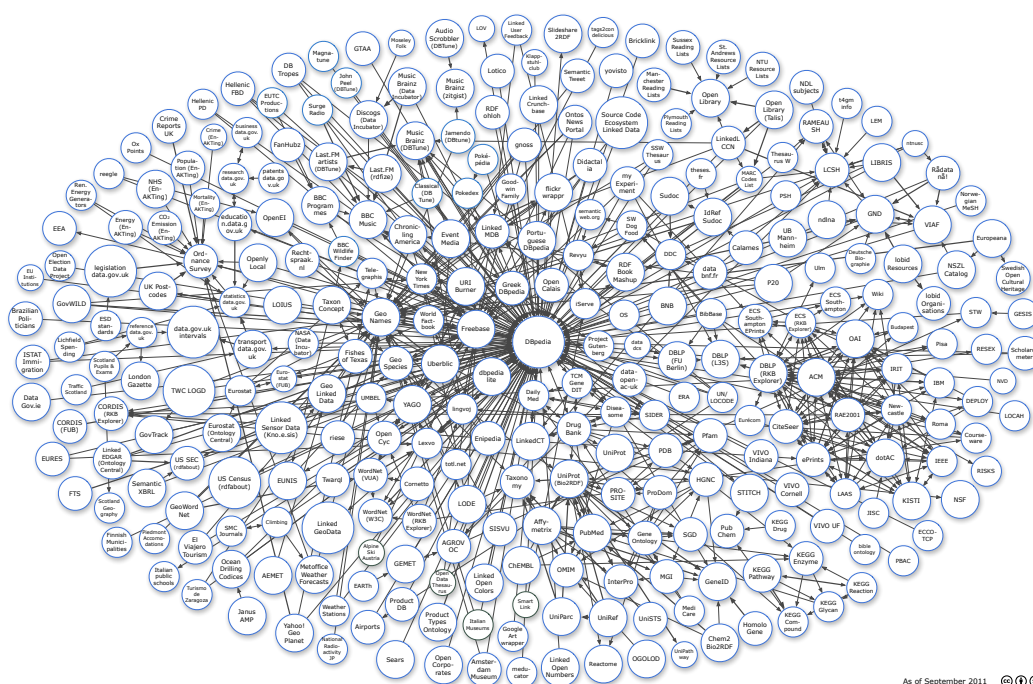


Figure 2.5: The Linking Open Data cloud diagram - datasets that have been published in Linked Data format.

2.9.1 RDF

Resource Description Framework⁴ (RDF) is a model representing data as a graph. It was specified by W3C⁵. As was already mentioned, RDF consists of statements. Each statement is defined by a triple: *subject*, *predicate* and *object* [3]. Entities are represented as nodes in the graph and relationships between them as edges [23].

⁴<http://www.w3.org/RDF/>

⁵World Wide Web Consortium: <http://www.w3.org/>

2.9.2 SPARQL

SPARQL (Simple Protocol And RDF Query Language) is a query language designed to manipulate data in RDF storages. In the W3C documentation for SPARQL is written: *“Most forms of SPARQL query contain a set of triple patterns called a basic graph pattern. Triple patterns are like RDF triples except that each of the subject, predicate and object may be a variable. A basic graph pattern matches a subgraph of the RDF data when RDF terms from that subgraph may be substituted for the variables and the result is RDF graph equivalent to the subgraph.”* [34].

2.9.3 DBpedia

“DBpedia is one of the more famous pieces of it (Linked Data Project).”[33] - Sir Tim Berners-Lee -

It is a knowledge base extracted out of Wikipedia and converted into structured, machine-readable format (RDF⁶). The DBpedia data set currently describes over 3.7 million items (in English) including more than 400 000 persons, 500 000 places and 170 000 organizations [27].

⁶Resource Description Framework: <http://www.w3.org/RDF/>

Chapter 3

Realization

In this chapter we will describe the realization part of the information extraction phase. We will present our ideas, suggested approaches and important implementation details. We also provide overview of the technology that we have used and architecture of the whole system. Our system was given a name - *VISUE* which is further divided into *VISUE Extraction* and *VISUE Visualization*.

3.1 Goal of the Application

The goal of the extraction phase can be summarized as follows: The application should accept files containing text in natural language (English). It should identify information containing utterance about money (with some numerical value) and should try to extract this information including the subject of the financial cost. Of course, when the information is extracted, there is a natural need to store acquired information in a convenient way, therefore this should be also considered.

The second part of the thesis is dedicated to visualization of data obtained in the extraction phase.

3.2 Technology

3.2.1 Programming Language and Development Environment

The selection of the most suitable programming language was significantly influenced by the choice of a toolkit for natural language processing (described in the following section). Except of this factor, the strongest reasons for using Java Programming Language (Java) are:

Object oriented aspect This feature is, besides other things, useful for the extensibility of the application and offers means for rapid prototyping.

Platform independency Platform independency is perhaps the reason why Java is so widely used for NLP purposes. It allows to port a ready-made application to different environments without giving much effort.

Java 7 was used as the main programming language for the final application. In some cases we used Python but it was mainly for experimenting¹ with data corpus. The application was written in Eclipse development environment. We chose Git as the version control system. Our repositories are hosted on GitHub (public code) and Assembla (private code). Parts that required server were deployed to Apache Tomcat 7.

3.2.2 Overview of NLP Frameworks

IE toolkits offer scalable and robust NLP tools which can be easily integrated in the final IE system. The integration of analytic components is referred as NLP pipeline. Selection of the framework was, most of all, driven by the degree of use in current state-of-the-art extraction systems, by the support of modules and, last but not least, by the quality of documentation/community activity. Extensive list of NLP toolkits can be found on Wikipedia².

Two final candidates were:

Apache OpenNLP The Apache OpenNLP library³ is a machine learning based framework for natural language processing written in Java. Same as GATE, it supports variety of NLP tasks such as tokenization, sentence detection, part-of-speech tagging, chunking, parsing, named entity extraction and coreference resolution.

¹Repository with our helper functions: <https://github.com/jendarybak/NYTC-Helpers>

²http://en.wikipedia.org/wiki/Natural_language_processing_toolkits

³<http://opennlp.apache.org>

GATE - General Architecture for Text Engineering GATE is an open-source modular NLP tool whose architecture is defined by specialized components. Development of new components is encouraged and supported by the community and defined by the interface. List of available components can be found on the official webpage⁴. However, from our own experience, we would recommend searching in GATE's discussion forum if the component with desired functionality is not in the list.

We decided to use GATE for its extensibility. Apache OpenNLP can be, if necessary, run in GATE as a component. A comparison of these two systems can be found in [40].

⁴<http://gate.ac.uk/>

3.3 Architecture

3.3.1 Pipeline Architecture

The whole system can be viewed as a set of interconnected components that are organized in a way that the output from one part is the input to another. This kind of modular architecture brings the advantage of component interchangeability which also means better flexibility.

The pipeline is displayed and described in the figure below:

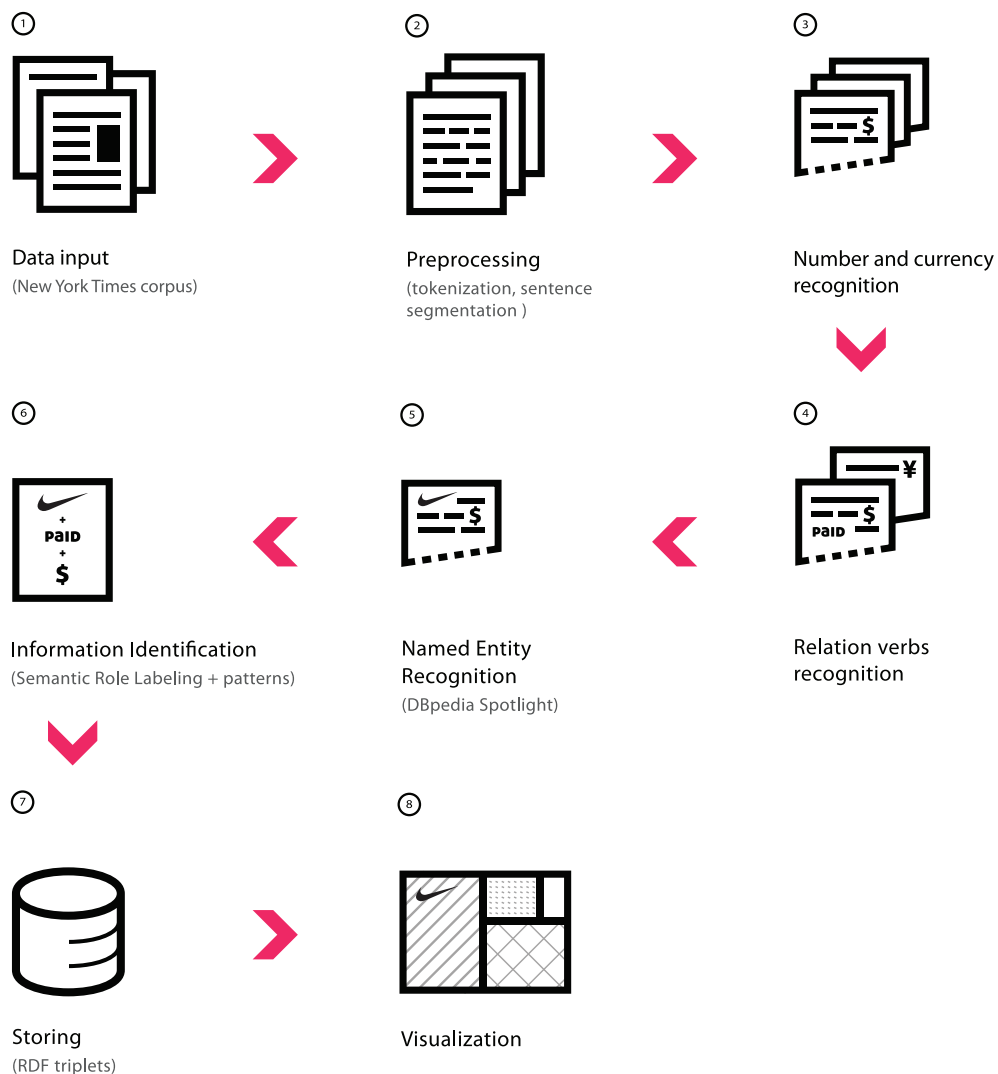


Figure 3.1: Pipeline architecture of the system

1. Data input

As described in the Section 3.4, data are provided by user in the form of The New York Times Corpus files. Each article is pre-processed in the step no.2.

2. Pre-processing steps

Pre-processing is a part of nearly every NLP system. In our application we utilize tokenization and sentence segmentation. Due to noun phrase chunking (Section 3.7.2) there was the need to include part-of-speech recognition as well.

3. Number and currency recognition

The recognition of finance-related information, as defined in the scope of this thesis, crucially depends on our ability to identify monetary information (number with currency). This step works at a sentence level and detailed description can be found in Section 3.5.

4. Relation verbs recognition

We created an ontology of finance-related verbs. Such verb might be promising indicator of a relation describing financial cost of some entity. All sentences containing monetary information are being examined for the presence of financial verb. Ontology creation and tagging process is further described in Section 3.6.

5. Originator Recognition

Originator Recognition is the part where possible subjects of the cost, in this thesis called *originators*, are marked. To put it simply, we considered only those subjects which have an entry in Wikipedia. Our approach is explained in Section 3.7.

6. Information Identification

Putting all the previous steps together gives us a range of possible information parts (money, relations, originators). In this stage we tried to find related pieces of information with a method called Semantic Role Labeling (Section 3.8.1).

6. Information Extraction

The final step of the actual extraction is identification of triples "*originator - has-Value - cost*" and extraction in RDF format. This process is described in Section 3.8.2.

7. Storing

The triples are stored in an RDF storage (Section 3.9).

8. Visualization

The second part of the thesis deals with data visualization.

3.4 Data

Our system is capable of processing free text. However, because of many advantages it brings, we had decided to use *The New York Times Corpus* as the source of data. We build a wrapper for parsing XML files of the corpus which exploits metadata described below.

3.4.1 New York Times corpus

In 2008 the New York Times newspapers released a corpus - The New York Times Corpus (NYTC) - containing articles from the period of 1987 - 2007. These articles are mostly provided with metadata and manually summarized and tagged content. The total sum of articles in this corpus is close to 1.8 million [41].

The corpus includes [41]:

- More than 1.8 million articles (excluding wire services articles that appeared during the covered period).
- More than 650,000 manually written article summaries.
- More than 1,500,000 articles manually tagged by library scientists with tags drawn from a normalized indexing vocabulary of people, organizations, locations and topic descriptors.
- More than 275,000 algorithmically-tagged articles that have been hand verified by the online production staff at nytimes.com.
- Java tools for parsing corpus documents from .xml into a memory resident object.

This collection contains over 650,000 article-summary pairs which may prove to be useful in the development and evaluation of algorithms for automated document summarization. Also, over 1.5 million documents have at least one tag. Articles are tagged for persons, places, organizations, titles and topics using a controlled vocabulary that is applied consistently across articles.”[41]

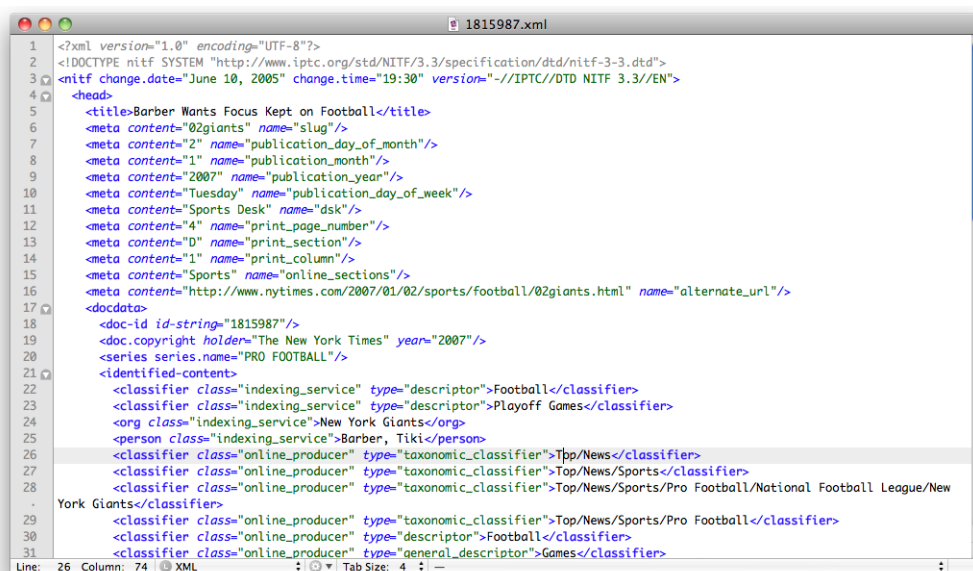
More details about the corpus can be obtained from the official catalogue web page⁵ of the NYTC, where the information in this section comes from, or from the corpus manual⁶.

⁵[urlhttp://www ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2008T19](http://www ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2008T19)

⁶[urlhttp://www ldc.upenn.edu/Catalog/docs/LDC2008T19/new_york_times_annotated_corpus.pdf](http://www ldc.upenn.edu/Catalog/docs/LDC2008T19/new_york_times_annotated_corpus.pdf)

Structure

The New York Times Annotated Corpus is provided as a collection of XML documents that conform to version 3.3 of the News Industry Text Format (NITF) specification. An example of one modified corpus file is a part of the accompanying CD.



```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE nitf SYSTEM "http://www.iptc.org/std/NITF/3.3/specification/dtd/nitf-3-3.dtd">
3 <nitf change.date="June 10, 2005" change.time="19:30" version="//IPTC//DTD NITF 3.3//EN">
4 <head>
5 <title>Barber Wants Focus Kept on Football</title>
6 <meta content="02giants" name="slug"/>
7 <meta content="2" name="publication_day_of_month"/>
8 <meta content="1" name="publication_month"/>
9 <meta content="2007" name="publication_year"/>
10 <meta content="Tuesday" name="publication_day_of_week"/>
11 <meta content="Sports Desk" name="dsk"/>
12 <meta content="4" name="print_page_number"/>
13 <meta content="D" name="print_section"/>
14 <meta content="1" name="print_column"/>
15 <meta content="Sports" name="online_sections"/>
16 <meta content="http://www.nytimes.com/2007/01/02/sports/football/02giants.html" name="alternate_url"/>
17 <docdata>
18 <doc-id id-string="1815987"/>
19 <doc.copyright holder="The New York Times" year="2007"/>
20 <series series.name="PRO FOOTBALL"/>
21 <identified-content>
22 <classifier class="indexing_service" type="descriptor">Football</classifier>
23 <classifier class="indexing_service" type="descriptor">Playoff Games</classifier>
24 <org class="indexing_service">New York Giants</org>
25 <person class="indexing_service">Barber, Tiki</person>
26 <classifier class="online_producer" type="taxonomic_classifier">Tpp/News</classifier>
27 <classifier class="online_producer" type="taxonomic_classifier">Top/News/Sports</classifier>
28 <classifier class="online_producer" type="taxonomic_classifier">Top/News/Sports/Pro Football/National Football League/New
  York Giants</classifier>
29 <classifier class="online_producer" type="taxonomic_classifier">Top/News/Sports/Pro Football</classifier>
30 <classifier class="online_producer" type="descriptor">Football</classifier>
31 <classifier class="online_producer" type="general_descriptor">Games</classifier>

```

Figure 3.2: Example of a file from The New York Times Corpus

Advantages

- Size of the corpus.
- The corpus contains meta data including e.g., category of the article, date, url, keywords.
- Timespan of the corpus - temporal aspect.
- Possibility of interlinking with NYT articles published on-line.

3.4.2 Twitter Data

Twitter⁷ messages, so called *tweets*, could be ideal candidates for information extraction by our system. There are several reasons why we think so. First of all, Twitter is truly popular and many news channels are using this service to spread out messages condensed into 140 characters. The length limitation is forcing authors to be factual and to express the important information preferably in one sentence. No co-reference resolution is therefore needed.

The availability of the data is another crucial factor. Twitter provides API that is with some limitations publicly available. Last but not least, selection of topics/tweets is up to the user and can be influenced by subscribing to various Twitter channels. Although we experimented with Twitter data, we were too busy with processing NYTC so that we are not able to publish any representable results with Twitter data at the moment.

⁷Social network, url: <http://www.twitter.com>

3.5 Monetary Information Recognition

In the majority of financial articles every mention about money consists of two parts - **amount** and **currency**. Therefore finding a number and a currency symbol as a prefix or a suffix to it apparently gives us a piece of monetary information.

```

Currency: USD
Value:    3 900 000 000

```



BP's profit dropped to **\$3.9 billion.**

Figure 3.3: Monetary information recognition

In the next two sections we will describe extraction of numbers and currency symbols. Let's now assume that these two key elements have been already identified. A simple grammar in JAPE format shown in listing 3.1 is able to annotate the pair as a 'money data' and assign normalized financial amount and currency identifier to the annotation (see Figure 3.4).

Listing 3.1: Grammar in JAPE format (currency.jape)

```

Phase: myCurrencyPhase

Input: Number Token SpaceToken Lookup
Options: control = appelt

Rule: MyCurrencyCategory
Priority: 100
(
  {Number} ({SpaceToken})*
  ({Lookup.majorType == "currency_unit",
    Lookup.minorType == "post_amount"}):cur
  |
  ({Lookup.majorType == "currency_unit",
    Lookup.minorType == "pre_amount"}):cur
  ({SpaceToken})* {Number}

): label
-->
:label.MyCurrency = {type = "Money",

```

```
value = :label.Number.value,
currency = :cur@string}
```

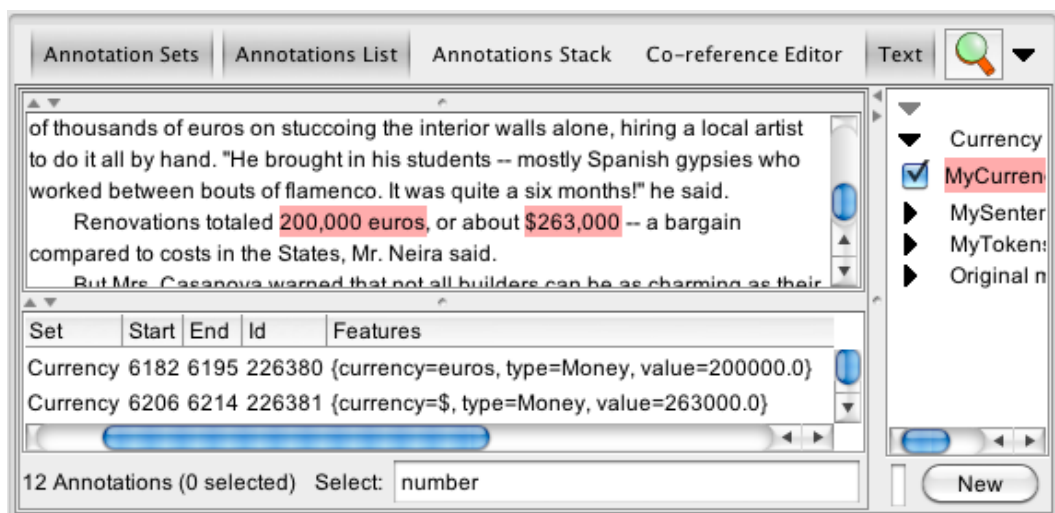


Figure 3.4: Annotated and normalized money data in GATE

3.5.1 Number Recognition

GATE in the standard implementation comes with a plugin which is capable of number recognition and tagging. The plugin is called *Tagger_Numbers*[14]. It needs the text to be tokenized and sentences to be segmented in advance. The plugin works on the rule-based principle where regular expression patterns are defined for numbers in numerical notation and gazetteer lists are used to recognize numbers in word form notation. After identification of either digits or numeral words, its numerical value is computed and assigned as a feature of the tag.

There were cases when the tagger failed. This was caused mostly by headline style of many news sources. It is common to use the following notation for large amounts:

- Would You Pay **\$329K** for this Phone?
- Here's what **\$100M** gets you in NY.
- Dell and Virgin Media in **\$20bn** deals
- World's biggest economies face **\$7.6T** debt.
- According to Henry's calculations, **\$6.3tn** of assets is owned by only 92,000 people.!

When one writes digits in combination with word-format numbers, usually there is a white space in between. That does not apply when using abbreviations mentioned before.

- John Paulson gives **\$100 million** to Central Park.
- John Paulson gives **\$100M** to Central Park.

We have solved these issues firstly by augmenting gazetteer lists with possible abbreviations and secondly by changing settings of the plugin so that it attempts to find numbers even within words. Plugin gives satisfactory results after the modification.

Names of Large Numbers

We would like to point out that there are differences in names of "large numbers" in British and American English. Since we are using the New York Times corpus whose origin is American, it is logical to parse numbers according to the *short scale*⁸ notation used in the United States. Example of differences in the short and the long scale is listed in the table3.1.

Table 3.1: Short (American) and Long (European) scale notation for large numbers.

<i>Name</i>	<i>Short scale value</i>	<i>Long scale value</i>
Million	10^6	10^6
Milliard	-	10^9
Billion	10^9	10^{12}
Trillion	10^{12}	10^{18}

⁸Comprehensive overview of large number notations can be found here:http://en.wikipedia.org/wiki/Names_of_large_numbers

3.5.2 Currency Recognition

Testing Data from The New York Times Annotated Corpus

Since there is no testing dataset for currency recognition problem available, we had to prepare our own. Initial idea was to manually search for monetary information in The New York Times Annotated Corpus⁹, extract only those relevant articles which contains numerical financial data and populate XML files with the following structure:

Listing 3.2: Testing data example

```
<TestDataItem id="83">
  <Sentence>
    <string>The global prime brokerage business generates
    $8 billion to $10 billion a year, according to an estimate
    by the Vodia Group, a consulting firm for the financial
    services industry.</string>
    <value currency="dollar">8000000000</value>
    <value currency="dollar">10000000000</value>
  </Sentence>
  <Source>
    <year>2007</year>
    <month>01</month>
    <day>02</day>
    <filename>1815975.xml</filename>
  </Source>
</TestDataItem>
```

id Unique identification number of each dataset item.

string Original sentence containing monetary expression.

currency Currency of the monetary expression. In the example above it is not clear what kind of dollar (CAD, USD, etc.) is mentioned in the sentence, therefore general label 'dollar' is used.

currency value Normalized numerical value.

year/month/day Publication date of the article.

filename Name of the XML file containing the sentence in The New York Times Annotated Corpus.

⁹<http://www ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2008T19>

Testing Data from Various Sources

However, we found out that the notation of currency expressions in the corpus is mostly following the same pattern:

'\$' + number + (million | billion | trillion).

Variety of the testing data was no longer expanding with increasing number of testing sentences. In order to obtain dataset which can satisfy testing of currency extraction from general text sources, we had to consider sentences from various sources.

Example of monetary notations from The New York Times Annotated Corpus:

- \$250
- \$5,000
- \$2.3 million
- \$24 million
- \$2 billion
- \$3,693,101.49

Example of monetary notations from random webpages:

- \$4.655
- 200 USD
- £1.55bn
- \$700m
- \$1tn
- \$16,066,241,407,385.80
- EUR1350
- ¥8,871.4 bn
- 200 Japanese yen
- GBP 400,000-600,000

Currency Recognition in GATE

Our system is using gazetteer to identify and annotate entity names. Currency units and symbols has to be reliably recognized, otherwise the monetary information would be ignored in the following phases of IE. We included all possible currency units, symbols and notations into gazetteer lists for currency prefixes and postfixes (`currency_prefix.lst`, `currency_unit.lst`). A tag format for labeling a currency unit has the following syntax:

```
Lookup.majorType == "currency_unit" .
```

Example of variability of notations for *Euro* (€):

- EUR3.14
- EUR 3.14
- 3.14 EUR
- €3.14
- € 3,14
- 3,14 euros
- 1 euro

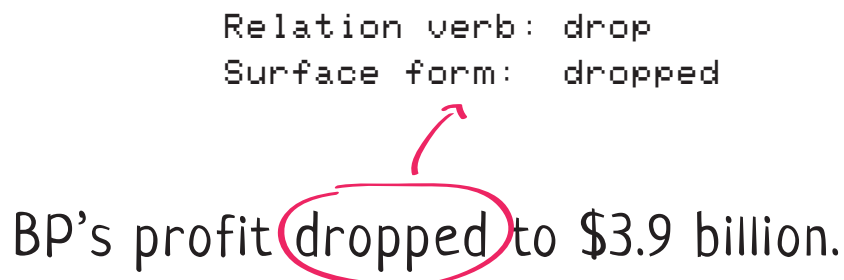
3.6 Relation Extraction

Finding a text snippet that contains a piece of monetary information indicates with high probability that the content is finance-related. However, the task is to process only those sentences that give us interesting information about a value of some subject. Such piece of information can be described as a relation between the subject and the cost of it (monetary information that we have already extracted). In human language relations are expressed by (linking) verbs and subjects are denoted by nouns (noun groups). This leaves us with the fact that we need to find finance-related verbs (relations) and identify the object/subject of the relation. In other words, we want to know who or what is the originator/agent of a transaction.

```

Relation verb: drop
Surface form:  dropped

```



BP's profit **dropped** to \$3.9 billion.

Figure 3.5: Financial verb contained in the ontology is recognized.

We decided to organize the list of finance-related verbs in the form of an ontology that can be utilized later on, if needed. The reason for ontology creation is that its structure brings in information about the hierarchy of the verbs (i.e. relations), which can be important for further process of extraction.

3.6.1 Ontology

Financial Relations Ontology

Our system is considered for the operation in relatively narrow domain - financial information in news articles. If we consider that there is a limited amount of possible relation types in the field and that ontologies can be scaled easily, thus utilizing an ontology to hold information about financial relations can give good extraction results.

We searched for an existing ontology of relations appearing in financial context but we did not succeed in finding one which would suit our needs. There are either ontologies dedicated only to financial terminology¹⁰ or relation ontologies that are too

¹⁰<http://fadyart.com/>

general¹¹. This was the reason why we created the ontology by ourselves.

Financial Relations Ontology Creation

According to the previously mentioned approaches (see Section 2.8), we used the combination of Top-Down and Bottom-Up development methods. We established two main classes as the most important criterion for further division:

"INTERACT"

- a relation requiring two participants (originator and recipient); an object of some value is the reason of this interaction.

"HAVE VALUE"

- a relation expressing financial value of an object.

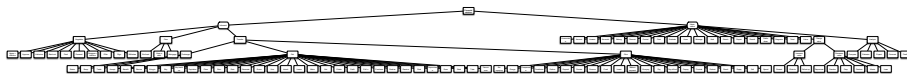


Figure 3.6: Financial relationships ontology.

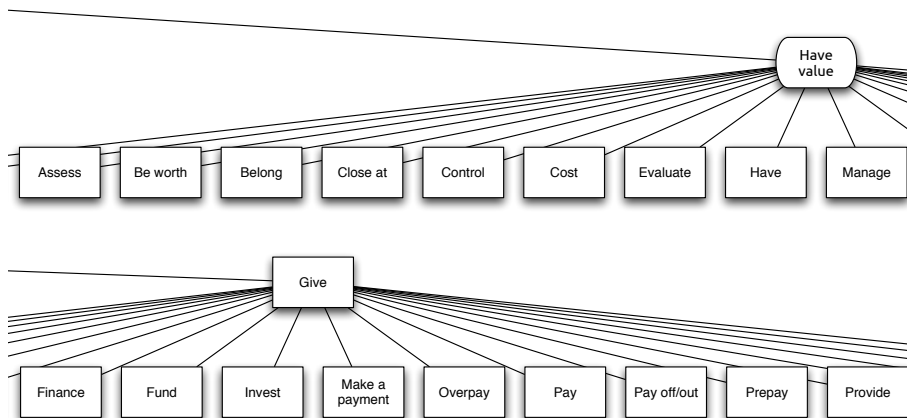


Figure 3.7: Detail of financial relationships ontology.

¹¹<http://www.obofoundry.org/ro/>

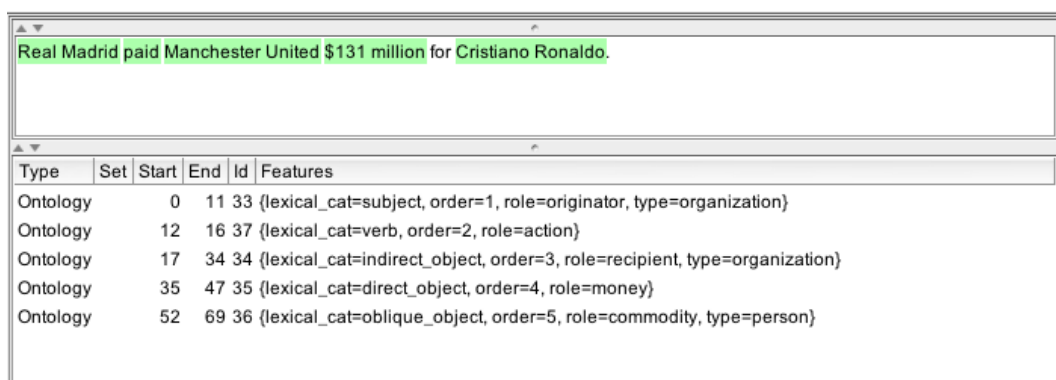
We applied the following algorithm:

1. We extracted a sample set of finance-related sentences from NYT corpus.
2. In every sentence we manually looked up relations among monetary information, subject and object and made a list of corresponding verbs.
3. We found and appended synonyms of previously acquired verbs in the list.
4. We made 2 main classes ("Interact", "Have value").
5. We grouped verbs with similar meaning and similar hypernym ¹².
6. We made a superclass from the given hypernym.
7. If there were two hypernyms with similar meaning, we merged them into one class.
8. We repeated steps 5) and 6) for all verbs until we reached one of the main classes.
9. We manually revised placement of few verbs.

For acquisition of synonyms, hypernyms and hyponyms a large lexical database of English called WordNet¹³ was used.

Ontology Tagging

There is a specialized plugin for ontology tagging in GATE. However, due to its relatively small size we decided to convert the ontology to gazetteer list and use GATE transducer component to annotate financial verbs in the text. The reason for this step was the improved speed of annotation.



The screenshot shows a window with a text area containing the sentence "Real Madrid paid Manchester United \$131 million for Cristiano Ronaldo." The words "Real Madrid", "Manchester United", "\$131 million", and "Cristiano Ronaldo" are highlighted in green. Below the text area is a table with the following data:

Type	Set	Start	End	Id	Features
Ontology		0	11	33	{lexical_cat=subject, order=1, role=originator, type=organization}
Ontology		12	16	37	{lexical_cat=verb, order=2, role=action}
Ontology		17	34	34	{lexical_cat=indirect_object, order=3, role=recipient, type=organization}
Ontology		35	47	35	{lexical_cat=direct_object, order=4, role=money}
Ontology		52	69	36	{lexical_cat=oblique_object, order=5, role=commodity, type=person}

Figure 3.8: Annotated sentence in GATE.

¹²Hypernym - a word that is more generic than a given word

¹³<http://wordnet.princeton.edu/>

3.7 Originator Recognition

Once we are able to recognize a text snippet containing monetary information and relation verb of our interest, we need to find out what does this information relate to. In other words, who or what is the originator/actor of a transaction. First step is to recognize all possible noun groups that might be in the relation with the financial verb. After that it is necessary to select one candidate which is actually in the relationship with the verb and the cost. The aim is to extract popular pieces of information from well known news sources that is why we are primarily focused on generally known originators. In addition, we attempt to link extracted originators with publicly available datasets such as DBpedia¹⁴. This feature allows users to instantly gain more knowledge on the topic.



Figure 3.9: Example of originator (named entity) identification.

3.7.1 DBpedia

Wikipedia is a well-known source of quality information which is being edited by thousands of volunteers. We believe that considering Wikipedia articles as a set of admissible originators is a good heuristics how to separate possibly interesting pieces of information. Moreover, it gives us the possibility to link the originator with a Wikipedia article and thus offer relevant background information about the topic.



Figure 3.10: Entities linked with DBpedia.

¹⁴<http://dbpedia.org/>

Extracting all information from Wikipedia pages would be tedious process. Fortunately there is a very convenient alternative called DBpedia which is described in Section 2.9.3.

NER - Named Entity Recognition

In the overview of state-of-the-art in NLP, we mentioned several methods used for named entity recognition. In the system we gave preference to NER utilizing the ontology and the data set of DBpedia that we mentioned above.

The main advantage of such approach is the constant growth of the named entities database with no extra effort from our side. In addition, it is possible to effectively influence the results of the whole system in the very beginning of the extraction process. By limiting number of the named entities in the text, the number of sentences for further processing can be proportionally reduced. As we describe in the following section, we can write custom queries to obtain arbitrary set of DBpedia items. These queries are written in RDF query language called SPARQL and can be arbitrarily specific. If desired, we can list for example just athletes who are older than 25 years, IT companies from Europe or countries belonging to G8. However, in our application we specified more general concepts to be annotated such as persons¹⁵, organizations¹⁶ and places¹⁷.

The main challenge of the named entities recognition is ambiguity in annotated data. There are entities that can have several meanings in the given form but without considering the context, like humans do, there is no way how to automatically choose the correct one (*e.g. Apple - is it a fruit or a company?*). Damljancovic and Bontcheva[15] have recently developed a disambiguation method which improved F-measure of NER from 0.24 to 0.82. This method uses context of the particular entity and various similarity metrics described in the paper. Inspired by the disambiguation improvement we have decided to include context-aware method of NER in the application.

DBpedia Spotlight

DBpedia Spotlight¹⁸ is an open-source tool for annotating text with DBpedia resources. It is able to recognize entities from DBpedia in a text and simultaneously link these entities with the knowledge base. *"DBpedia Spotlight allows users to configure the annota-*

¹⁵RDF label: "dbpedia-ont:Person"

¹⁶RDF label: "dbpedia-ont:Organisation"

¹⁷RDF label: "dbpedia-ont:Place"

¹⁸<http://github.com/dbpedia-spotlight/>

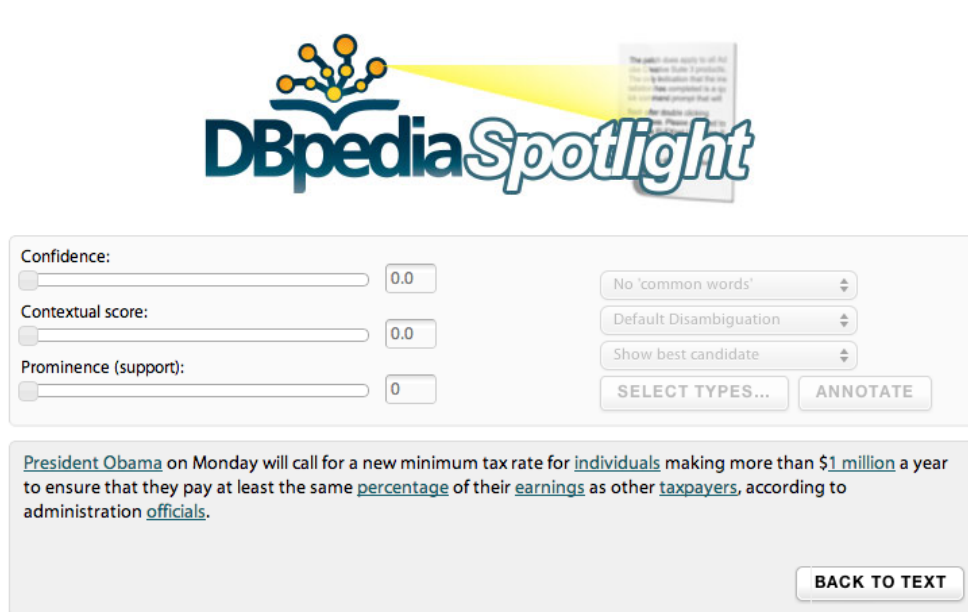


Figure 3.11: DBpedia Spotlight online demo.

tions to their specific needs through the DBpedia Ontology and quality measures such as prominence, topical pertinence, contextual ambiguity and disambiguation confidence”[26].

As mentioned before, the main advantage of DBpedia Spotlight is its ability to influence the ambiguity of NER. In default settings DBpedia Spotlight performs annotation with all DBpedia Ontology categories, that is more than 270 classes (not only people, organizations and places). On the other hand, it is possible to restrict annotation to instances of particular classes and subclasses, alternatively used as fine-grained SPARQL queries to constrain the set of classes to annotate. The system is highly customizable and can be set up in a way which exactly reflects our needs.

Custom configuration of DBpedia Spotlight

Resource Set to Annotate User can provide a list of allowed (whitelist) or forbidden (blacklist) resources to annotate. Classes are provided by DBpedia Ontology. For more specific domain restriction SPARQL queries are available.

Resource Prominence To avoid annotation of rather rare and uninformative resources the support parameter can be used. This parameter denotes how many times a resource is referenced in Wikipedia (so called backlinks).

Topic Pertinence The topical relevance indicates how close a paragraph is to a DBpedia

resource's context. This is measured by similarity score on the scale 0-1 (0 being unrelated, 1 totally relevant).

Contextual Ambiguity If there is more than one candidate with high topical pertinence in the given context, the contextual ambiguity is counted as the relative difference in topic score between the first two candidates.

Disambiguation Confidence This parameter is defined by the user and combines previously mentioned factors - topic relevance and context ambiguity. Its value is ranging from 0 to 1 and a higher threshold results in omitting incorrect annotations at the risk of losing some correct annotations.

GATE Module

In order to use DBpedia Spotlight's annotations we had to create a GATE module for this purpose. DBpedia Spotlight offers RESTful¹⁹ and SOAP²⁰ web services for the annotation and disambiguation processes with the possibility to set all the supported parameters. Returned results can be obtained in several formats including XML, JSON, HTML, etc. Our module currently works with XML format and supports all possible parameters of DBpedia Spotlight.


We made the plugin available to public²¹ on GitHub²². Up to date instructions to install and run the module can be found on the repository webpage or in README file. Directory structure of the plugin is depicted in Figure 3.12.

¹⁹Representational State Transfer



²⁰Simple Object Access Protocol

²¹http://github.com/jendarybak/GATE-DBpedia_Spotlight

²²GIT repository: <http://github.com/>

GATE-DBpedia_Spotlight /  17 commits

Readme edited.

 jendarybak authored 3 months ago latest commit efb404cc2 










 lib	4 months ago	First working version (except Type/SPARQL customization). [jendarybak]
 resources	4 months ago	Folders for external files and libraries added. [jendarybak]
 src	4 months ago	outputASName bug fixed / variables types fixed [jendarybak]
 .gitignore	4 months ago	First working version (except Type/SPARQL customization). [jendarybak]
 README	4 months ago	Initial CREOLE files. [jendarybak]
 README.md	3 months ago	Readme edited. [jendarybak]
 build.properties	4 months ago	Initial CREOLE files. [jendarybak]
 build.xml	4 months ago	Basic structure of GATE plugin [jendarybak]
 creole.xml	4 months ago	outputASName bug fixed / variables types fixed [jendarybak]

Figure 3.12: Directory structure of DBpedia Spotlight Tagger on Git.

Functionality of the plugin can be described by the following pseudocode:

Listing 3.3: Annotation process in DBpedia Spotlight Tagger

```

initialize the component;
read and check validity of input arguments;
read contents of the input document;
prepare HTTP request (POST);
send HTTP request to DBpedia Spotlight server;
receive XML answer A;
parse XML;

for (each entity in A) {
    extract arguments of the entity; // e.g. URI, surface form, etc.
    annotate corresponding text in the input document with the arguments;
}

```

Large Knowledge Base (LKB) Gazetteer in GATE

We used DBpedia Spotlight's RESTful web service with its disambiguation capability. Therefore, the final results of our system are more precise, but time consuming. If it is the case that the speed is prioritized over precision, we propose a solution using gazetteers with lists of DBpedia resources.

One of the plugins implemented in GATE is the ontology-based gazetteer called LKB Gazetteer. At the time of plugin initialization, a connection to remote ontology is established, prepared queries are sent and a new gazetteer list is generated from the result.

After the lookup, URIs are assigned to entities in the text.

We prepared queries to retrieve list of people, organizations and places. These lists are loaded every time GATE is launched and are therefore up to date with DBpedia (Wikipedia).

Querying DBpedia

DBpedia is based on Linked Data principles [3] and RDF data model, therefore accessing the knowledge base is very easy. To query the DBpedia data set a public SPARQL endpoint²³ is offered. SPARQL is a query language designed to manipulate data stored in RDF format. By writing SPARQL query we can obtain a list of items in arbitrary Wikipedia category. "SPARQL is able to query by triple patterns, conjunctions, disjunctions, and optional patterns." [35]

²³<http://dbpedia.org/sparql>

Example (query to show all organizations):

Listing 3.4: SPARQL query which lists organizations.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?Name ?Org ?Cls
FROM <http://www.ontotext.com/disable-sameAs>
WHERE {
  ?Org a ?Cls ; rdfs:label ?Name .
  FILTER (lang(?Name) = "en")
  FILTER (?Cls = <http://dbpedia.org/ontology/Organisation>)
}
```

3.7.2 Noun Phrase Chunking

Chunking is a technique used to divide sentences into its constituent parts which do not overlap and are syntactically correlated. Noun Phrase Chunking is a special case whose task is the identification of noun groups.



Figure 3.13: Example of noun phrase chunking.

Our idea was to use this technique to expand previously recognized named entities. Considering named entities as sufficient expressions for the action originators can in fact result in misleading information (see example below):

Example:

The entire **United States** ad market *is worth* about *\$286 billion*.

The entire **United States ad market** *is worth* about *\$286 billion*.

Approach

Because Machine Learning methods of noun phrase chunking are arguably on the up-trend and their results are significantly better compared to pattern recognition methods, we have opted for the statistical approach. We adapted this technique to our needs by combination with regular expressions. The standard part of GATE system is Ramshaw and Marcus BaseNP chunker[37] and we decided to use this one.

Process of extended noun chunk identification:

1. Assign POS tags to all words in the corpora.
2. Label all noun chunks with the tagger.
3. Remove label from chunks that do not contain any named entity (pers., org. , loc.).
4. Expand the label according to regular expression so that it covers all noun chunks connected by prepositions.

Even though this solution was working well, we decided to omit this part from the final system because it became redundant after implementing Semantic Role Labeling.

3.7.3 Part of Speech Tagging (POS)

Part of Speech Tagging is a process of assigning lexical category (verb, noun, etc.) and syntactic labels (gerund, possessive pronoun, etc.) to each word in a sentence or a document. We are using these tags for the purposes of noun phrase chunking. Semantic role tagger is also using part-of-speech tags but it is utilizing its own tagger. We rely on Mark Hepple's Brill-style POS tagger[22] which is a standard part of GATE framework. Since the whole system is modular, in case of need, the replacement for another implementation does not take much effort.

3.7.4 Extension of Noun Phrases using Regular Expressions

As mentioned above, noun chunks are non-overlapping phrases. In the case of two or more noun chunks connected by a preposition, it is very likely that we can consider the whole connection of phrases as the originator of some action. That is why we prepared regular expression rules to expand the label in such cases.

Example for location noun chunks:

Listing 3.5: Noun chunk extension with Regular Expressions

```
Rule: originatorExtraction
Priority: 100
(
  (
    (
      (
        {NounChunk, Token.kind != "number"}
        ({SpaceToken})?
        ({Token.category == "IN"} {SpaceToken})
        ?)*
      (
        {NounChunk contains LOCATION}
        ({NounChunk})
        ?)
    ):OrigLoc
  )
)
-->
:origLoc.Originator = {type="Location"}
```

3.8 Information Identification and Extraction

In the previous sections, our approaches to identification of monetary information, financial verbs and named entities in a text were described. Since there is no guarantee that all sentences contain only one piece of monetary information, one financial verb and one named entity and that there is necessarily a relationship to each other, we had to find a solution to identify and extract only those parts that belong to each other. First, we were thinking about applying custom methods of machine learning on this problem, but we found more appropriate solution in the technique called Semantic Role Labeling.

3.8.1 Semantic Role Labeling

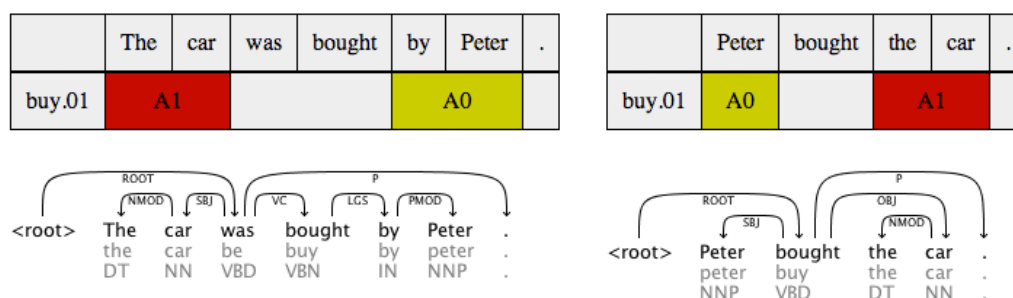


Figure 3.14: SRL representation of two sentences.

MATE Tools

For the purpose of SRL we used a state-of-the-art semantic role labeler developed by Anders Björkelund, Love Hafdell, and Pierre Nugues [4]. According to [4] the SRL module achieves an average labeled semantic $F1$ of 80.90 when trained and tested on the English corpus of *CoNLL 2009*²⁴. There are more SRL systems with comparable results available such as *Illinois Semantic Role Labeler*[36] or *SENNA*[12]. The one we decided to use offered the best combination of results quality, memory performance, suitability for GATE implementation and ease of installation/usage.

The SRL system is data-driven and therefore dependent on trained models. Models for English, Chinese, Spanish and German are provided on project's website²⁵. The application is running as a HTTP server that accepts a text snippet as the input and provides three format types for the output:

²⁴<http://ufal.mff.cuni.cz/conll2009-st/>

²⁵<https://code.google.com/p/mate-tools/downloads/list>

1. HTML

Returns a webpage with a table formatted according to CoNLL-2009 data format, arguments visualization and arguments dependency graph (fig. 3.15).

2. Raw text

Returns plain text document in CoNLL-2009 data format.

3. RDF/N3

Returns data in RDF/N3 format.

The screenshot shows a web browser window titled 'Semantic Role Labeler Demo' with the URL 'barbar.cs.lth.se:8081/parse'. The main content area displays a table with the following data:

	Google	bought	YouTube	in	October	for	\$	1.65	billion	.
buy.01	A0		A1	AM-TMP				A3		

Below the table, it states 'Parsing sentence required 11ms.' and shows a dependency graph with nodes for each word and their grammatical roles (e.g., SBJ, OBJ, TMP, PMOD, ADV, DEP, AMOD). The graph shows dependencies between 'Google' (SBJ), 'bought' (ROOT), 'YouTube' (OBJ), 'in' (TMP), 'October' (PMOD), 'for' (ADV), '\$' (AMOD), '1.65' (DEP), and 'billion' (DEP).

At the bottom, there is a detailed table with the following columns: ID, Form, Lemma, PLemma, POS, PPOS, Feats, PFeats, Head, PHead, Deprel, PDeprel, IsPred, Pred, and Args: buy.01. The data rows are as follows:

ID	Form	Lemma	PLemma	POS	PPOS	Feats	PFeats	Head	PHead	Deprel	PDeprel	IsPred	Pred	Args: buy.01
1	Google	google	google	NNP				2		SBJ				A0
2	bought	buy	buy	VBD				0		ROOT		Y	buy.01	
3	YouTube	youtube	youtube	NNP				2		OBJ				A1
4	in	in	in	IN				2		TMP				AM-TMP
5	October	october	october	NNP				4		PMOD				
6	for	for	for	IN				2		ADV				A3
7	\$	\$	\$	\$				6		PMOD				
8	1.65	1.65	1.65	CD				7		DEP				

Figure 3.15: Example of SRL web interface.

GATE Module

At the time of writing this thesis there was no SRL module available as a plugin for GATE and that is why we decided to implement MATE Tools module and make it public on GitHub²⁶. Contrary to DBpedia Spotlight module, this plugin does not require any additional parameters except of the SRL server address (fig. 3.16).

²⁶SRL module for GATE is available at: <https://github.com/jendarybak/GATE-SRL>

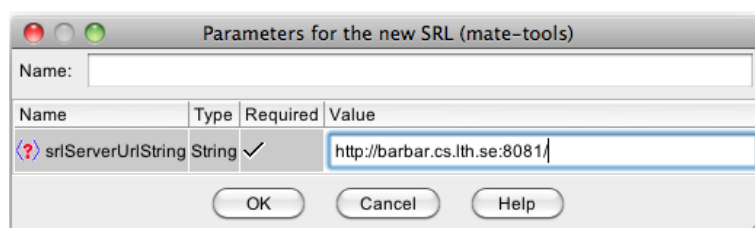


Figure 3.16: SRL module in GATE.

The module sends a sentence to the SRL server, which is listening on a specified address, and receives an answer in CoNLL-2009 format. We had to create a parser of the CoNLL-2009 format that is able to recognize predicates and read their arguments. All recognized arguments for each predicate are annotated in the GATE file. Later on, in the triple extraction phase, only the finance-related predicates are considered and extracted.

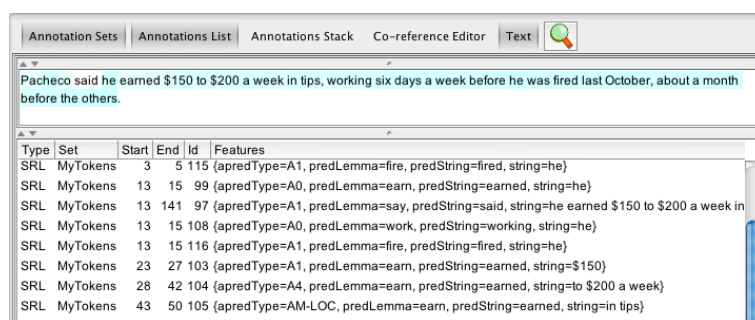


Figure 3.17: SRL annotation in GATE.

3.8.2 Extraction of Related Triples

At this point, we have only those sentences which meet our requirements to be good candidates for information extraction, i.e. sentences containing a DBpedia entity, monetary information and a finance-related verb. We have also identified semantic roles of predicate's arguments. The key questions now are whether the money reference is related to the verb and whether the DBpedia entity describes participants of the financial transaction.

With the help of the PropBank role sets, it is possible to identify which arguments should describe the *agent*, *commodity*, and *financial value* of the commodity. Let's have a look at the following role set for the predicate *buy*.⁰¹ (Fig. 3.18).

Predicate: *buy**buy*: Frames file for 'buy' based on first sentences in large corpusRoleset id: **buy.01** , *purchase*, vncls: **13.5.1**, framnet:**Roles:**

Arg0: *buyer* (vnrole: 13.5.1-Agent)
Arg1: *thing bought* (vnrole: 13.5.1-Theme)
Arg2: *seller* (vnrole: 13.5.1-Source)
Arg3: *price paid* (vnrole: 13.5.1-Asset)
Arg4: *benefactive* (vnrole: 13.5.1-Beneficiary)

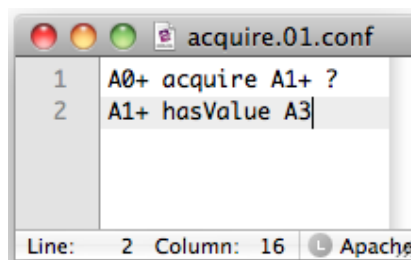
Figure 3.18: Role set for predicate *buy*.01.

In the output XML file (see Appendix A) you can see that the predicate is identified by its role set and lemma. All arguments have an attribute called `apredType` which denotes their semantic role. If there is a DBpedia entity contained in the argument, a `spotlight` child element is created to carry information about it. The same applies to monetary information, just the element is called `finance`.

Extraction Patterns

We created a system of patterns for specification of the predicate and arguments that we want to extract as a triple. For each predicate there is one configuration file (fig. 3.19) which holds one triple pattern per line.

Writing a question mark (?) behind the triple pattern signifies that the triple can be skipped if there are no matching arguments. User can define whether a DBpedia entity is required in the particular argument by simply writing a plus sign (+) after the argument's label. Relation called `hasValue` has special meaning because the argument which follows this keyword must contain financial information, otherwise no triples are extracted.

Figure 3.19: Example of extraction pattern for predicate *acquire*.01

3.9 Information Storage



Figure 3.20: Information represented as an RDF triplet.

We decided to store acquired information in the form of RDF triples. This format is theoretically described in section 2.9.1. The reason to use RDFs for storage and operation with data instead of, for instance, relational database is simple - the data can be easily shared as Linked Data. Another benefit it brings is the possibility of interlinking our data with, for example, data from Wikipedia.

3.9.1 RDF Triples

Turtle format

Turtle (Terse RDF Triple Language) is one of many RDF syntaxes. We decided to use this format because it is more readable than the others. Another aspect is its compatibility with many RDF storages, especially with Sesame, the one we are using. Complete grammar of the Turtle syntax can be found at the official W3C specification web page [16]. Turtle supports three RDF term types:

- URI references
- Literals
- Blank nodes

In our application we use all of them. Worth mentioning is the concept of blank nodes. It is extremely useful when a unique identifier is needed. Following syntax (`_:inf`) will ensure the assignment of a unique id to every single piece of information:

Listing 3.6: Blank node in Turtle

```
_:inf rdf:type inf:information .
_:inf rel:file "1730066.xml" .
_:inf xsd:date "2006-01-07"^^xsd:date .
_:inf rel:sentence "Hilton said that it was acquiring ... for $5 billion."
```

XML to Turtle conversion

We implemented the process of triple statement retrieval (described in 3.8.2) in an application called `TurtleTransformer`. This application takes two arguments - a folder with the extraction patterns and a folder with extracted XML files. Each XML file represents, among other things, one predicate and its arguments. This application identifies the argument and provided there is a corresponding extraction pattern, an RDF triplet is composed and saved.

Example of RDF output (Sony's acquisition of CBS Records)

Listing 3.7: RDF output

```
@prefix rel: <http://www.rybak.in/schemas/relationship/> .
@prefix ent: <http://www.rybak.in/schemas/entities/> .
@prefix money: <http://www.rybak.in/schemas/money/> .

ent:Sony rel:stringValue "Sony" .
ent:Sony rel:seeAlso <http://dbpedia.org/resource/Sony> .
ent:CBS_Records rel:stringValue "CBS Records" .
ent:CBS_Records rel:seeAlso <http://dbpedia.org/resource/CBS_Records> .
ent:Sony rel:acquire ent:CBS_Records .

ent:CBS_Records rel:stringValue "CBS Records" .
ent:CBS_Records rel:seeAlso <http://dbpedia.org/resource/CBS_Records> .
ent:CBS_Records rel:hasValue ent:USD_3-4E12 .

ent:USD_3-4E12 rel:currency ent:USD .
ent:USD_3-4E12 rel:nominalValue 3.4E12 .
```

3.9.2 RDF Storage

We use Sesame 2.7.0²⁷ as the RDF framework for the storage and querying. Our decision for this selection was based mainly on the wide range of supported RDF formats (Turtle, N-Triples, RDF/XML, etc.), easy-to-use API and simple installation process (deployment to Apache Tomcat).

3.9.3 RDF Triple uploader

To avoid the tedious manual work of RDF uploading, we prepared a simple program, which connects to the Sesame RDF storage and uploads all RDF files from a specified

²⁷<http://www.openrdf.org/>

Workbench

Sesame server

Repositories
 New repository
 Delete repository

Explore
 Summary
 Namespaces
 Contexts
 Types
 Explore
 Query
 Saved Queries
 Export

Current Selections:
 Sesame server: [change](#)
 Repository: - none - [change](#)
 User (optional): - none - [change](#)

List of Repositories

Id	Description	Location
SYSTEM	System configuration repository	http://localhost:8080/openrdf-sesame-2.7.0/repositories/SYSTEM
memory	Memory Store	http://localhost:8080/openrdf-sesame-2.7.0/repositories/memory
financial	Financial relations	http://localhost:8080/openrdf-sesame-2.7.0/repositories/financial
Test	Test	http://localhost:8080/openrdf-sesame-2.7.0/repositories/Test

Figure 3.21: Sesame Web Interface

Subject	Predicate	Object
_model7nopus4x45	rdf:object	<http://www.rybak.i
_model7nopus4x45	rdf:predicate	<http://www.rybak.i
_model7nopus4x45	rdf:subject	<http://www.rybak.i
<http://www.rybak.in/visue/schemas/entities#to_the_private_equity_firm_Texas_Pacific_Group>	rdf:value	"to the private equity"
<http://www.rybak.in/visue/schemas/entities#its_underperforming_Myer_department_store_unit>	rdf:value	"its underperforming"
<http://www.rybak.in/visue/schemas/entities#to_the_private_equity_firm_Texas_Pacific_Group>	rdfs:seeAlso	<http://dbpedia.org/
<http://www.rybak.in/visue/schemas/entities#its_underperforming_Myer_department_store_unit>	rdfs:seeAlso	<http://dbpedia.org/
_model7nopus4x44	<http://www.rybak.in/visue/schemas/relationship#relation>	_model7nopus4x4
_model7nopus4x46	rdf:object	<http://www.rybak.i
_model7nopus4x46	rdf:predicate	<http://www.rybak.i
_model7nopus4x46	rdf:subject	_model7nopus4x4
_model7nopus4x47	<http://www.rybak.in/visue/schemas/money#currency>	"USD"
_model7nopus4x47	<http://www.rybak.in/visue/schemas/money#value>	1.13E9
_model7nopus4x44	rdf:type	<http://www.rybak.i
_model7nopus4x44	<http://www.rybak.in/visue/schemas/relationship#field>	"1838002"
_model7nopus4x44	<http://www.rybak.in/visue/schemas/relationship#file>	"1838002.xml"
_model7nopus4x44	<http://www.rybak.in/visue/schemas/relationship#source>	"NewYorkTimesCorp"
_model7nopus4x44	rdfs:seeAlso	<http://query.nytime
_model7nopus4x44	xsd:date	2007-04-04
_model7nopus4x44	<http://www.rybak.in/visue/schemas/relationship#relation>	_model7nopus4x4
_model7nopus4x48	rdf:type	<http://www.rybak.i

Figure 3.22: RDF repository

folder automatically.

Chapter 4

Evaluation

4.1 Evaluation Data

In our experimental setup we processed New York Times articles from the year 2006. The total number of articles published that year is 87429. We used a limited ontology of finance-related verbs, which contains the 25 most essential predicates describing a financial activity (acquire, buy, sell, pay, etc.).

4.2 Evaluation Metrics

Our methodology of the extraction evaluation is based on the commonly used measures for information retrieval performance. *Precision* is in the scope of this thesis understood as a number of relevant information triples in the result set. By the term relevant we mean a triple, which fulfills the following criteria:

1. Financial value

Both the currency and numerical value are correct and standardized.

2. DBpedia entity

The originator of the financial cost contains one or more DBpedia entities, which are relevant to the subject.

3. Originator phrase

The name of the originator does not necessarily have to be one or two words long. Nevertheless, phrases containing words with no relevance to the subject are considered incorrect.

4. RDF triple

The RDF triple file has to be in a valid form and has to represent the information correctly.

Formula used for precision evaluation looks as follows:

$$(P)recision = \frac{|\text{relevant information triples}|}{|\text{all information triples extracted}|} \quad (4.1)$$

4.3 Evaluation results

4.3.1 Precision

We manually evaluated 112 randomly picked information triples¹ and the resulting precision was 48,21%. This may seem like a poor performance, but in general, 94,64% of the triples carried meaningful information. Most of the errors were caused by the wrong assignment of DBpedia entity. Bellow we provide an analysis of error causes:

Table 4.1: Sources of inaccuracy in extraction of triples.

<i>Preprocessing</i>	<i>Money identification</i>	<i>DBpedia entity</i>	<i>Semantic Labeling</i>	<i>Triple extraction</i>
2.94%	1.47%	80.88%	11.76%	2.94%

To illustrate an example of incorrect result, let's have a look at the following triple:

Subject: all-wool king-size mattress
Relation: has value
Cost: USD 2000
DBpedia: http://dbpedia.org/resource/Elvis_Presley

Except of the entity assignment by DBpedia Spotlight, the information is correct. Nevertheless, it is clear that the mattress in this sentence is not called king-size because of Elvis Presley.

¹This sample can be viewed at <http://rybak.in/visue-evaluation/>

4.3.2 Recall

Our application is meant to be precision-oriented. The recall can be improved by extending the ontology of financial relations. Instead of manual (and extremely time-consuming) evaluation of recall, we decided to measure number of sentences in every single step of the extraction process. We don't know the exact number of sentences in the unprocessed corpus, but our estimation is 150 000 - 200 000 sentences. However, we can use the number of sentences containing monetary information as a reference number, since the money recognizer is reliable part with high recall.

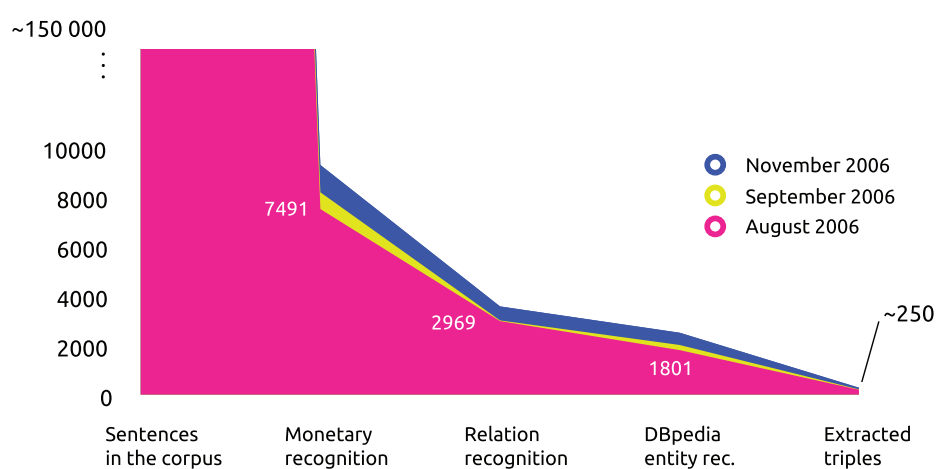


Figure 4.1: Distribution of sentences in subsequent phases of the extraction.

Table 4.2: Number of sentences in each phase of the extraction (year 2006).

<i>Month</i>	<i>Money identification</i>	<i>Financial relations rec.</i>	<i>DBpedia</i>	<i>Extracted triples</i>
November	9 276	3 560	2 496	272
September	8 177	3 001	2 004	204
August	7 491	2 969	1 801	195

4.4 Evaluation Interface

As the evaluation process requires manual work, we prepared an interactive webpage which makes this process much easier - we call it VISUE Evaluation Tool². This webpage loads extraction XML files together with the final RDF triples and shows them in a convenient way to the evaluator. All steps of the extraction phase can be marked as correctly/incorrectly processed. Results are collected in our database and further examined. For faster navigation there is a set of keyboard shortcuts that are listed in Table 4.3. This tool was developed in PHP, HTML, CSS, JavaScript (jQuery) and MySQL database.

Table 4.3: VISUE Extraction Tool keyboard shortcuts.

Q	previous file
W	next file (skip evaluation)
S	open a window with <i>Semantic Role Labeling</i> service
D	open a window with <i>DBpedia Spotlight</i> service
F	open an output file from the extraction phase (XML)
G	open an extraction pattern
A	open all previously mentioned services and files
Enter	confirm evaluation form and proceed to the following file

The screenshot displays the VISUE Evaluation tool interface. At the top, it shows the file name '1838254.xml', the date '2007-4-6', and the online source 'http://query.nytimes.com/gst/fullpage.html?res=9406E0D8163FF935A35797C0A9619C8B63'. The main content area shows a sentence: 'Christie's estimates the painting will bring \$25 million to \$35 million, exceeding the record for Warhol set at Christie's in November, when a 1972 portrait of Mao was bought by Joseph Lau, a Hong Kong collector, for \$17.3 million.' Below the sentence, the predicate is 'buy.01 (bought)'. The arguments section lists: A1: November, when a 1972 portrait of Mao was; R-AM-TMP: when; A1: a 1972 portrait of Mao; A0: by Joseph Lau, a Hong Kong collector,; A3: for \$17.3 million. The DBpedia section lists entities: Mao (http://dbpedia.org/resource/Mao_Zedong), Hong Kong (http://dbpedia.org/resource/Hong_Kong_Island), and collector (http://dbpedia.org/resource/The_Collector). The Finance section shows '\$17.3 million' and 'USD 1.73E7'. The RDF Triples section shows: ent:by_Joseph_Lau_-_a_Hong_Kong_collector-rel:stringValue "by Joseph Lau, a Hong Kong collector,"; ent:by_Joseph_Lau_-_a_Hong_Kong_collector-rel:seeAlso http://dbpedia.org/resource/Hong_Kong_Island; ent:by_Joseph_Lau_-_a_Hong_Kong_collector-rel:stringValue "a 1972 portrait of Mao"; ent:a_1972_portrait_of_Mao-rel:seeAlso http://dbpedia.org/resource/Mao_Zedong; ent:by_Joseph_Lau_-_a_Hong_Kong_collector-rel:buy ent:a_1972_portrait_of_Mao; ent:a_1972_portrait_of_Mao-rel:stringValue "a 1972 portrait of Mao"; ent:a_1972_portrait_of_Mao-rel:seeAlso http://dbpedia.org/resource/Mao_Zedong; ent:USD-1-73E7-rel:hasValue ent:USD-1-73E7; ent:USD-1-73E7-rel:currency ent:USD; ent:USD-1-73E7-rel:nominalValue 1.73E7.

At the bottom, there is a control panel with buttons for 'Pre-processing', 'Finance', 'DBpedia', 'Arguments', and 'Triple'. Below these are checkboxes for 'Identified all' and 'Identified correctly'. A 'CONFIRM (Enter)' button is also present. At the very bottom, there are links for 'Previous file (Q)', 'Next file (W)', and 'Logout'.

Figure 4.2: VISUE Evaluation tool.

²<http://www.rybak.in/visue/>

Part II

Information Visualization

Chapter 5

Background

5.1 Introduction

"The purpose of computing is insight, not numbers." - Richard Hamming -

Information visualization is a study of abstract data representation through the use of interactive visual interfaces. Its main purpose is simply to reinforce human cognition, make understanding of complex structures hidden in data easier and, perhaps, provide us with a decision-making tool.

The human vision is a very convenient channel for processing large amounts of data because from all our senses it has the largest bandwidth. It is said that our brain receives around 80% of all information from visual cues [44]. This makes the visual representation a suitable aid for external cognition, as described in [43].

Information visualization is a complex field, which builds on information theory, computer science, statistics, human-computer interaction, graphic design and many other areas of research. It has emerged from the need to get an insight into huge volumes of data, which we have to face more and more often. With the rise of computer era, it is now possible to include advanced forms of interaction and data manipulation into visual presentation. That, with no doubt, helps us to perceive phenomena or patterns in data.

5.2 History

Visual representation of data as such is not a new invention; it has been used for many centuries. However, visualization as a matter of scientific research is relatively new, it dates back to mid-80s.

William Playfair

One of the most influential persons in the field of visualization, widely held to be the father of data graphics [19], is William Playfair (1759-1823). He is believed to be the inventor of the most popular means of data visualization today: the line graph (Fig. 5.1) and the pie chart. Sometimes he is also credited to be the first one to use the bar chart, but two earlier pioneers - Nicole Oresme in 14th century and Joseph Priestley in 18th century - verifiably used the similar method to express quantitative information.

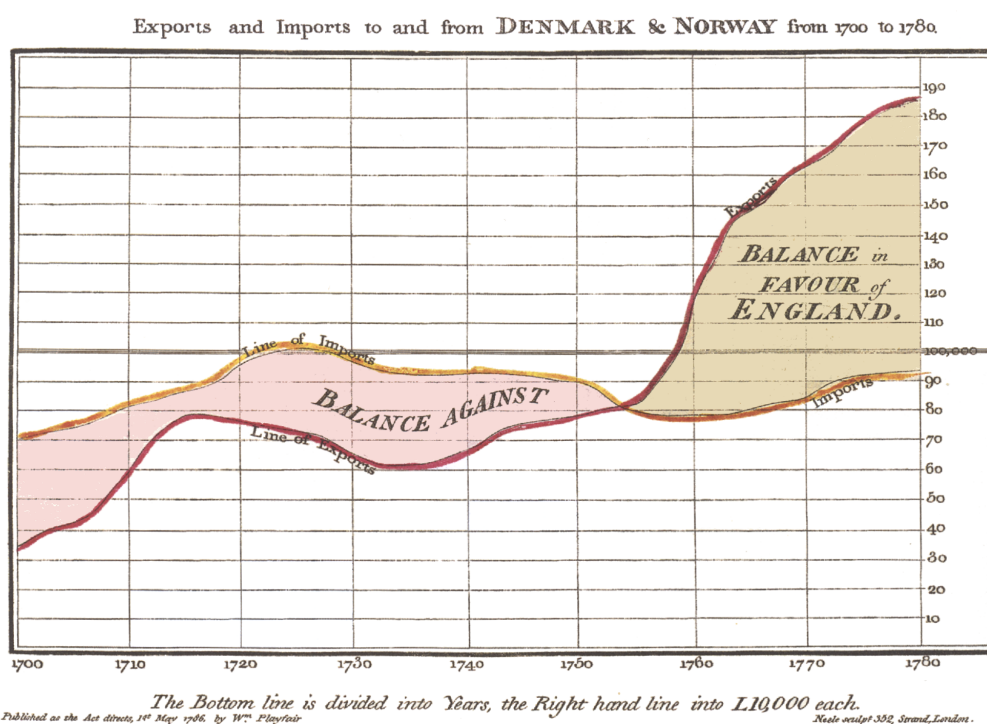


Figure 5.1: Playfair's trade-balance time-series chart (Commercial and Political Atlas, 1786)

Charles Joseph Minard

Another very innovative graphical presentation of data is a flow map depicting Napoleon's march into Russia in 1812 designed by Charles Joseph Minard (1781- 1870). This graph very clearly reveals the successive loss of French Army soldiers, which at some points correlates with the weather temperature. The visualization portrays several variables at once:

- the geographical position of the army with respect to the date

- the size of the army and successive loss of men
- the direction that the army was traveling
- the weather temperature

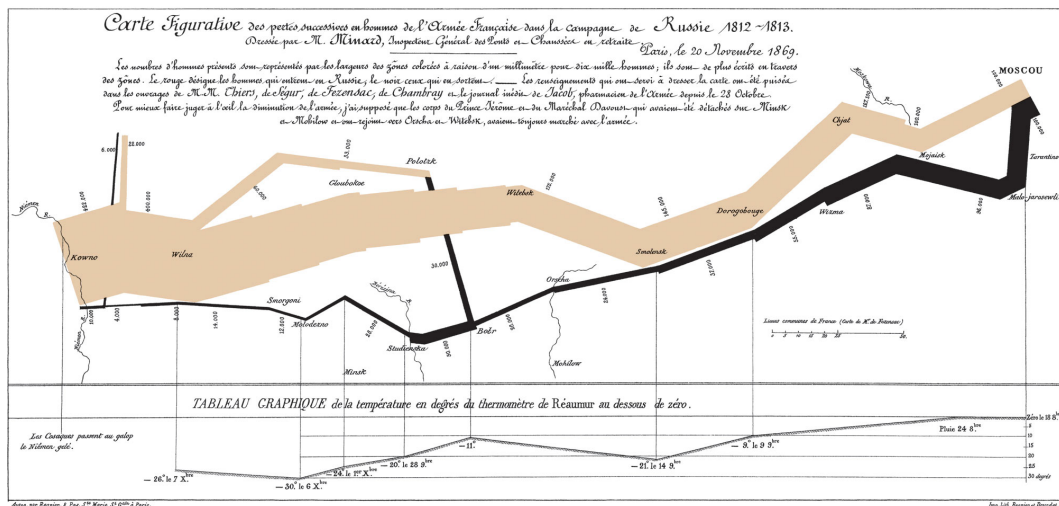


Figure 5.2: Charles Minard's flow map of Napoleon's March (1869)

Despite its age, this visualization was described as *"probably the best statistical graphic ever drawn"* by a contemporary guru in the field - Edward Tufte¹.

¹<http://www.edwardtufte.com/tufte/posters>

Chapter 6

Visualization Methods

Understanding the nature of data that we want to visualize is absolutely essential for the right choice of the visualization method. In the scope of this thesis we analyse financial costs and thus we are interested in the methods of quantitative information presentation. From the graphical and human perception point of view this matter was thoroughly examined in Edward Tufte's book called *Visual Display of Quantitative Information* [48]. For a recent overview of visualization techniques and concepts we would recommend to peruse the book *Data Points: Visualization That Means Something* [53] by Nathan Yau.

In this section, we give introduction to basic visualization concepts. Although we examined suitability of more visualization methods, we will limit our explanation just to the treemap layout (Section 6.2) which we actually applied in the final application.

6.1 Process

Nathan Yau [53] describes the process of data exploration for the purpose of visualization (see Figure 6.1) as a consequence of answering the following questions in the given order:

1. What kind of data are we dealing with?
2. What do we want to know about the data?
3. Which method should we use to visualize it?
4. Are we getting the desired outcome from the visualization?

If the result does not reflect our initial ideas, we should consider improving one of the steps of the whole process. For instance, if we want to implement chronological ordering of data while there is no temporal information in the source data set, we have to improve the process of data acquisition. Otherwise, we would have to reconcile to leaving this feature out.

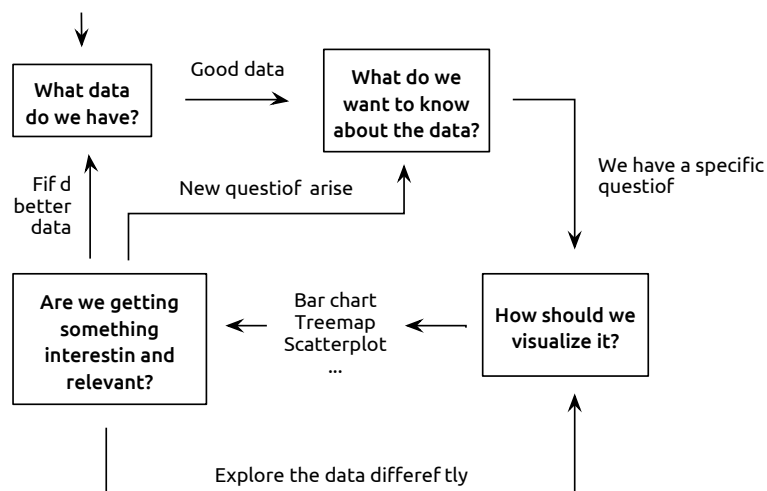


Figure 6.1: Iterative process of the data exploration for the visualization purpose introduced by Nathan Yau.

6.2 Treemap

The treemap is a form of visualization intended to represent large sets of quantitative data using a planar space-filling approach [46]. Its target field are usually hierarchically organized (tree-structured) data sets. This method was originally designed in 1990s by Ben Shneiderman¹ to show files and their sizes on a hard drive [46]. Nowadays its usage is widely diversified.

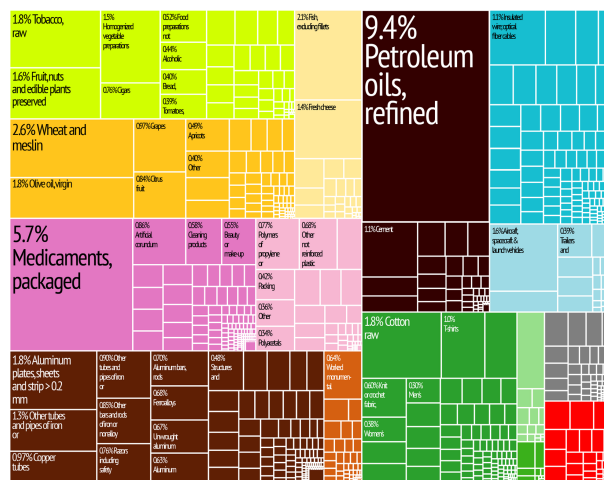


Figure 6.2: Treemap depicting Greece's product exports. (Source: Wikipedia)

The main principle lies in dividing the main visualization area into rectangles. The area of the rectangles proportionally corresponds to some quantifiable quality of data, for example a financial cost of the particular data item. If there are nested items (child nodes), they are visualized by further segmentation of the subarea (the parent's node area). Another dimension of the data can be encoded by assigning the node a background color. This is often used to express an affiliation to various categories. The obvious advantage of treemaps is their capability to use visualization space efficiently (100% coverage of the area).

6.2.1 Layout Algorithms

The area-segmentation (tiling) algorithm ([46], [7]) is a very important aspect of the treemap method itself. Various ways exist to layout the rectangles. The final visual perception, which can be called *readability*, is strongly dependent on choosing the suitable one. A common metrics to evaluate the readability of the map is the aspect ratio r of

¹<http://www.cs.umd.edu/hcil/treemap-history/>

rectangles R . The best possible algorithm is expected to produce map's average ratio close to 1 (square):

$$\forall r \in R, \frac{\max(\text{width}(r), \text{height}(r))}{\min(\text{width}(r), \text{height}(r))} \approx 1 \quad (6.1)$$

The overview of the most important algorithms is given in [18]. We present two methods important for this thesis.

Slice and Dice

Slice and Dice is the original implementation [46] of the treemap visualization method. For each level of the hierarchy the layout alternates between horizontal and vertical rectangles. The advantage of this algorithm is that it preserves the order of the elements and its performance, since only one traversal through the tree is needed, is $O(n)$. Though, the main drawback is that the average ratio is very high and the final treemap contains many thin rectangles.

Squarified Layout

The problem that this method is addressing is: how to recursively tile an area into rectangles, so that their aspect ratio is as close to 1 as possible? As mentioned by the authors of the algorithm - Bruls, Huizing and Wijk - there is a very large set of possible tessellations that makes this problem NP-hard [7]. Nevertheless, we seek for good enough solution, not the optimal one.

In simple terms, the goal of this algorithm is to create square-like rectangles for sibling items from one level, which makes it easier to fit items in in the second iteration. The algorithm starts by deciding whether the first division has to be in the horizontal or vertical direction - it depends which side of the initial rectangle is longer. Assuming that the horizontal side is longer, the first item is fitted to the left part of a given area and the aspect ratio is computed. Next, the second rectangle is added above the first one, which modifies its aspect ratio. If the ratio is closer to 1, another item is added above the previous two. If the ratio is worse, the last added item is removed and aligned to the right side. The algorithm is depicted in the Figure 6.3 [7] and described by a pseudo-code (Listing 6.1) from [18].

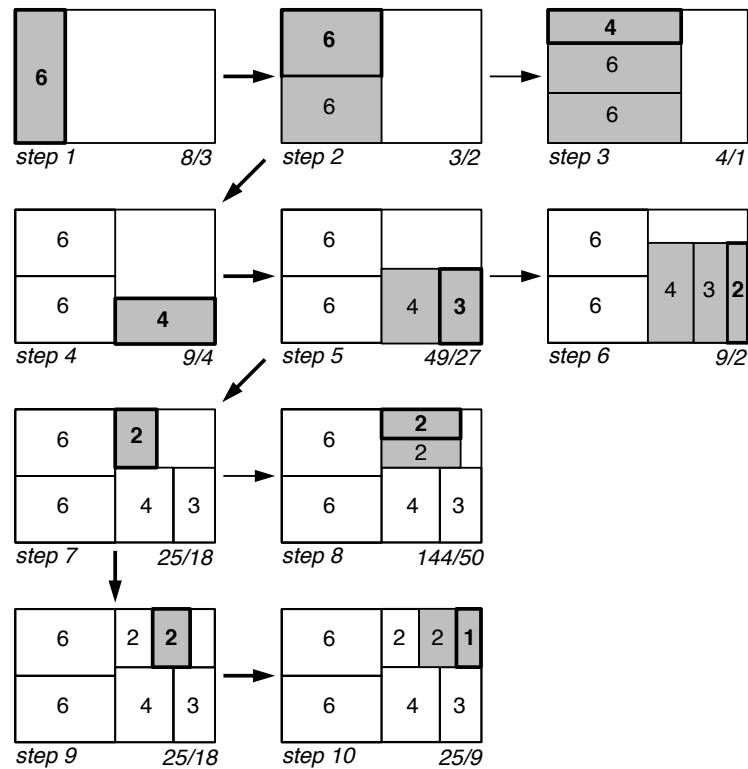


Figure 6.3: Squarified Layout area segmentation.

Squarified Layout advantages:

- Square items are easy to read.
- Square items are better for area comparison.
- Number of pixels used for the border (if > 0) is proportional to the item's area.

Listing 6.1: Squarified Layout pseudo-code

```

squarify(Queue nodes) {
  Queue currentRow;
  nodes.sort(); // Sort on size

  while {nodes.length > 0} {
    Item current := nodes.dequeue();
    if (worst(currentRow + current) < worst(currentRow))
      currentRow.enqueue(current);
    else {
      layoutRow(currentRow);
      currentRow.clear();
    }
  }
}

```

```
        currentRow.enqueue(current);
    }
}

foreach (Item node in nodes)
    squarify(node.children);
}
```

Chapter 7

Realization

We decided to implement the visualization part of the thesis as a web application. We believe that it is the most straightforward option to make it accessible for the general public. The application structure and technologies used are described in the following sections.

7.1 Architecture

Our interactive visualization is a stand-alone web application, in other words, it is not interconnected with the extraction system (except of the data storage). The main benefit of this approach is the universality of the application - provided that we keep a defined format of input data, it does not matter how the data are acquired.

Architecture of the system is depicted on the following figure:

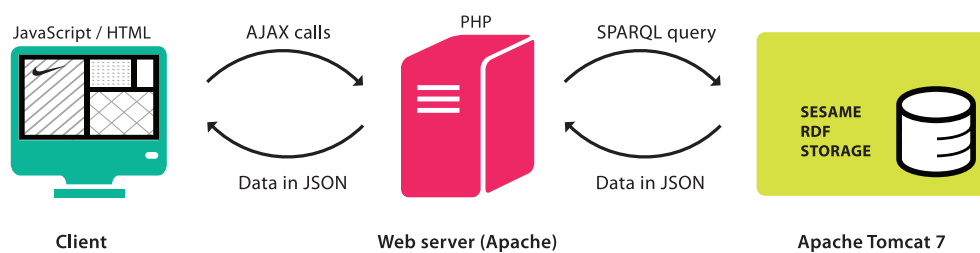


Figure 7.1: Architecture of the visualization system

7.2 Technology

The static part of the application is written in HTML5 and CSS3 which is today's standard. The application logic is mostly implemented in JavaScript while utilizing many libraries with the latest version of `jQuery` at the first place. In addition to that we use `accounting.js`¹ for money formatting, `tablesorter`² for sorting table columns, `FileSaver.js`³ for saving exported files in the browser and `seed-random.js`⁴ to implement the function of random number generation according to a given seed value.

We found a suitable framework for data visualization, `D3.js`, which we describe more in section 7.2.2. During the implementation we ran into a problem with the *Same Origin Policy* (SOP). This security concept prevents scripts from one domain (origin) to interact with a resource from the second one. In our case, it protected our JavaScript from accessing RDF storage and receiving data directly. As we did not want to rely on the fact, that the RDF storage has to be running in the same domain as the visualization application, we bypassed this obstacle by creating a PHP proxy script (`loadJSON.php`).

7.2.1 SPARQL

As mentioned above, the application receives data from RDF storage⁵. When a user specifies search parameters and submits the form, a SPARQL query is constructed on the client side and sent over HTTP to the RDF storage endpoint. The user is not disturbed by the page being reloaded, because we use asynchronous Ajax calls. Once the request is sent, the application is expecting results in the form of a *JSON*⁶ file.

¹<http://josscrowcroft.github.io/accounting.js/>

²<http://tablesorter.com/>

³<https://github.com/eligrey/FileSaver.js>

⁴<http://davidbau.com/encode/seedrandom.js>

⁵Sesame 2.7.0.

⁶JavaScript Object Notation

Listing 7.1: Simplified example of the SPARQL query with maximal cost filter set to 1000.

```
...prefixes...

SELECT DISTINCT ?id ?date ?currency ?cost ?url
WHERE {
  ?id rdf:type inf:information .
  ?id xsd:date ?date .
  ?id rdfs:seeAlso ?url .

  ?id rel:relation ?rel .
  ?rel rdf:predicate pred:hasValue .
  ?rel rdf:object ?object .

  ?rel rdf:subject ?money .
  ?money money:value ?cost .

  FILTER(?cost <= 1000)
}
```

7.2.2 Data-Driven Documents (D3.js)

D3.js⁷ is a very popular open-source JavaScript library for data visualization. It was designed and written by Mike Bostock [5] during his PhD studies at Stanford Visualization Group⁸. D3.js is a framework based on *Document Object Model* (DOM) which solves efficient manipulation of documents based on data [5], as expressed by its author on the project webpage:

"D3 allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. For example, you can use D3 to generate an HTML table from an array of numbers. Or, use the same data to create an interactive SVG bar chart with smooth transitions and interaction."

The importance of D3.js for our application lies in the variety of predefined visualization methods, in its smooth interaction processing and its overall performance.

⁷<http://d3js.org/>

⁸<http://vis.stanford.edu/>

7.3 Interface

In the scope of the following sections, we would like to describe application's interface together with implementation details. Our goal was to design a system, which is easy to navigate and intuitive even for novice users. The initial page is depicted on figure 7.2.

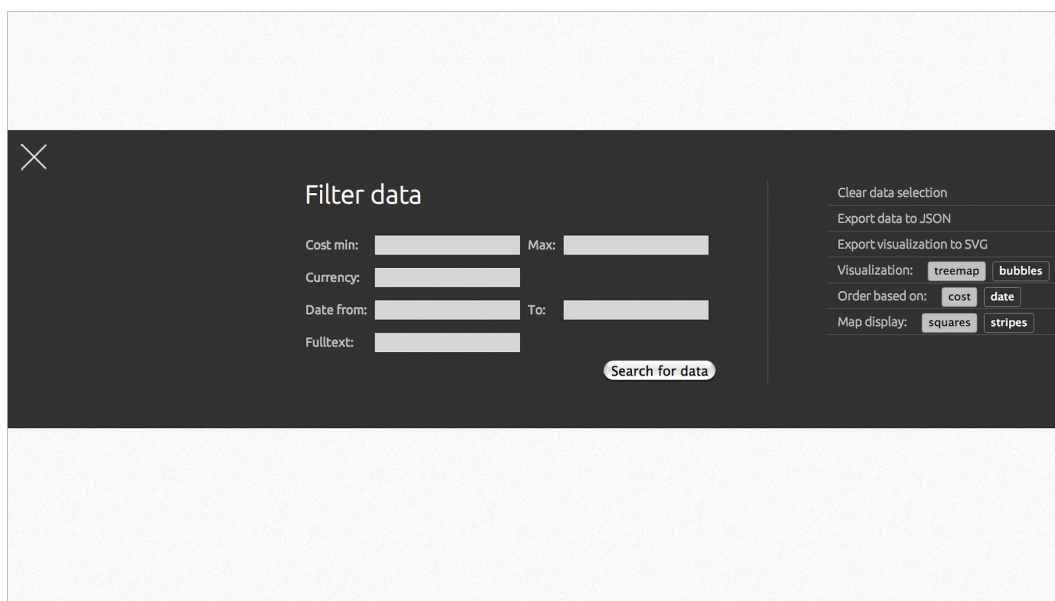


Figure 7.2: Initial page of the application.

7.4 Data Selection

The first step that the user has to take is data selection. The user can influence the data returned from the RDF storage by specification of the following parameters:

Cost range The minimal and maximal financial amounts to be considered.

Currency The currency code (USD, EUR, etc.).

Date range Only information published within the date range will be presented.

Fulltext Only entities with a given keyword in their names are returned.

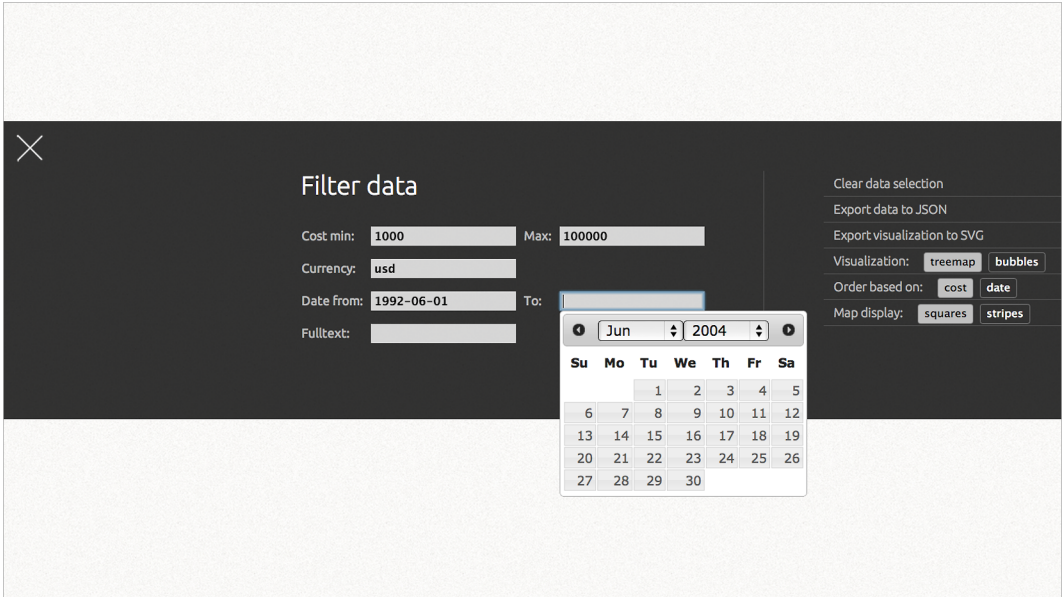



Figure 7.3: User is encouraged to search in data by specifying search parameters.

After successful data search, the menu hides to the left side and the user is provided with tabular data reflecting his searching preferences (Fig 7.4). By clicking on a row, the corresponding data item is marked for visualizing and its background color turns green. Repeating the same action removes the item from the selection. The user can select, or

Entity	Cost	Currency	Date	Selected
 The monthlong trip to Iran	6,000	USD	2007-03-17	<input checked="" type="checkbox"/>
application for a \$6,000 private loan through Wells Fargo to help pay outstanding expenses just before graduation	6,000	USD	2007-06-10	<input type="checkbox"/>
Mount Vernon's combined prize money	6,000	USD	2006-02-12	<input type="checkbox"/>
June rental homes	5,500	USD	2006-05-05	<input type="checkbox"/>
The average diamond ring bought at the Blue Nile site	5,500	USD	2007-01-07	<input checked="" type="checkbox"/>
patch of rail yard next to the New Jersey Turnpike	5,000	USD	2006-02-05	<input type="checkbox"/>
Payton's funeral	5,000	USD	2006-01-01	<input checked="" type="checkbox"/>
plaque	5,000	USD	2006-01-08	<input type="checkbox"/>
Congress details of the financial support from more than 100 people who gave or promised to raise at least \$5,000 for the project	5,000	USD	2001-03-03	

Visualize

Figure 7.4: Selection of items for visualization.

alternatively deselect, all items using the relevant button on the page.

For better understanding of the information in each table row, the user can navigate the mouse cursor over the entity name and the information box where additional details will appear (Fig 7.5).

The image shows a data table on the left and an information box on the right. The table has columns for description, price, and currency. The 'Kalashnikov' row is highlighted in green. The information box for 'Kalashnikov' displays various details including a cost of USD 500, a sentence from a news article, a DBpedia link, a corpus file, a publication date, and an original story link. A 'Visualize' button is located at the bottom right of the information box.

higher-priced model with a larger hard drive	599	USD	Kalashnikov
made-to-measure formal shirt by Charvet of Paris	595	USD	
The trip	560	USD	Cost: USD 500
real thing,	540	USD	Sentence: The activity increased during the second intifada of 2000 when the incentive for taking such risks was high: a Kalashnikov that cost \$500 in Egypt fetched as much as \$2,500 in the Gaza Strip at the time.
10 small-group sessions	540	USD	
500 Howard	512	USD	
bonds	502	USD	DBpedia: http://dbpedia.org/resource/Mikhail_Kalashnikov
apply for work permits and green cards for which they were not eligible	500	USD	Corpus file: 1734366.xml
Kalashnikov	500	USD	Publication date: 2006-01-24
external Rev drive from Iomega	500	USD	Original story: http://query.nytimes.com/gst/fullpage.html?res=9507E7D6133FF937A15752C0A9609C8B63
rice farmer from Thailand	500	USD	
information leading to the arrest and conviction of whoever vandalized the mural	500	USD	

Figure 7.5: Information box with details about the particular data item.

Confirming the data selection by pushing the **"Visualize"** button transforms the interface into fullscreen data visualization. An icon for menu/settings is always present on the browser's left side. In order to open the menu, the user has to click on the icon (Fig. 7.7).

7.5 Visualization Methods

Treemap Layout

The implicit visualization method is the treemap based on squarified algorithm (Section 6.2.1) with descending cost order. Since we do not use categorization of data, no special meaning is assigned to the color. Its value is based on a random number. Our visualization is build on top of `d3.layout.treemap()` layout implementation in `D3.js`.



Figure 7.6: Implicit visualization - treemap with ordering by cost.

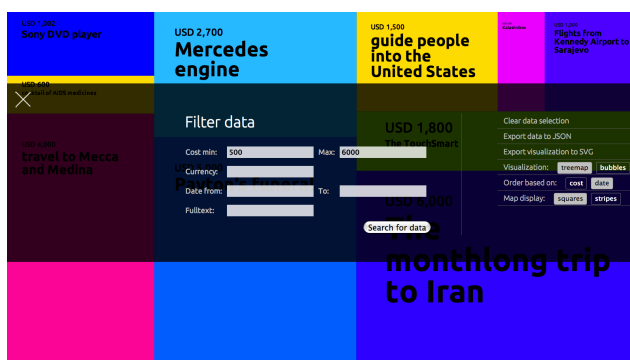


Figure 7.7: Visualization method/settings can be modified in the menu.

Data items in the treemap can be also ordered by date of information publication. The oldest information is positioned in the top left corner and the most recent one is in the bottom right corner (see Figure 7.8).

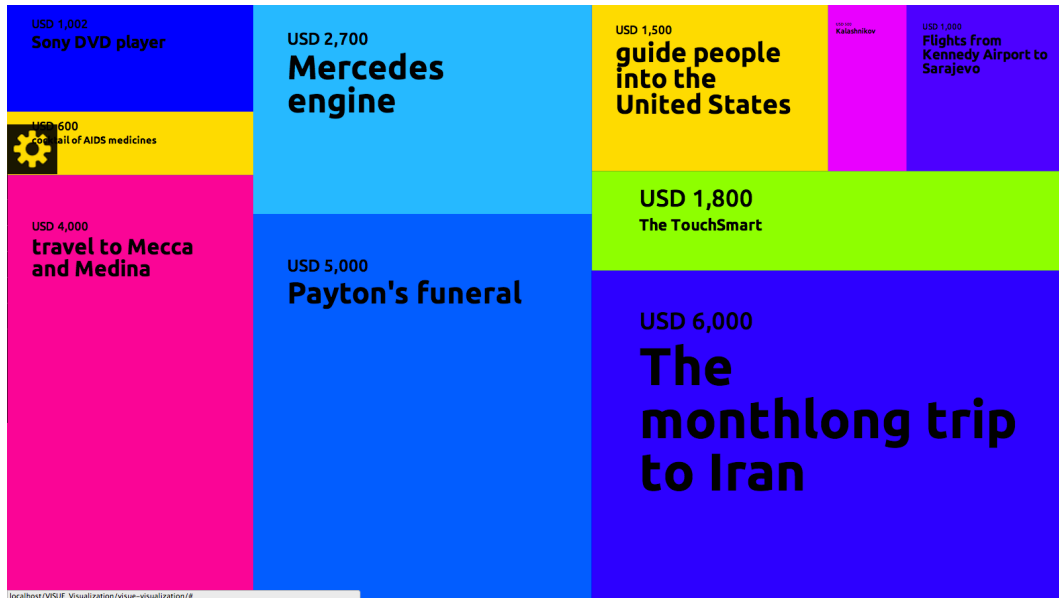


Figure 7.8: Treemap with chronological ordering.

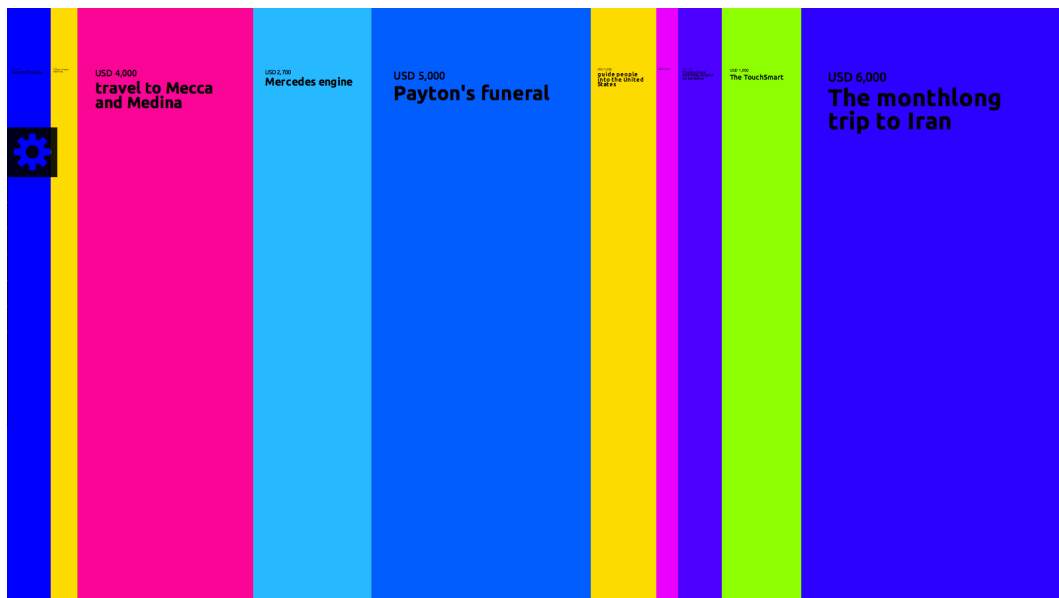


Figure 7.9: Treemap using dice algorithm is convenient when ordering by date.

However, for the chronological ordering the slice-dice algorithm is more suitable. Each piece of information occupies one vertical line and the global overview is much clearer. This mode is labeled as **"stripes"** in the menu.

Circle Pack Layout

In addition to the treemap layout, our application offers visualization using circles with different radii. The size of each circle corresponds with the entity cost and its position is given by the type of ordering - by the date or the cost. Examples (Figure 7.11, 7.12) are depicted on the following page.

7.6 Interactivity

We included several possibilities of interactivity. Zooming is activated when scrolling the mouse wheel and dragging when user holds the left mouse button and moves it. These features are helpful especially when the differences among item costs are significant.

Another very important feature is the box with detailed information about an entity. It is displayed upon clicking on the data item (Fig. 7.10).

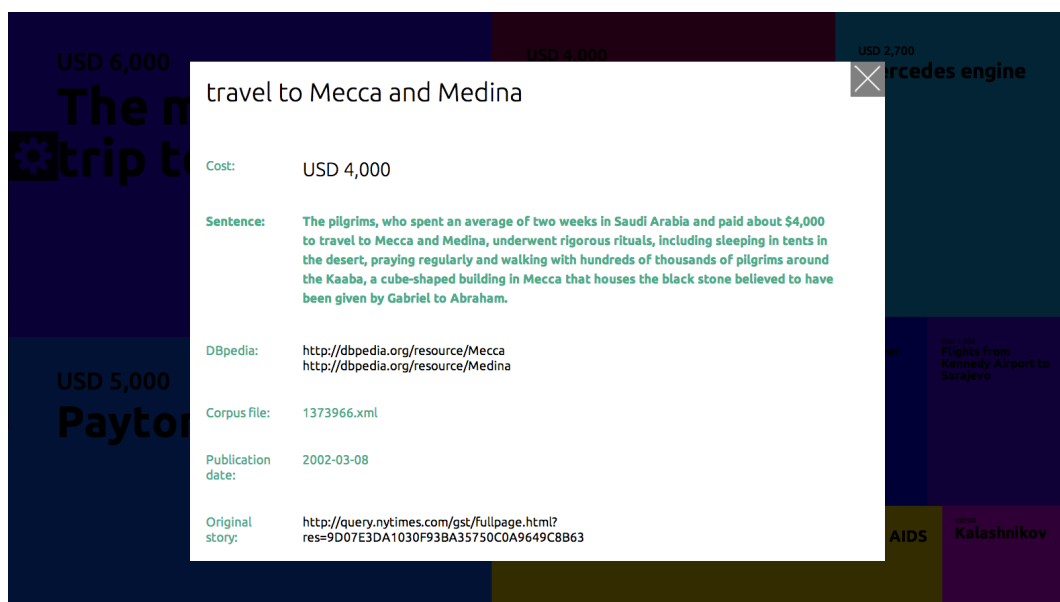


Figure 7.10: Detailed information is shown when user clicks on a data item.



Figure 7.11: Circle Pack Layout and data ordered by cost.



Figure 7.12: Circle Pack Layout presenting large amount of data.

7.7 Data Format and Export

Supported file formats for data export are:

- **JSON**
- **SVG**

The JSON format exported has the same structure as the JSON file processed in our application.

Listing 7.2: Simplified example of JSON processed in the application.

```
{
  "name": "VISUE-Data",
  "results": {
    "bindings": [
      {
        "id": {"value": "node17plbpkc6x7531"},
        "object_value": {"value": "Atari collection"},
        "cost": {"value": "30000.0"},
        "currency": {"value": "USD"},
        "date": {"value": "2007-06-15"},
        "dbp": {"value": "http://dbpedia.org/.../Atari"},
        "file": {"value": "1854539.xml"},
        "url": {"value": "http://query.nytimes.com/..."},
        "sentence": {"value": "But that does mean that the Atari ..."}
      },
      ...
    ]
  }
}
```


Chapter 8

Conclusion and Future Work

8.1 Conclusion

The subject of this thesis was motivated by the following question:

Can we build a system that will be able to automatically extract and visualize costs of various interesting things?

And the simple answer is yes. We have built an information extraction system, which is processing a text in natural language. We proposed a solution to identify potentially interesting pieces of information. It is based on two main ideas. Firstly, the popular newspapers, such as The New York Times, are relevant source of information for this purpose and secondly, the entities with an article on Wikipedia are widely known and perhaps interesting. We created an ontology of finance-related verbs that helps us with the relationship recognition. The "proof of concept" was successfully implemented and tested.

In the second part of the thesis we focused on the presentation of the extracted data. According to the assignment, we studied and implemented the treemap layout, which is a very convenient way of visualizing quantitative data. In addition, we included some useful features such as data filtering or the second visualization method utilizing circles of various radii to express the subject's cost.

We can conclude that the objective of this thesis was fulfilled. It is worth noting that we contributed to the GATE framework by providing previously unimplemented modules for the semantic role labelling and the DBpedia entity labelling. In the section about the future work we present some ideas how the current system can be enhanced.

8.2 Future Work

Several suggestions for the improvements of the system have emerged during the work on the thesis. The development of the current version was challenging and time-consuming so that we were not able to implement the enhancements stated below. However, we believe that any of them would add up to the overall quality and/or usability.

8.2.1 Information Extraction part

Coreference resolution

Since many sentences are introduced by a pronoun referring to the subject of previous sentence, this method would substantially increase the number of correctly extracted triples.

Entity linking

We suppose that by using machine-learning based methods for linking the subject's name with a DBpedia entity would produce more precise results.

Twitter data input

As we discussed in Section 3.4.2, data from Twitter (tweets) are very promising source of the text input. They contain complete information in 140 characters and it is easy to select just the profiles providing some relevant news.

DBpedia Spotlight recognition by the whole article

Due to the long processing times of DBpedia Spotlight tool we are processing just single sentences containing financial information. If we were able to process the whole articles, the entity disambiguation would be more accurate.

8.2.2 Visualization

Linked Data filtering

Data filtering based on the entity type (person, company, etc.) would be a very useful feature in the visualization.

Finance flow visualization

With enough data processed by our system, it would be possible to visualize financial flows between individual entities. For instance, being able to see money spent and earned by Google company could be interesting.

Data categorization

Graphical objects could be colored by their affiliation to various categories (time range, predicate type, location, etc.).

List of Figures

1.1	Billion Dollar o-Gram by David McCandless	3
1.2	Mock-up of our visualization	4
2.1	SRL representation of two sentences.	16
2.2	Example of a role set from NomBank.	17
2.3	Example of an entry in PropBank for predicate acquire.01	18
2.4	Example of an ontology - BBC Sports Ontology	21
2.5	The Linking Open Data cloud diagram - datasets that have been published in Linked Data format.	23
3.1	Pipeline architecture of the system	28
3.2	Example of a file from The New York Times Corpus	31
3.3	Monetary information recognition	33
3.4	Annotated and normalized money data in GATE	34
3.5	Financial verb contained in the ontology is recognized.	39
3.6	Financial relationships ontology.	40
3.7	Detail of financial relationships ontology.	40
3.8	Annotated sentence in GATE.	41
3.9	Example of originator (named entity) identification.	42
3.10	Entities linked with DBpedia.	42
3.11	DBpedia Spotlight online demo.	44
3.12	Directory structure of DBpedia Spotlight Tagger on Git.	46
3.13	Example of noun phrase chunking.	48
3.14	SRL representation of two sentences.	51
3.15	Example of SRL web interface.	52
3.16	SRL module in GATE.	53
3.17	SRL annotation in GATE.	53

3.18	Role set for predicate <code>buy.01</code>	54
3.19	Example of extraction pattern for predicate <code>acquire.01</code>	54
3.20	Information represented as an RDF triplet.	55
3.21	Sesame Web Interface	57
3.22	RDF repository	57
4.1	Distribution of sentences in subsequent phases of the extraction.	60
4.2	VISUE Evaluation tool.	61
5.1	Playfair's trade-balance time-series chart (Commercial and Political Atlas, 1786)	64
5.2	Charles Minard's flow map of Napoleon's March (1869)	65
6.1	Iterative process of the data exploration for the visualization purpose intro- duced by Nathan Yau.	67
6.2	Treemap depicting Greece's product exports. (Source: Wikipedia)	68
6.3	Squarified Layout area segmentation.	70
7.1	Architecture of the visualization system	72
7.2	Initial page of the application.	75
7.3	User is encouraged to search in data by specifying search parameters.	76
7.4	Selection of items for visualization.	76
7.5	Information box with details about the particular data item.	77
7.6	Implicit visualization - treemap with ordering by cost.	78
7.7	Visualization method/settings can be modified in the menu.	78
7.8	Treemap with chronological ordering.	79
7.9	Treemap using dice algorithm is convenient when ordering by date.	79
7.10	Detailed information is shown when user clicks on a data item.	80
7.11	Circle Pack Layout and data ordered by cost.	81
7.12	Circle Pack Layout presenting large amount of data.	81

List of Tables

2.1	Difference between grammatical relations and semantic roles.	16
2.2	Example of Brown corpus part-of-speech tags [2].	20
3.1	Short (American) and Long (European) scale notation for large numbers. . . .	35
4.1	Sources of inaccuracy in extraction of triples.	59
4.2	Number of sentences in each phase of the extraction (year 2006).	60
4.3	VISUE Extraction Tool keyboard shortcuts.	61

Appendix A

Extraction XML File

Listing A.1: Arguments assigned to the predicate 'buy.01' in an output XML

```
<extractionResult>
  <sentence>Google bought YouTube in October for $1.65 billion.</sentence>
  <extractedData>
    <predicate roleSet="buy.01" lemma="buy">bought</predicate>
    <arguments>
      <argument apredType="A0" s="0" e="6">
        <string>Google</string>
        <spotlight>
          <entity uri="http://dbpedia.org/resource/Google" s="0" e="6">
            Google</entity>
          </spotlight>
        </argument>
      <argument apredType="A1" s="14" e="21">
        <string>YouTube</string>
        <spotlight>
          <entity uri="http://dbpedia.org/resource/YouTube" s="14" e="21">
            YouTube</entity>
          </spotlight>
        </argument>
      <argument apredType="AM-TMP" s="22" e="32">
        <string>in October</string>
      </argument>
      <argument apredType="A3" s="33" e="50">
        <string>for $1.65 billion</string>
        <finance>
          <money currency="USD" value="1.65E12">$1.65 billion</money>
        </finance>
      </argument>
    </arguments>
  </extractedData>
</extractionResult>
```

```
</arguments>  
</extractedData>  
</extractionResult>
```

Appendix B

User Manual

In this section we will describe how to install and run programs developed in the scope of this thesis.

B.1 Visue Extraction Tool

The main application for the extraction is a console program written in Java, that is why Java Runtime Environment has to be present on the destination system. We included executable **jar** file, so there is no need to compile it. This application requires two external tools to be running:

DBpedia Spotlight

DBpedia Spotlight¹ is essential for named entity annotation. DBpedia Spotlight server can run either locally or remotely. This software can be used under the terms of the Apache Licence, 2.0.

Mate Tools - Semantic Role Labelling

Another inseparable part of the application is SRL tool. Latest version can be found on the enclosed CD. We have prepared all the configuration files, therefore running `run_http_server.sh` script should start the server at `http://localhost:8072/parse`

Our application is started by the following command:

```
java -jar Visue.jar
```

¹<https://github.com/dbpedia-spotlight/dbpedia-spotlight>

Supported arguments are:

-i , --inputDir

Path to the directory with input files. If this argument is not provided, default directory is considered as the source folder. Default directory is: `app_root_folder/documents/input`

-o , --outputDir

Path to the directory with the results of the extraction are stored. Default directory is: `app_root_folder/documents/extraction_results` .

-ds , --dbpediaSpotlight

This arguments expects an address of the DBpedia Server. Default value is DBpedia demo address at `http://spotlight.dbpedia.org/rest/annotate/` .

-srl , --semanticRoleLabeling

This parameter serves for setting the address of Semantic Role Labeling server.

B.2 Sesame RDF Uploader

This application is used for uploading extracted RDF files to the RDF storage (Sesame). We strongly recommend to use the latest version (currently: `openrdf-sesame-2.7.0`) of the storage, because there were serious problems with SPARQL queries in the previous versions.

The application is started by the following command:

```
java -jar RdfUploader.jar
```

and accepts these arguments:

-a , --address

The address of the repository server.

-r , --repository

The identifier (name) of the repository.

-i , --inputFolder

Folder, where the input RDF files are expected. By default, this is `app_root_folder/rdf/`

B.3 Visualization

As was already mentioned, the visualization is implemented as a web application. It requires a web server supporting PHP programming language. Since this application is obtaining data from an RDF storage, there has to be one on disposal and its address has to be specified in the file `loadJSON.php` .

Running demo can be seen at: <http://rybak.in/visue/>.

B.4 RDF storage

We have good experience with the latest release of Sesame RDF storage². It requires Apache Tomcat 7 and as long as there is no need for storing data in database, no installation or configuration is required.

²<http://www.openrdf.org/>

Appendix C

Digital Attachments

This thesis is accompanied with a DVD containing the source codes, example data and third-party software.

Hierarchy of the DVD:

/	
├	Code
├	├ SesameRdfUploader Tool for uploading RDF triples to storage.
├	├ VisueEvaluationTool Tool evaluating extraction results.
├	├ VisueExtraction Information Extraction application.
├	├ VisueVisualization Visualization web application.
├	└ ExtractedData Examples of extraction results.
├	├ RDF
├	├ XML
├	└ New York Times Corpus Example Modified version of a file from NYTC.
├	└ Ontology Ontology of finance-related verbs
├	└ Thesis Source files of this thesis.
├	└ Third Party Software
├	├ Sesame RDF storage RDF storage - version 2.7.0.
├	├ SRL-MateTools Pre-configured SRL tool.

Bibliography

- [1] *MUC6 '95: Proceedings of the 6th conference on Message understanding*, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics.
- [2] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly, Beijing, 2009.
- [3] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
- [4] Anders Björkelund, Love Hafdell, and Pierre Nugues. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task, CoNLL '09*, pages 43–48, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
- [5] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011.
- [6] David Bounie and Laurent Gille. International production and dissemination of information: Results, methodological issues, and statistical perspectives. *International Journal of Communication* 6, 2011.
- [7] Mark Bruls, Kees Huizing, and Jarke van Wijk. Squarified treemaps. In *In Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42. Press, 1999.
- [8] Chia-Hui Chang, Mohammed Kayed, Moheb Ramzy Girgis, and Khaled Shaalan. A survey of web information extraction systems, 2006.
- [9] Chaomei Chen. *Information Visualization Beyond the Horizon, Second Edition*. Springer-Verlag London Limited, 2006.
- [10] Fei Chen, Xixuan Feng, Christopher Re, and Min Wang. Optimizing statistical information extraction programs over evolving text. In Anastasios Kementsietsidis and Marcos Antonio Vaz Salles, editors, *ICDE*, pages 870–881. IEEE Computer Society, 2012.
- [11] Kenneth W. Church and Mark Y. Liberman. A status report on the acl/dci. *The Proceedings of the 7th Annual Conference of the UW Centre for the New OED and Text Research: Using Corpora*, pages 84–91, 1991.

- [12] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 999888:2493–2537, November 2011.
- [13] Jim Cowie and Wendy Lehnert. Information extraction. *Commun. ACM*, 39(1):80–91, January 1996.
- [14] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, 2002.
- [15] D. Damjanovic and K. Bontcheva, editors. *Named Entity Disambiguation using Linked Data*, May 2012.
- [16] Tim Berners-Lee David Beckett. Turtle - terse RDF triple language, W3C team submission, 2008. See: <http://www.w3.org/TeamSubmission/turtle/>.
- [17] Padhraic Smyth David J. Hand, Heikki Mannila. *Principles of Data Mining*. The MIT Press, 2001.
- [18] BJÖRN Engdahl. Ordered and unordered treemap algorithms and their applications on handheld devices. *Master's degree project, Department of Numerical Analysis and Computer Science, Stockholm Royal Institute of Technology, SE-100*, 44, 2005.
- [19] Benjamin J. Fry. *Computational information design*. PhD thesis, School of Architecture and Planning, Massachusetts Institute of Technology, Cambridge, Mass., USA, 2004.
- [20] Ralph Grishman. *Information Extraction: Capabilities and Challenges*. Rovira i Virgili University, Tarragona, Spain, January 2012.
- [21] Ralph Grishman and Beth Sundheim. Message understanding conference - 6: A brief history. In *Proceedings of the International Conference on Computational Linguistics*, 1996.
- [22] Mark Hepple. Independence and commitment: Assumptions for rapid training and execution of rule-based pos taggers. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong, October 2000.
- [23] Jiewen Huang, Daniel J. Abadi, and Kun Ren. Scalable sparql querying of large rdf graphs. *PVLDB*, 4(11):1123–1134, 2011.
- [24] Roman Klinger and Katrin Tomanek. Classical Probabilistic Models and Conditional Random Fields. Technical Report TR07-2-013, Department of Computer Science, Dortmund University of Technology, December 2007.
- [25] Ryan McDonald, Fernando Pereira, Seth Kulick, Scott Winters, Yang Jin, and Pete White. Simple algorithms for complex relation extraction with applications to biomedical ie. In *In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 491–498, 2005.

- [26] Pablo Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *In the Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*, 2011.
- [27] Pablo N. Mendes, Max Jakob, and Christian Bizer. Dbpedia for nlp: A multilingual cross-domain knowledge base. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May 2012.
- [28] Marie-Francine Moens. *Information Extraction: Algorithms and Prospects in a Retrieval Context (The Information Retrieval Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [29] E. Nist Ac. Automatic Content Extraction 2008 Evaluation Plan (ACE08), April 2008.
- [30] Natalya F. Noy and Deborah L. McGuinness. Ontology development 101: A guide to creating your first ontology. Online, 2001.
- [31] David D. Palmer. Text pre-processing. In Nitin Indurkha and Fred J. Damerau, editors, *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group, Boca Raton, FL, 2010. ISBN 978-1420085921.
- [32] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Comput. Linguist.*, 31(1):71–106, March 2005.
- [33] podcasts.s3.amazonaws.com. Sir tim berners-lee talks with talis about the semantic web, 2012.
- [34] E. Prud'hommeaux and A. Seaborne. Sparql query language for rdf. w3c recommendation 15 january 2008., 2008.
- [35] Eric Prud'hommeaux and Andy Seaborne. SPARQL query language for rdf. *W3C Recommendation*, 4:1–106, 2008.
- [36] V. Punyakanok, D. Roth, and W. Yih. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2), 2008.
- [37] Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. *CoRR*, cmp-lg/9505040, 1995.
- [38] Ellen Riloff and Jeffrey Lorenzen. Extraction-based text categorization: Generating domain-specific role relationships. In Tomek Strzalkowski, editor, *Natural language information retrieval*, pages 167–196. Kluwer Academic Publishers, Dordrecht, NL, 1999.
- [39] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*, chapter Constraints Satisfaction Problems, pages 137–160. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, 2003.
- [40] Tárík Saleh Salem. Automatické zodpovídání dotazů založené na sumarizaci textů. Master's thesis, ZÁPADOČESKÁ UNIVERZITA V PLZNI, Fakulta aplikovaných věd, 2012.
- [41] Evan Sandhaus. The new york times annotated corpus catalogue.

- [42] Sunita Sarawagi. Information extraction. *Found. Trends databases*, 1(3):261–377, March 2008.
- [43] Mike Scaife and Yvonne Rogers. External cognition: How do graphical representations work? *INTERNATIONAL JOURNAL OF HUMAN-COMPUTER STUDIES*, 45:185–213, 1996.
- [44] A. Seiderman, S.E. Marcus, and D. Hapgood. *20/20 is not enough: the new world of vision*. Knopf, 1989.
- [45] Satoshi Sekine and Chikashi Nobata. Definition, Dictionaries and Tagger for Extended Named Entity Hierarchy.
- [46] Ben Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph.*, 11(1):92–99, January 1992.
- [47] J.J. Thomas and Eds. K.A. Cook. *Illuminating the Path: Research and Development Agenda for Visual Analytics*. IEEE Press, 2005.
- [48] Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, USA, 1986.
- [49] Jonathan J. Webster and Chunyu Kit. Tokenization as the initial phase in nlp. In *COLING*, pages 1106–1110, 1992.
- [50] Daya C. Wimalasuriya and Dejing Dou. Ontology-based information extraction: An introduction and a survey of current approaches. *J. Information Science*, 36(3):306–323, 2010.
- [51] Fei Wu and Daniel S. Weld. Autonomously semantifying wikipedia. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 41–50, New York, NY, USA, 2007. ACM.
- [52] Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. Texrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, NAACL-Demonstrations '07*, pages 25–26, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.
- [53] N. Yau. *Data Points: Visualization That Means Something*. John Wiley & Sons, Inc., 2013.