WIRTSCHAFTSUNIVERSITÄT WIEN
BAKKALAUREATSARBEIT

Titel der Bakkalaureatsarbeit:
Tagging Technologies

Englischer Titel der Bakkalaureatsarbeit:
Tagging Technologies

VerfasserIn:	Ing., Gerald Weber
Matrikel-Nr.:	0125536
Studienrichtung:	Wirtschaftsinformatik
Kurs:	0175 IT Vertiefungskurs VI
Textsprache	Englisch
BetreuerIn:	Dr. Dipl.-Ing. Mag., Albert Weichselbraun

Abstract

The continous growth of data available, calls for efficient algorithms to extract sensible information. Tagging technologies therefore deal with the identification and classification of relevant parts of structured and unstructured data. Analysis of data in their embedded context is more complex but leads to higher quality conclusions about the content than other approaches.
This paper deals with different tagging technologies and describes their main fields of application as well as their implementations in different programming languages and usage in frameworks.

Zusammenfassung

Da die Datenmengen die laufend anfallen immer umfangreicher und größer werden, benötigt man effiziente Algorithmen, um sinnvolle Informationen extrahieren zu können. Tagging Technologies befasst sich deshalb mit der Identifizierung und Kategorisierung von interessierenden Teilen strukturierter sowie unstrukturierter Datenbasen. Analysen der Daten in ihrem Kontext sind weit komplexer, führen aber zu qualitativ hochwertigen Aussagen über den Inhalt. Der Inhalt dieser Arbeit widmet sich verschiedenen Tagging Verfahren und beschreibt deren häufigste Anwendungsgebiete sowie Implementierungen in verschiedenen Programmiersprachen und Verwendung in Frameworks.

Key Words
tagging technologies, text mining, text analysis, information retrieval, information extraction

Contents

1 Introduction

While text reading and understanding is relatively simple for human beings, computers struggle with each word. On the other side, humans are not used to summarize millions of documents to get an overview of knowledge available worldwide. So computers, that can process large information in a much faster way should have the ability to build knowlegde databases from unstructured data (each form of written natural language which contains no further semantic information) where humans can select the relevant documents without having to read through all documents. But developing these linguistic and statistical algorithms for computers should turn out as a tedious task which covers the academic fields of data mining, information retrieval & extraction as well as artificial intelligence.

This paper will not cover the algorithms developed to discover additional information in unstructured data, but present some software implementations that have the competence to resolve this problem. Some of them are developments that resolve the problem whereas others provides a combination of existing tools adding only few enhancements but with the target to distribute natural language processing (NLP) technology to everybody.

The second chapter presents some information and history about information retrieval and tagging in the field of text processing. Then, chapter three will reveal some free available implementations for the task of text mining and describe them in detail. Additional to each implementation, a simple task of processing text is evaluated. Chapter four summarises the main characteristics of the implementations in form of a table where the gentle reader can hopefully identify an implementation long-needed at first glance. The last section summarises the paper and tries to identify the one-and-only software package that covers all parts of text mining.

2 Information Extraction Background

"Information extraction (IE) 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, providing additional aids to access and interpret the unstructured data by information systems" [1]. The process of generating written text starts with an idea at an abstract level that needs to be split up into smaller parts which are reflected in a set of sentences. The components of these parts textit"are translated into a set of semantic roles, which are in their turn translated in a set of grammatical and lexical concepts" [1]. After the convertion into character sequences the result can be written down. Information extraction therefore can be seen as the inversion of this process - extract relevant information form a set of words, sentences and documents. Information retrieval (IR) is the more general form of IE and assumes that ''we already have a need for information that we are able to formulate, and then find relevant items in a store (collection) of items" [2].

Before Salton first defined retrieval systems in 1966 (rephrased 1986) as *"The SMART retrieval system takes both documents and search requests in unrestricted English, performs a complete content analysis automatically, and retrieves those documents which most nearly match the given request"* [3], the United States military indexed german scientific research documents. In the 1950 America again started research in mechanized literature search because knowledge or more precise, lack of knowledge about russian technology motivated them. The 1960s and 70s saw a boom in IR research which developed the fundamental measurements for the quality of IR systems: precision and recall. Further advances in computer technology allowed systems feasable of full text searches that pushed commercial application. The next two decades brought plenty of IR algorithms i.e. the probabilistic model, the vector space model and the fuzzy logic model. At the end of the 1980s Tim Berners-Lee developed a hypertext system which is broadly known as the World Wide Web (WWW). This turned out to be a new challenge for IR because of the awesome growth of the system (185,497,213 web sites in January 2009 counted by netcraft.com).

Before 1990, IR systems proofed their function under laboratory conditions and had little commercial impact. But the spread of the WWW lead to new IR systems, the web search engines. These systems were designed to process more information than ever before and should end-users help to find information spread all over the world.

While first engineers researched to retrieve the documents containing the information they were looking for, it became clear, that their goal has to be the understanding of the natural language. One task of NLP is part-of-speech tagging or simply *tagging*. This task covers text annotation with additional information (meta information) that is not as obvious to the computer as for humans, but essential for the analysis of the informational background, the meaning of the reviewed text. *"The widespread interest in tagging is founded on the belief that many NLP applications will benefit from syntactically disambiguated text."* [4]. At the beginning of POS tagging, the accuracy of an early deteministic rule-based tagger (in principle if-then statements) labled only 77% of the words correctly (Greene and Rubin 1971). Nowadays, POS taggers can achieve an accuracy between 96% and 97% [4]. *"The insight that tagging is an intermediate layer of representation that is useful . . . is due to the corpus linguistics work that was led by Francis and Kučera at Brown University in the 1960s and 70s"* [4]. They first published the "Brown corpus"[1] in 1964 which is a manually annotated representation of present-day edited American English which *"consists of 1,014,312 words of running text of edited English prose printed in the United States during the calendar year 1961"* [5].

This corpora is spilt into different domains and is used to train POS taggers. Most modern taggers pass through two phases:

- a learning phase is used to build up a probability model that can be applied in
- the extraction phase do determine the correct part-of-speech of the text.

As you can imagine, not every word can be covered by the Brown corpus (many other corpora for specific domains are also available) which leaves place for improvements. So it is not surprising, that many papers present and discuss different POS algorithms, models and implementations trying to find the ideal pathfinder for succeeding IE tasks.

[1]Brown corpus http://khnt.aksis.uib.no/icame/manuals/brown/

3 Available Implementations

This chapter covers some important implementations of tagging technologies. It includes a general description, a list of their features, their distribution characteristics and links to get further information. At the end of each description a short getting started guide will help you to get an insight to the according toolkit. The better part of the implementations are downloadable, open source software projects whereas others are web services (TermExtractor, OpenCalais) which can only be used online. Additionally to the web services, only TreeTagger does not reveal its sources. Another detail, worth to keep in mind is, that many toolkits implements tagging algorithms on their own, whereas some others consists of a collection of readily implemented software components with good documentation to provide easy access to NLP (natural language processing) technologies.

3.1 Rapidminer

RapidMiner is the successor of YALE and its developer Rapid-i claims that it is *"is the world-wide leading open-source data mining solution due to the combination of its leading-edge technologies and its functional range."* [6] It is written in Java and is

Figure 3.1: Rapidminer Logo

distributed under the Affero GPL. RapidMiners features can be accessed through a command line, a grafical user interface (GUI) as well as by a Java API (application programming interface).

A main feature of RapidMiner, winner of the Open Source business Foundation Award 2008 [7] is *"the modular operator concept which allows the design of complex nested operator chains for a huge number of learning problems in a very fast and efficient way (rapid prototyping)"* [6]. The company behind RapidMiner is the open-source data mining specialist Rapid-i that enables other companies to use leading-edge technologies for data mining and business intelligence [6]. Rapid-i, which is based in Dortmund, Germany, distributes an enterprise and a community version that is currently available in version 4.3 for windows and linux. The enterprise version is available as an annual subscription service which *"gives you the highest level of Rapid-i support and access to a worldwide acting team with decades of experience supporting the most demanding enterprise deployments"* [6]. Rapid-I writes as follows: *"Enterprise Edition = Community Edition + More Features + Services + Guarantees"* [6]. The core benefits of the paid version in comparision to the community project would be additional operators, support and ticket system, guaranteed response times as well as a bug fixing guarantee.

The main features of RapidMiner are [6]

- freely available open-source knowledge discovery environment
- 100% pure Java (runs on every major platform and operating system)
- Knowledge discovery (KD) processes are modeled as simple operator trees which is both intuitive and powerful
- operator trees or subtrees can be saved as building blocks for later re-use
- internal XML representation ensures standardized interchange format of data mining experiments
- simple scripting language allowing for automatic large-scale experiments
- multi-layered data view concept ensures efficient and transparent data handling
- Flexibility in using RapidMiner:
  - GUI for interactive prototyping
  - command line mode (batch mode) for automated large-scale applications
  - Java Application Programming Interface (API) to ease usage of RapidMiner from your own programs
- simple plugin and extension mechanisms, a broad variety of plugins already exists and you can easily add your own
- powerful plotting facility offering a large set of sophisticated high-dimensional visualization techniques for data and models
- more than 400 machine learning, evaluation, in- and output, pre- and post-processing, and visualization operators plus numerous meta optimization schemes

- machine learning library WEKA (see section 3.5) is fully integrated
- RapidMiner was successfully applied on a wide range of applications where its rapid prototyping abilities demonstrated their usefulness, including text mining, multimedia mining, feature engineering, data stream mining and tracking drifting concepts, development of ensemble methods, and distributed data mining.

While knowledge discovery is a highly complex process it requires careful analysis, specification, implementation and testing. Rapid prototyping is an important part because it helps to identify adequate methods and parameters enabling developers to make crucial design decisions as early as possible [8].

Furthermore, RapidMiner supports a large number of plugins for all aspects of data mining, eg. it uses meta operators to reduce effort spent to adjust each single step, adds a large assortment of visualisation techniques to represent the output and provides functionality to place breakpoints after each operator.

Operators are used in RapidMiner for any analysis tasks. It can be seen as a functional unit that *"receives its input, performs a defined action and delivers some output"* [8]. Furthermore it is possible to connect operators in a way that the outcome of an operator can be used as input for the next operator which enables more complex experiments.

Over 400 operators are available and covers different fields of application [6]:

- In- and output: for different formats including
  - Arff, C4.5, CSV, Excel files, datasets from databases, text files
- Machine learning algorithms: learning schemes for regression, classification and clustering tasks
  - Support vector machines, Decision trees and rule learners, Bayesian learners, Meta learning, Clustering
- Data preprocessing: useful for dataset preparation
  - Discretization, Normalization, Sampling, Dimensionality reduction
- Feature operators
  - Feature selection, Feature weighting and relevance, Feature construction
- Performance evaluation: schemes to estimate performance
  - Cross-validation, Training and test set splitting, Significance tests
- Meta operators: optimization operators for experiment design
  - Parameter optimization, Learning curves, Experiment loops and iterations
- Visualization: for logging and presenting results

  - 1D, 2D and 3D plots, Built-in color histogram and distribution plots, Quartile / box plots, Lift charts

While RapidMiner includes WEKA it additionally provides operators for WEKA learning schemes and attribute elevators.

In the case of missing functionality, it is possible to implement and provide specialized operators. The forthcoming plugin system provides interfaces for enhancements from different authors, *"e.g. plugins for time series processing or text document to word vector transformation"* [6]. At the moment there are four groups of plugins available:

- Text plugins for input texts in different formats
- Value plugins for series methods for feature extraction from series data - includes loaders for audio files
- Data stream plugins for data stream mining and for learning drifting concepts
- CRF (conditional random field) plugins for named entity recognition

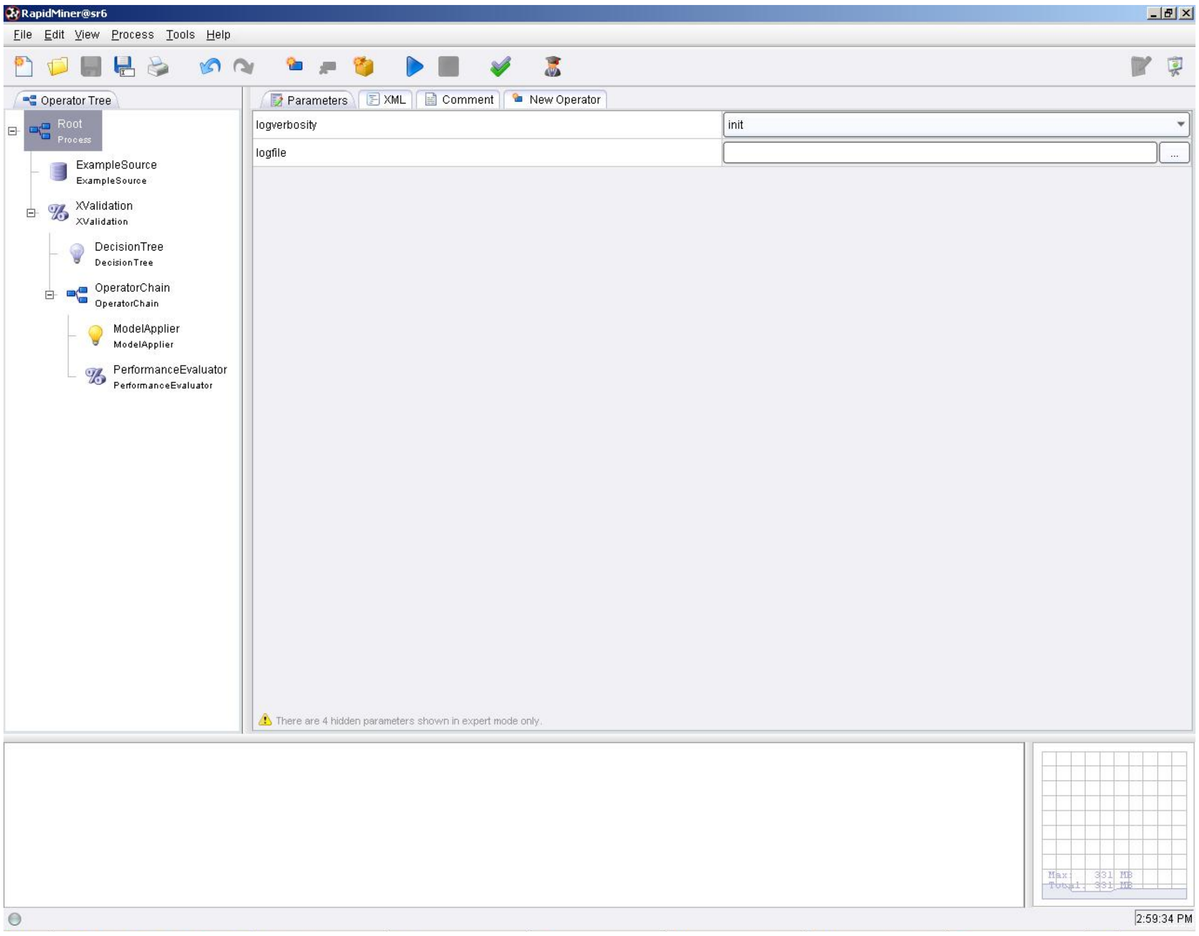
"RapidMiner's most important characteristic is the ability to nest operator chains and build complex operator trees. In order to support this characteristic, the RapidMiner data core acts like a data base management system and provides a multi-layered data view concept on a central data table which underlies all views" [6]. *"For example, the first view can select a subset of examples and the second view can select a subset of features. The result is a single view which reflects both views. Other views can create new attributes or filter the data on the fly. The number of layered views is not limited"* [6].

The homepage of RapidMiner[1] provides an installation guide, a tutorial, a GUI manual and finally a Java API documentation. The documents show very detailed information for novices beginning data mining with RapidMiner. For a quick overview Rapid-i recorded a screencast (interactive tour) to get in touch with RapidMiner. For a more detailed introduction, Rapid-i offers courses for beginners as well as for machine learning experts.

Some global, well-known companies like Ford, IBM, HP, Cisco, BNP Paribas uses RapidMiner. One of the references on the Rapid-I homepage points to the Austrian organisation "Mobilkom Austria". It uses datamining to categorize and forward their 80.000 monthly incoming, spam cleaned customer emails to the according departments [9].

After downloading the correct package from Rapid-i[2] and extracting the packed archive, I started Rapidminer by typing *java -jar rapidminer.jar* in the lib subdirectory. This

[1] http://rapid-i.com/
[2] http://rapid-i.com/content/view/26/84/lang,de/



logverbosity init

logfile

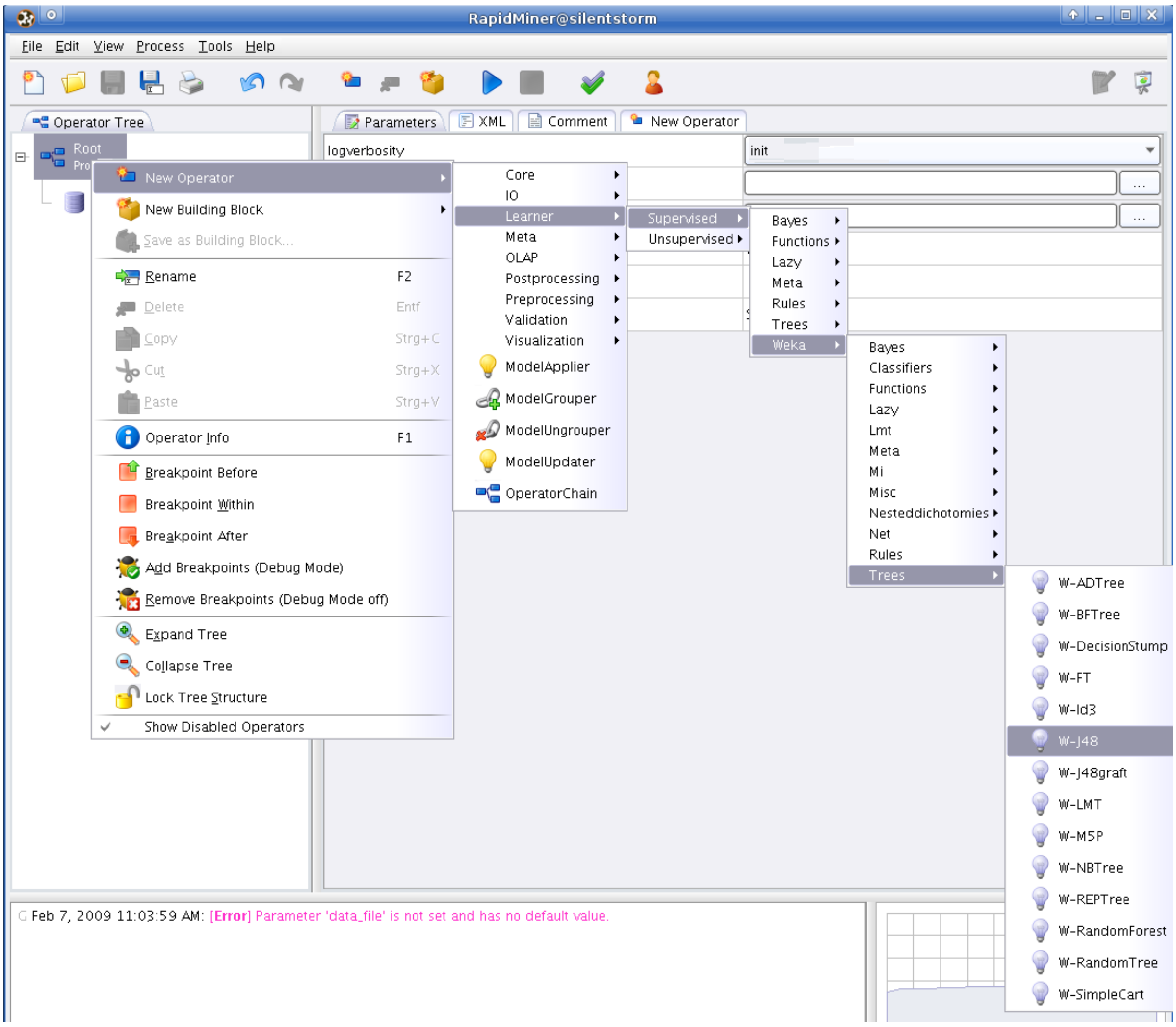
There are 4 hidden parameters shown in expert mode only.

Max: 331 MB
Total: 331 MB

2:59:34 PM

Figure 3.2: Rapidminer screenshot: Main gui

initializies the Rapidminer GUI. Then I opened the provided screencast and followed the instructions to get in touch with the program. I created a operator chain consisting of an ARFF input source and a J48 Decision Tree.

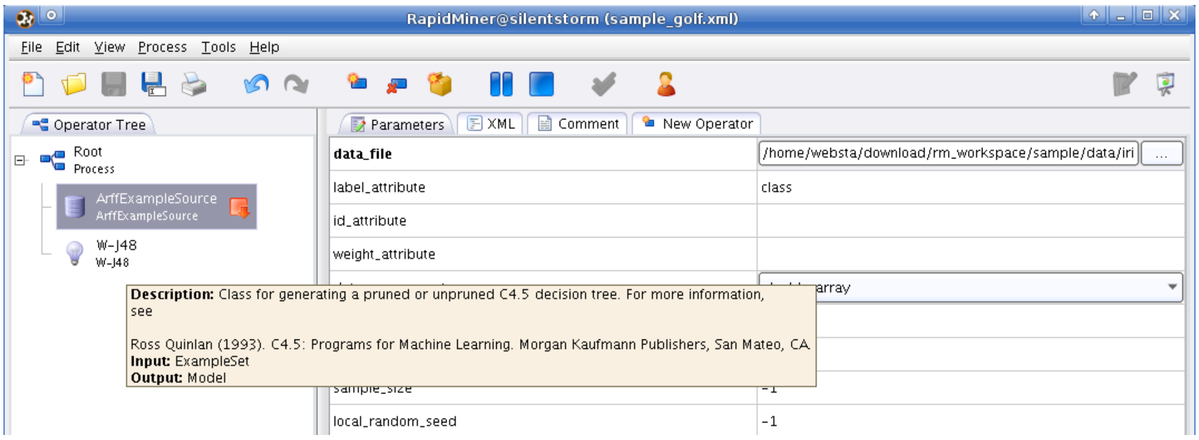


Feb 7, 2009 11:03:59 AM: [Error] Parameter 'data_file' is not set and has no default value.

Figure 3.3: Rapidminer screenshot: Adding a new operator to the tree

After running the classification I tried some visualisation plugins to present the results.

Furthermore it is possible to set breakpoints before / after each operator and watch at the data processed so far. All these test can also be achieved by using the command line or the Java API. To screencast was easy to follow although it was recorded with an earlier version.

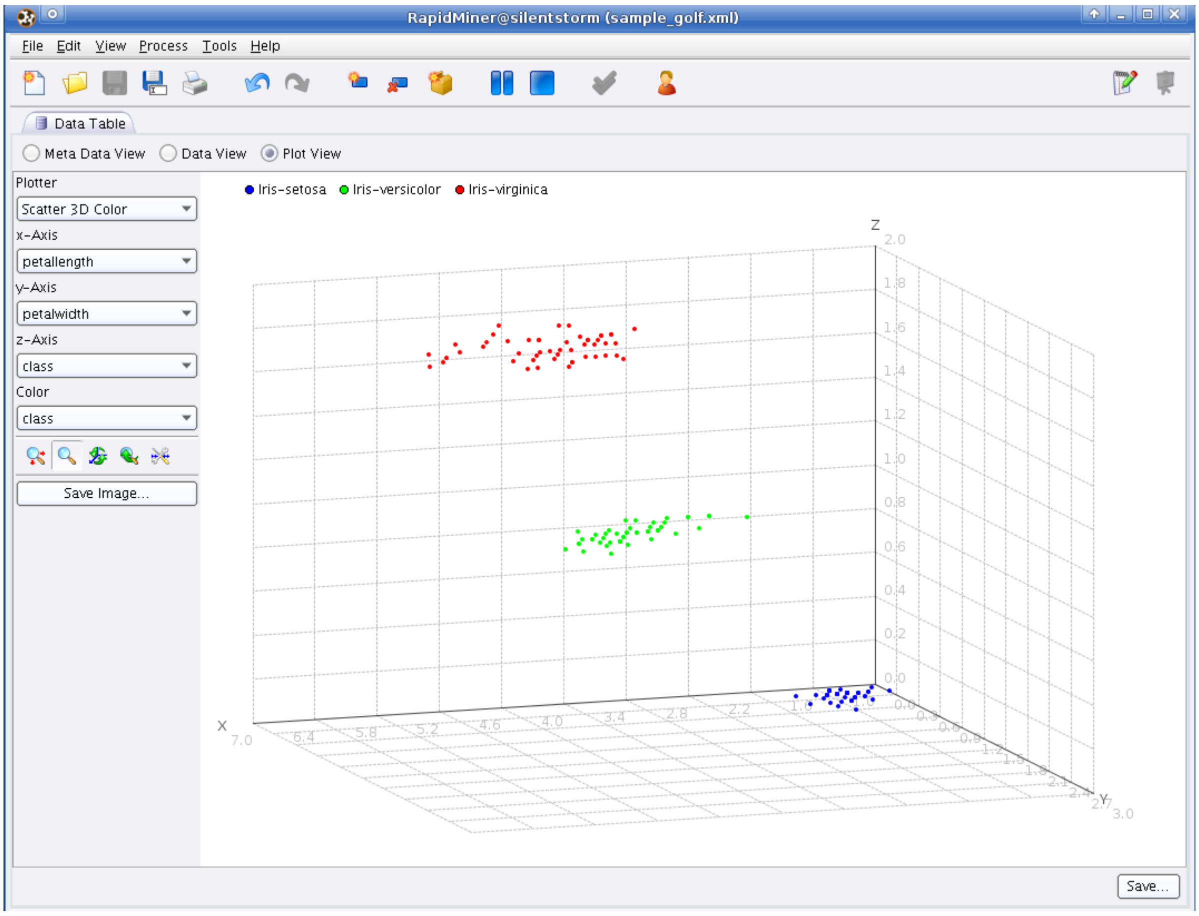


data_file	/home/websta/download/rm_workspace/sample/data/iri
label_attribute	class
id_attribute	
weight_attribute	
[illegible]	array
[illegible]	[illegible]
[illegible]	[illegible]
sample_size	-1
local_random_seed	-1

Description: Class for generating a pruned or unpruned C4.5 decision tree. For more information, see

Ross Quinlan (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA.
Input: ExampleSet
Output: Model

Figure 3.4: Rapidminer screenshot: Operator chain of a classification problem



Plotter: Scatter 3D Color
x-Axis: petallength
y-Axis: petalwidth
z-Axis: class
Color: class

Figure 3.5: Rapidminer screenshot: Scatter 3D plot of classification result

3.2 Natural Language Toolkit NLTK

NLTK is a collection of python modules and corpora for statistical natural language processing. Python and this toolkit are released under the GPLv2 and allow any programmer to get started with NLP tasks without having to spend too much time on gathering resources. *"The most important advantage of using NLTK is that it is entirely self-contained. Not only does it provide convenient functions and wrappers that can be used as building blocks for common NLP tasks, it also provides raw and pre-processed versions of standard corpora used in NLP literature and courses"* [10]. To use this toolkit, python 2.4 is required but the team around the main developers Steven Bird, Edward Loper and Ewan Klein already plan to migrate to python 3.0 in the year 2009.

For active development with NLTK, *"it provides basic classes for representing data relevant to natural language processing; standard interfaces for performing tasks such as tokenization, part-of-speech tagging, and syntactic parsing; and standard implementations for each task which can be combined to solve complex problems"* [11].

NLTK was designed with four primary goals in mind [11]:

- Simplicity - the developers *"tried to provide an intuitive and appealing framework along with substantial building blocks"* [11]
- Consistency - *"all the data structures and interfaces are consistent"* [11]
- Extensibility - *"the toolkit is organized so that it is usually obvious where extensions would fit into the toolkit's infrastructure"* [11]
- Modularity - simple, well-defined interfaces makes it easier to change and extend the toolkit

The documentation mentions three futher announcements that users should be aware of:

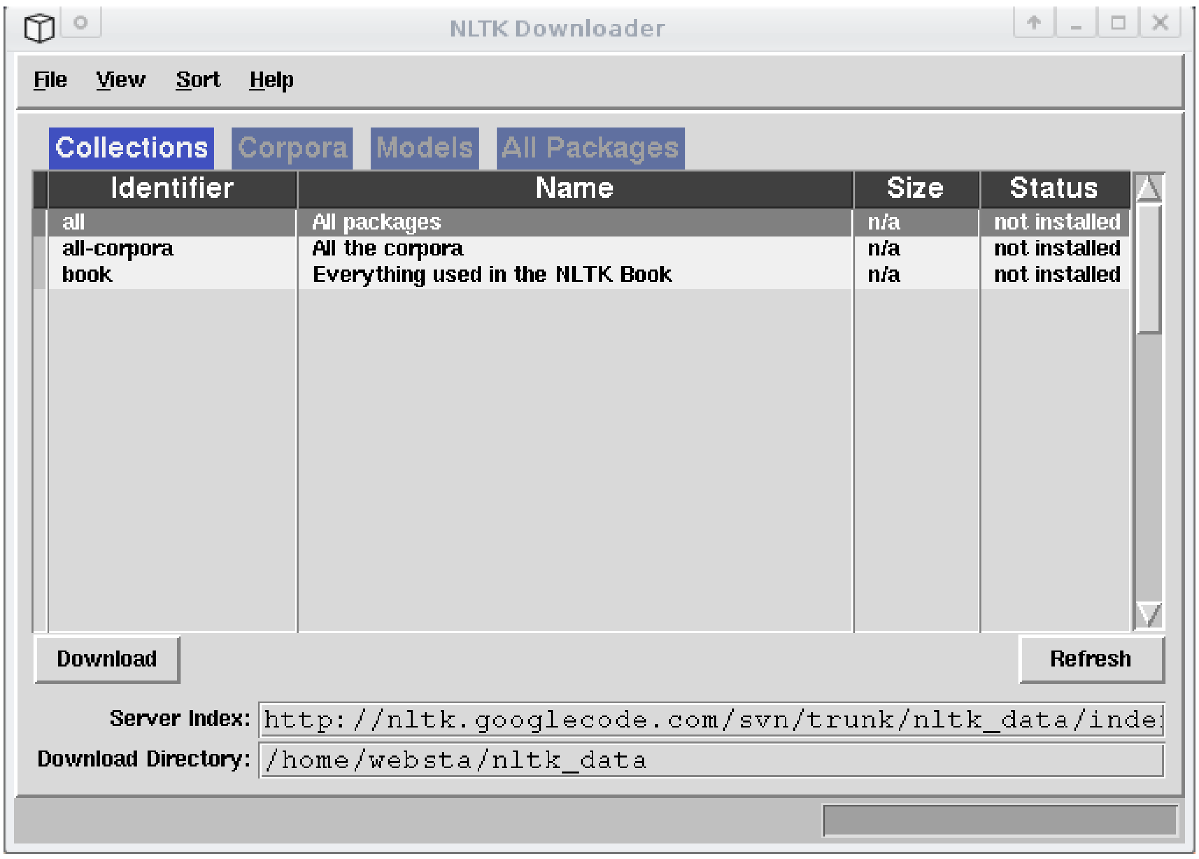
- The toolkit is very mature until now but will further be developed to meet new requirements.
- It does not need to be highly optimized for runtime performance. Suitable algorithms would require the use of programming languages like C or C++ which would make the toolkit less accessible and more difficult to install.
- A clear implementation is more preferable as ingenious yet indecipherable ones.

The get in touch with the project, an online book, which will be published too in mid 2009 is available. Additional installation instructions and how to's are available too.

The website[3] is well arranged and provides downloads for different operating system platforms.

Downloading and extracting of the source package[4] was straight forward. To finish installation open a terminal, change to the directory where the extracted sources are and type *sudo python setup.py install*. To work with corpora[5], these have to be downloaded. The start download of corpora and other features start the python interactive console and type the following statements:

```
>>> import nltk
>>> nltk.download()
```



Identifier	Name	Size	Status
all	All packages	n/a	not installed
all-corpora	All the corpora	n/a	not installed
book	Everything used in the NLTK Book	n/a	not installed

Server Index: http://nltk.googlecode.com/svn/trunk/nltk_data/inde

Download Directory: /home/websta/nltk_data

Figure 3.6: NLTK screenshot: The downloader helps to download additional resources and plugins

[3]http://www.nltk.org/
[4]http://nltk.googlecode.com/files/nltk-0.9.7.zip
[5]http://nltk.googlecode.com/svn/trunk/nltk_data/index.xml

After downloading the all corporas (100 MB) you can work with them. For this example I used a corpora from the NY Times and started to discover named entities[6].

```
>>> from nltk.corpus import ieer
>>> docs = ieer.parsed\_docs('NYT\_19980315')
>>> tree = docs[1].text
>>> print tree
...
 ``It's
  like
  when
  you
  start
  a
  colony,''
  said
  (PERSON Mike Godwin)
  ,
  chief
  counsel
  for
  the
  (ORGANIZATION Electronic Frontier Foundation)
  ,
  an
  advocacy
  group
  that
  only
  a
  (DURATION few years)
  ago
  was
  practically
  alone
  in
  the
  field.
...
```

Then I wanted to work with own texts and decided to download a news article from the NY Times web site. To transform the HTML format into required, tokenized format for

[6]extracting named entities with NLTK: http://nltk.googlecode.com/svn/trunk/doc/howto/relextract.html

the POS Tagger I used the following instructions[7][8][9]

```
>>> from urllib import urlopen
>>> import nltk
>>> from nltk.corpus import brown  // import brown corpus
>>> brown\_news = brown.tagged\_sents(categories='news') // use brown categorie
     news for unigram tagger
>>> tagger = nltk.UnigramTagger(brown\_news)
>>> raw = urlopen("http://www.nytimes.com/2009/02/07/business/economy/07bailout
    .html?hp").read()
>>> text = nltk.clean\_html(raw)  // remove html tags
>>> tokens = nltk.wordpunct\_tokenize(text) // tokenize text
>>> list(tagger.tag(tokens)) // apply unigram tagger
[('New', 'JJ-TL'), ('U', None), ('.', '.'), ('S', 'NN'), ('.', '.'), ('Plan', '
    NN-TL'), ('to', 'TO'), ('Help', 'NN-HL'), ('Banks', 'NNS-TL'), ('Sell',
    None), ('Bad', 'JJ-HL'), ('Assets', None), ('-', None), ('NYTimes', None),
    ('.', '.'), ('com', None), ('Skip', 'NP'), ('to', 'TO'), ('article', 'NN'),
     ('Try', None), ('Electronic', None), ('Edition', None), ('Log', None), ('
    In', 'IN'), ('Register', 'NN-TL'), ('Now', 'RB'), ('Home', 'NN-TL'), ('Page
    ', 'NP'), ('Today', 'NR'),
```

Afterwards I was surprised how simple it was to work with the NLTK package. Installation was easy, the functions well documented on the NLTK web site.

3.3 General Architecture for Text Engineering (GATE)

GATE is a open source Java implementation which is maintained by the Sheffield NLP Group and funded by institutions and other projects like The European Commission, EPSRC (Engineering and Physical Sciences Research Council), AKT (Advanced Knowledge Technology: sponsor of ISWC2005 and ESWC2005), NeOn Project and much more. It *"is an architecture that contains functionality for plugging in all kinds of NLP software, such as POS taggers, sentence splitters, named entity recognizers, etc."* [12]. It is licenced under LGPL and hosted at SourceForge.

The GATE website describes it as [12]

- the Eclipse of Natural Language Engineering, the Lucene of Information Extraction, a leading toolkit for Text Mining

[7]Dealing with HTML: http://nltk.googlecode.com/svn/trunk/doc/en/ch03.html
[8]Tagging: http://nltk.googlecode.com/svn/trunk/doc/howto/tag.html
[9]General how to: http://dtl.unimelb.edu.au/view/action/singleViewer.do?dvs=1234005884999 825&locale=de_DE&sea 3&frameId=1&usePid1=true&usePid2=true

- used worldwide by thousands of scientists, companies, teachers and students
- comprised of an architecture, a free open source framework (or SDK) and graphical development environment
- used for all sorts of language processing tasks, including Information Extraction in many languages
- funded by different research councils, the EU and other commercial users
- 100% Java reference implementation of ISO TC37/SC4 and used with XCES in the ANC
- 10 years old in 2005[12], used in many research projects and compatible with IBM's UIMA
- based on MVC, mobile code, continuous integration, and test-driven development, with code hosted on SourceForge

The framework has benefits for scientists performing experiments with language and computation [12]:

- Repeatability - by making it easier to repeat comparable experiments across different sites and platforms GATE makes it easier to be sure that a particular result is not a glitch
- Quantitative evaluation - GATE includes a built-in system for comparing annotation data on documents and generating quantitative metrics such as precision and recall
- Collaboration - Multi-site collaboration puts a premium on software integration and portability, both areas which GATE-based software excels
- Reuse not reinvention - Language processing resources that have been integrated in GATE are likely to have a longer working life and to be reused more often because using them does not require learning fresh installation and usage conventions for every tool

While GATE components are kinds of Java Beans, these appear in three different shapes[12]:

- Language Resource (LR): refers to data-only resources such as lexicons, corpora, thesauri or ontologies
- Processing Resource (PR): refers to resources whose character is principally programmatic or algorithmic, such as lemmatisers, generators, translators, parsers or speech recognisers. For example, a part-of-speech tagger is best characterised by reference to the process it performs on text. PRs typically include LRs, e.g. a tagger often has a lexicon; a word sense disambiguator uses a dictionary or thesaurus.

- Visual Resources (VRs): visualisation and editing components interacting with the GUI

All resources are packed into JAR[10] files bundled with XML (eXtensible Markup Language) configuration files. Theses resources are known as CREOLE (Collection of REusable Objects for Language Engineering).

The GATE developers (20 active programmers) suggest usage of their framework for the disciplines [12]:

- Computational Linguistics
- NLP
- Language Engineering

Computational linguistics is a part of the science of language that uses computation as an investigative tool. On the other side, NLP is part of the science of computation whose subject matter is data structures and algorithms for human language processing. Finally, Language engineering engages in building language processing systems whose cost and outputs are measurable and predictable.

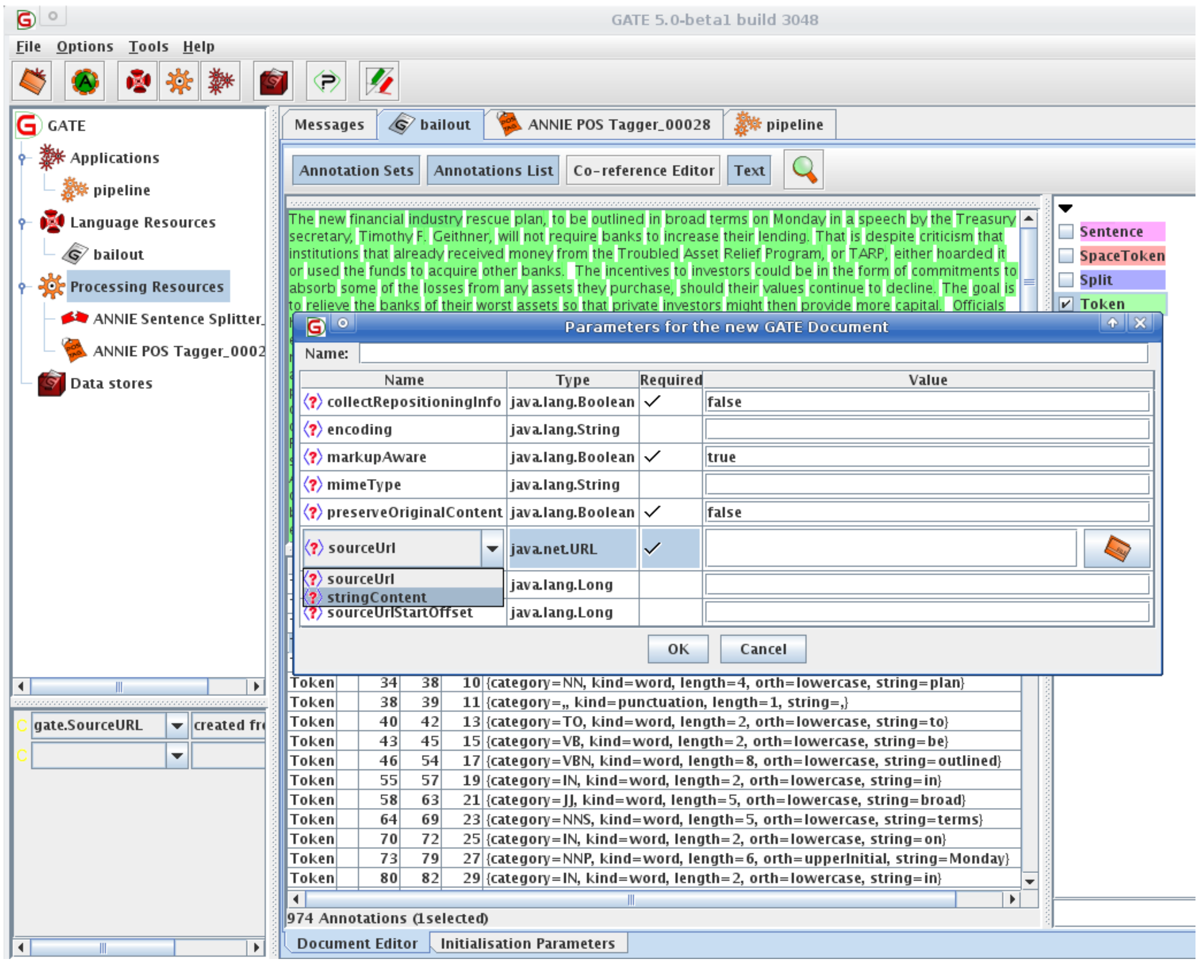
Because it is written in Java it supports each operating environment running Java 5 or newer. The project started in 1996 and uses popular projects like WEKA, TreeTagger, Snowball Stemmer, Google API and others. The website[11] provides different download packages, user guides, a programmers guide, links to academic publications as well as demos and example codes. Furthermore GATE supplies movie tutorials which are screencasts showing GATE in action. Some companies using GATE are AT&T, Master Foods NV, British Gas PLC, Syntalex Ltd., Thompson Corporation.

GATE provides a runnable jar for download. This file includes a grafical installer which is started by calling *java -jar gate-5.0-beta1-build3048-installer.jar*. After the installation, change into the GATE directory and start *./bin/gate.sh* to run the GATE GUI.

Working with the GUI is not as simple as with Rapidminer, but similar. After startup of the application you can see a tree view on the left side. There you can add a new application (pipeline in our case) which should process our source (document bailout, copy & paste from a NY Times article) through a sentence splitter followed by a POS tagger.

After processing the chain of operations our document is annotated. All found words are listed below and the text highlights them. If you choose one word from the list, the annotation of the text begins to blink. The properties on the right side allows you to choose which information should be highlighted.

[10] Java archive
[11] http://gate.ac.uk/



The new financial industry rescue plan, to be outlined in broad terms on Monday in a speech by the Treasury secretary, Timothy F. Geithner, will not require banks to increase their lending. That is despite criticism that institutions that already received money from the Troubled Asset Relief Program, or TARP, either hoarded it or used the funds to acquire other banks. The incentives to investors could be in the form of commitments to absorb some of the losses from any assets they purchase, should their values continue to decline. The goal is to relieve the banks of their worst assets so that private investors might then provide more capital. Officials

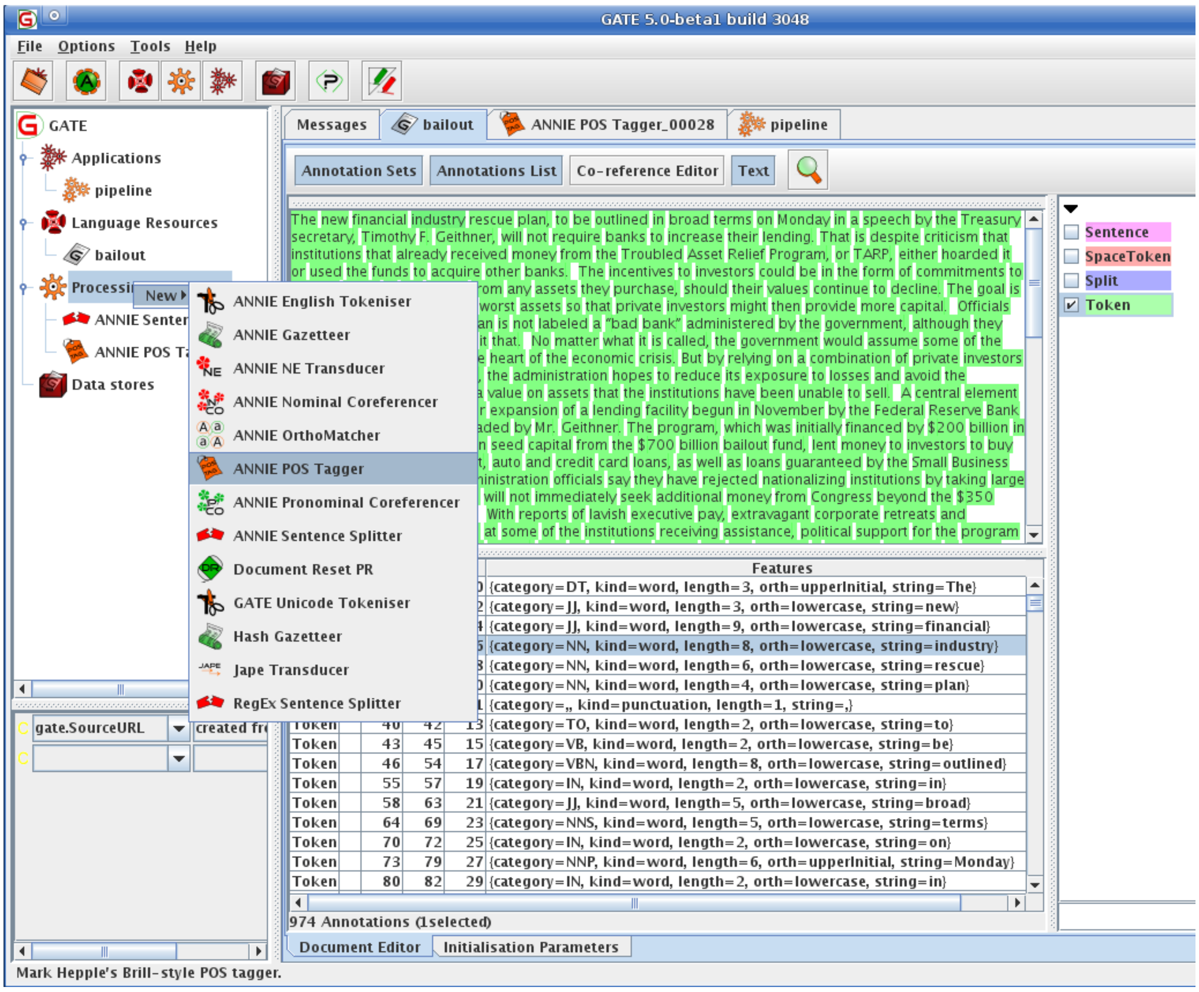
Name:

Name	Type	Required	Value
collectRepositioningInfo	java.lang.Boolean	✓	false
encoding	java.lang.String		
markupAware	java.lang.Boolean	✓	true
mimeType	java.lang.String		
preserveOriginalContent	java.lang.Boolean	✓	false
sourceUrl	java.net.URL	✓	
sourceUrl / stringContent	java.lang.Long		
sourceUrlStartOffset	java.lang.Long		

Token	34	38	10	{category=NN, kind=word, length=4, orth=lowercase, string=plan}
Token	38	39	11	{category=,, kind=punctuation, length=1, string=,}
Token	40	42	13	{category=TO, kind=word, length=2, orth=lowercase, string=to}
Token	43	45	15	{category=VB, kind=word, length=2, orth=lowercase, string=be}
Token	46	54	17	{category=VBN, kind=word, length=8, orth=lowercase, string=outlined}
Token	55	57	19	{category=IN, kind=word, length=2, orth=lowercase, string=in}
Token	58	63	21	{category=JJ, kind=word, length=5, orth=lowercase, string=broad}
Token	64	69	23	{category=NNS, kind=word, length=5, orth=lowercase, string=terms}
Token	70	72	25	{category=IN, kind=word, length=2, orth=lowercase, string=on}
Token	73	79	27	{category=NNP, kind=word, length=6, orth=upperInitial, string=Monday}
Token	80	82	29	{category=IN, kind=word, length=2, orth=lowercase, string=in}

974 Annotations (1selected)

Figure 3.7: GATE screenshot: Properties of our source document



gate.SourceURL | created fr

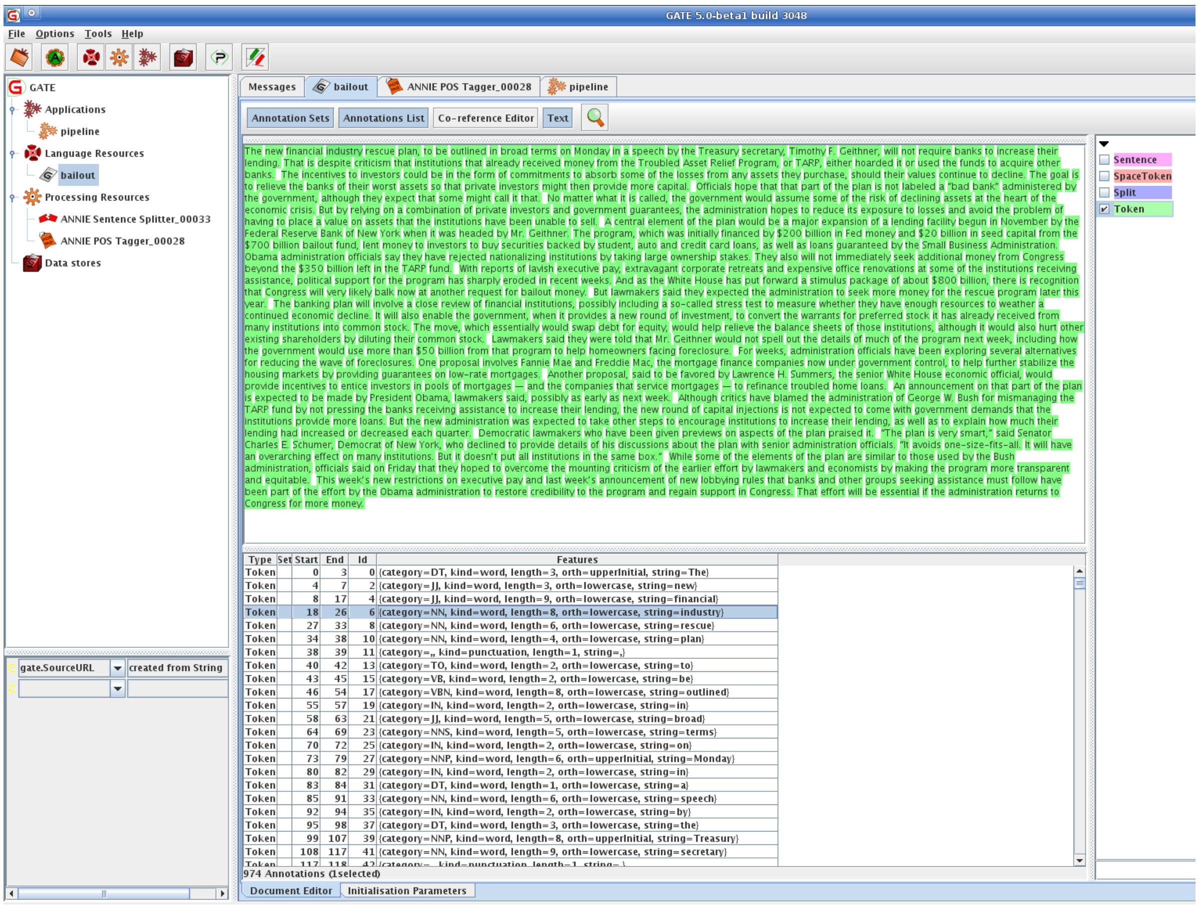
The new financial industry rescue plan, to be outlined in broad terms on Monday in a speech by the Treasury
secretary, Timothy F. Geithner, will not require banks to increase their lending. That is despite criticism that
institutions that already received money from the Troubled Asset Relief Program, or TARP, either hoarded it
or used the funds to acquire other banks. The incentives to investors could be in the form of commitments to
rom any assets they purchase, should their values continue to decline. The goal is
worst assets so that private investors might then provide more capital. Officials
an is not labeled a "bad bank" administered by the government, although they
it that. No matter what it is called, the government would assume some of the
e heart of the economic crisis. But by relying on a combination of private investors
, the administration hopes to reduce its exposure to losses and avoid the
a value on assets that the institutions have been unable to sell. A central element
r expansion of a lending facility begun in November by the Federal Reserve Bank
aded by Mr. Geithner. The program, which was initially financed by $200 billion in
n seed capital from the $700 billion bailout fund, lent money to investors to buy
t, auto and credit card loans, as well as loans guaranteed by the Small Business
inistration officials say they have rejected nationalizing institutions by taking large
will not immediately seek additional money from Congress beyond the $350
With reports of lavish executive pay, extravagant corporate retreats and
at some of the institutions receiving assistance, political support for the program

				Features
			0	{category=DT, kind=word, length=3, orth=upperInitial, string=The}
			2	{category=JJ, kind=word, length=3, orth=lowercase, string=new}
			4	{category=JJ, kind=word, length=9, orth=lowercase, string=financial}
			5	{category=NN, kind=word, length=8, orth=lowercase, string=industry}
			8	{category=NN, kind=word, length=6, orth=lowercase, string=rescue}
			0	{category=NN, kind=word, length=4, orth=lowercase, string=plan}
			1	{category=,, kind=punctuation, length=1, string=,}
Token	40	42	13	{category=TO, kind=word, length=2, orth=lowercase, string=to}
Token	43	45	15	{category=VB, kind=word, length=2, orth=lowercase, string=be}
Token	46	54	17	{category=VBN, kind=word, length=8, orth=lowercase, string=outlined}
Token	55	57	19	{category=IN, kind=word, length=2, orth=lowercase, string=in}
Token	58	63	21	{category=JJ, kind=word, length=5, orth=lowercase, string=broad}
Token	64	69	23	{category=NNS, kind=word, length=5, orth=lowercase, string=terms}
Token	70	72	25	{category=IN, kind=word, length=2, orth=lowercase, string=on}
Token	73	79	27	{category=NNP, kind=word, length=6, orth=upperInitial, string=Monday}
Token	80	82	29	{category=IN, kind=word, length=2, orth=lowercase, string=in}

974 Annotations (1selected)

Mark Hepple's Brill-style POS tagger.

Figure 3.8: GATE screenshot: Adding new processing resources



The new financial industry rescue plan, to be outlined in broad terms on Monday in a speech by the Treasury secretary, Timothy F. Geithner, will not require banks to increase their lending. That is despite criticism that institutions that already received money from the Troubled Asset Relief Program, or TARP, either hoarded it or used the funds to acquire other banks. The incentives to investors could be in the form of commitments to absorb some of the losses from any assets they purchase, should their values continue to decline. The goal is to relieve the banks of their worst assets so that private investors might then provide more capital. Officials hope that that part of the plan is not labeled a "bad bank" administered by the government, although they expect that some might call it that. No matter what it is called, the government would assume some of the risk of declining assets at the heart of the economic crisis. But by relying on a combination of private investors and government guarantees, the administration hopes to reduce its exposure to losses and avoid the problem of having to place a value on assets that the institutions have been unable to sell. A central element of the plan would be a major expansion of a lending facility begun in November by the Federal Reserve Bank of New York when it was headed by Mr. Geithner. The program, which was initially financed by $200 billion in Fed money and $20 billion in seed capital from the $700 billion bailout fund, lent money to investors to buy securities backed by student, auto and credit card loans, as well as loans guaranteed by the Small Business Administration. Obama administration officials say they have rejected nationalizing institutions by taking large ownership stakes. They also will not immediately seek additional money from Congress beyond the $350 billion left in the TARP fund. With reports of lavish executive pay, extravagant corporate retreats and expensive office renovations at some of the institutions receiving assistance, political support for the program has sharply eroded in recent weeks. And as the White House has put forward a stimulus package of about $800 billion, there is recognition that Congress will very likely balk now at another request for bailout money. But lawmakers said they expected the administration to seek more money for the rescue program later this year. The banking plan will involve a close review of financial institutions, possibly including a so-called stress test to measure whether they have enough resources to weather a continued economic decline. It will also enable the government, when it provides a new round of investment, to convert the warrants for preferred stock it has already received from many institutions into common stock. The move, which essentially would swap debt for equity, would help relieve the balance sheets of those institutions, although it would also hurt other existing shareholders by diluting their common stock. Lawmakers said they were told that Mr. Geithner would not spell out the details of much of the program next week, including how the government would use more than $50 billion from that program to help homeowners facing foreclosure. For weeks, administration officials have been exploring several alternatives for reducing the wave of foreclosures. One proposal involves Fannie Mae and Freddie Mac, the mortgage finance companies now under government control, to help further stabilize the housing markets by providing guarantees on low-rate mortgages. Another proposal, said to be favored by Lawrence H. Summers, the senior White House economic official, would provide incentives to entice investors in pools of mortgages — and the companies that service mortgages — to refinance troubled home loans. An announcement on that part of the plan is expected to be made by President Obama, lawmakers said, possibly as early as next week. Although critics have blamed the administration of George W. Bush for mismanaging the TARP fund by not pressing the banks receiving assistance to increase their lending, the new round of capital injections is not expected to come with government demands that the institutions provide more loans. But the new administration was expected to take other steps to encourage institutions to increase their lending, as well as to explain how much their lending had increased or decreased each quarter. Democratic lawmakers who have been given previews on aspects of the plan praised it. "The plan is very smart," said Senator Charles E. Schumer, Democrat of New York, who declined to provide details of his discussions about the plan with senior administration officials. "It avoids one-size-fits-all. It will have an overarching effect on many institutions. But it doesn't put all institutions in the same box." While some of the elements of the plan are similar to those used by the Bush administration, officials said on Friday that they hoped to overcome the mounting criticism of the earlier effort by lawmakers and economists by making the program more transparent and equitable. This week's new restrictions on executive pay and last week's announcement of new lobbying rules that banks and other groups seeking assistance must follow have been part of the effort by the Obama administration to restore credibility to the program and regain support in Congress. That effort will be essential if the administration returns to Congress for more money.

Type	Set	Start	End	Id	Features
Token		0	3	0	{category=DT, kind=word, length=3, orth=upperInitial, string=The}
Token		4	7	2	{category=JJ, kind=word, length=3, orth=lowercase, string=new}
Token		8	17	4	{category=JJ, kind=word, length=9, orth=lowercase, string=financial}
Token		18	26	6	{category=NN, kind=word, length=8, orth=lowercase, string=industry}
Token		27	33	8	{category=NN, kind=word, length=6, orth=lowercase, string=rescue}
Token		34	38	10	{category=NN, kind=word, length=4, orth=lowercase, string=plan}
Token		38	39	11	{category=,, kind=punctuation, length=1, string=,}
Token		40	42	13	{category=TO, kind=word, length=2, orth=lowercase, string=to}
Token		43	45	15	{category=VB, kind=word, length=2, orth=lowercase, string=be}
Token		46	54	17	{category=VBN, kind=word, length=8, orth=lowercase, string=outlined}
Token		55	57	19	{category=IN, kind=word, length=2, orth=lowercase, string=in}
Token		58	63	21	{category=JJ, kind=word, length=5, orth=lowercase, string=broad}
Token		64	69	23	{category=NNS, kind=word, length=5, orth=lowercase, string=terms}
Token		70	72	25	{category=IN, kind=word, length=2, orth=lowercase, string=on}
Token		73	79	27	{category=NNP, kind=word, length=6, orth=upperInitial, string=Monday}
Token		80	82	29	{category=IN, kind=word, length=2, orth=lowercase, string=in}
Token		83	84	31	{category=DT, kind=word, length=1, orth=lowercase, string=a}
Token		85	91	33	{category=NN, kind=word, length=6, orth=lowercase, string=speech}
Token		92	94	35	{category=IN, kind=word, length=2, orth=lowercase, string=by}
Token		95	98	37	{category=DT, kind=word, length=3, orth=lowercase, string=the}
Token		99	107	39	{category=NNP, kind=word, length=8, orth=upperInitial, string=Treasury}
Token		108	117	41	{category=NN, kind=word, length=9, orth=lowercase, string=secretary}
Token		117	118	42	{category=, kind=punctuation, length=1, string=,}

974 Annotations (1selected)

Figure 3.9: GATE screenshot: Examining the result of the operation

3.4 Open NLP

OpenNLP is a set of Java based NLP tools that *"are not inheriently useful by themselves, but can be integrated with other software to assist in the processing of text. ... The idea is to have a basic infrastructure in place for researchers and developers to collaborate on open source NLP projects. It is hoped that one day OpenNLP will be one of the major sites for cutting edge open source NLP software where people can find the software they need and get help on using and developing it"* [13]. Parts of it are licenced under the Apache Licence v2 whereas others are available under the LGPL. The project is additionally hosted at Sourceforge.

"OpenNLP also hosts a variety of java-based NLP tools which perform sentence detection, tokenization, POS tagging, chunking and parsing, named entity detection, and coreference[12] using the OpenNLP Maxent machine learning package" [13].

One project under the umbrella of OpenNLP is MAXENT - the maximum entropy framework. *"Maximum entropy modeling is a framework for integrating information from many heterogeneous information sources for classification "* [14] where classification problems are described by a number of features which corresponds to a constraint on the model. *"Choosing the maximum entropy model is motivated by the desire to preserve as much uncertainty as possible"* [14]. Maxent has perfomed well on difficult classification tasks like tasks like POS tagging, sentence detection, prepositional phrase attachment, and named entity recognition. For programmers new to Maxent, a how to is available on the Maxent website [14]. It briefly describes how to create and train a model for the purpose of finding names in a text.

The website[13] provides a download link, a Java API documentation and a short readme section which describes how to install and run the tools.

After downloading the opennlp package from Sourceforge.net, I consulted the online documentation for further information. I followed the suggestion to build the package with the supplied shell script (Java & Ant required) by calling *./build.sh* in the opennlp directory. For an english POS tagging test I continued with downloading a model[14] and extracted the content (*tag.bin*) into the opennlp directory. Before starting the test, I created a text file (*text.txt*) which content should be tagged. Finally I called

```
java -cp <jar files in the lib dir + opennlp.jar> opennlp.tools.lang.english.
    PosTagger tag.bin < test.txt
```

The annotated output of this operation looks like:

[12]coreference occurs when multiple expressions in a sentence or document have the same referent
[13]http://opennlp.sourceforge.net/
[14]POS model: http://opennlp.sourceforge.net/models/english/postag/tag.bin.gz

```
Google/NNP is/VBZ one/CD of/IN a/DT number/NN of/IN companies/NNS devising/VBG
    ways/NNS to/TO control/VB the/DT demand/NN for/IN electric/JJ power/NN as/
    IN an/DT alternative/NN to/TO building/NN more/JJR power/NN plants./, The/
    DT company/NN has/VBZ developed/VBN a/DT free/JJ Web/NNP service/NN called/
    VBD PowerMeter/NNP that/IN consumers/NNS can/MD use/VB to/TO track/VB
    energy/NN use/NN in/IN their/PRP\$ house/NN or/CC business/NN as/IN it/PRP
    is/VBZ consumed./VBN
Google/NNP is/VBZ counting/VBG on/IN others/NNS to/TO build/VB devices/NNS to/
    TO feed/VB data/NNS into/IN PowerMeter/NNP technology./, While/IN it/PRP
    hopes/VBZ to/TO begin/VB introducing/VBG the/DT service/NN in/IN the/DT
    next/JJ few/JJ months,/NNS it/PRP has/VBZ not/RB yet/RB lined/VBN up/IN
    hardware/NN manufacturers./.
```

The corresponding source code can be found in:

```
<opennlp dir>src/java/opennlp/tools/lang/english/PosTagger.java
```

and covers the relevant code for generating the tags.

```java
POSTaggerME tagger;   // part-of-speech tagger that uses maximum entropy
tagger = new PosTagger(model,(Dictionary)null);

BufferedReader in = new BufferedReader(new InputStreamReader(System.in)); //
    content of test.txt is System.in
for (String line = in.readLine(); line != null; line = in.readLine())
{
    System.out.println(tagger.tag(line));
}
```

3.5 Waikato Environment for Knowledge Analysis (WEKA)

The goal of the open source Java project WEKA is to *"build a state-of-the-art facility for developing machine learning techniques and to apply them to real-world data mining problems"* [15]. Target groups for the development are ML (machine learning) researchers and industrial scientists as well as students. Developer of the WEKA projects is the Univerity of Waikato in New Zealand.
WEKA commits itself to the following objectives [15]:

- make ML techniques generally available

Figure 3.10: The WEKA logo[15]

- apply them to practical problems that matter to New Zealand industry
- develop new machine learning algorithms and give them to the world
- contribute to a theoretical framework for the field

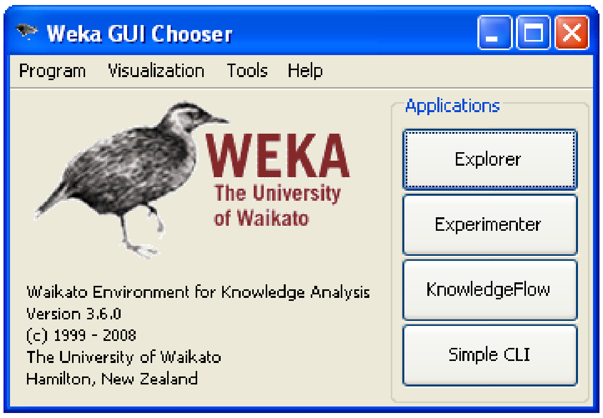
The software presents a *"collection of algorithms for solving real-world data mining problems"* [15] which are available under the GPL and includes tools for classification, regression, clustering as well as association rules. Facilities for data pre-processing and visualization are available too. All these tools can be used by implementing the provided Java API in your own projects, by passing relevant parameters to the command line executeable or just by starting the included GUI which is called WEKA Knowledge Explorer. *"The WEKA Knowledge Explorer is an easy to use GUI that harnesses the power of the weka software. Each of the major weka packages filters, classifiers, clusterers, associations and attribute selection is represented in the Explorer along with a visualization tool which allows datasets and the predictions of classifiers and clusterers to be visualized in two dimensions"* [15].

WEKA is available in version 3.6 (released 19.12.2008) for download on the homepage[15]. Furthermore, the WEKA project provides a API documentation in javadoc format, a wiki, some presentations and tutorials, and the book "Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)" written by Ian H. Witten and Eibe Frank, two professors on the University of Waikato. A community documentation is managed by Pentaho[16], a company providing commercial open source business intelligence solutions based on WEKA. An extensive manual is contained in the installation package of WEKA. This toolkit is frequently used in other NLP toolkits (eg. GATE).

After downloading the WEKA installer (I have downloaded the package for windows), you only have to start the application and follow the instructions to install it on your computer. The web site[17] gave the hint to add *-Xmx1024M* in the batch file starting

[15]http://www.cs.waikato.ac.nz/ ml/index.html
[16]http://www.pentaho.com/index.php
[17]WEKA installation: http://weka.wiki.sourceforge.net/Primer

WEKA. The main application consists of four buttons which allow you to open four different dialogs.



Waikato Environment for Knowledge Analysis
Version 3.6.0
(c) 1999 - 2008
The University of Waikato
Hamilton, New Zealand

Figure 3.11: WEKA screenshot: The small main application - entry point for all experiments

For our first experiment I choosed *Explorer* and loaded an ARFF sample file into WEKA which is easy to accomplish on the first tabbed pane of the application (*open file...*). I did not choose any filter but selected a J48 decision tree on the classification tab.

I selected default settings and started the experiment with the *Start* button. After some seconds WEKA presented the results in the text area labled *Classifier output*.

To get a grafical representation of the outcome, switch to the *Visualize* tab and select a plot. There you can assign the properties to the axis and change some plotter related attributes.

Another way to create an experiment is to use the *KnowledgeFlow Environment* which is further dialog of the main application. With KnowlegdeFlow you can design your problem in a graphical way. After designing your problem, you only have to load the data in the ArffLoader (context menu).

3.6 JwebPro

JWebPro is a Java based *"web processing toolkit that can interact with Google search via Google Web APIs and then process the returned web documents in a couple of ways."* [16] The output - a corpus - can be used for an independent natural language processing. It is published under GPL and hosted at Sourceforge.

JWebPro is based upon JTextPro and includes the following features [16]

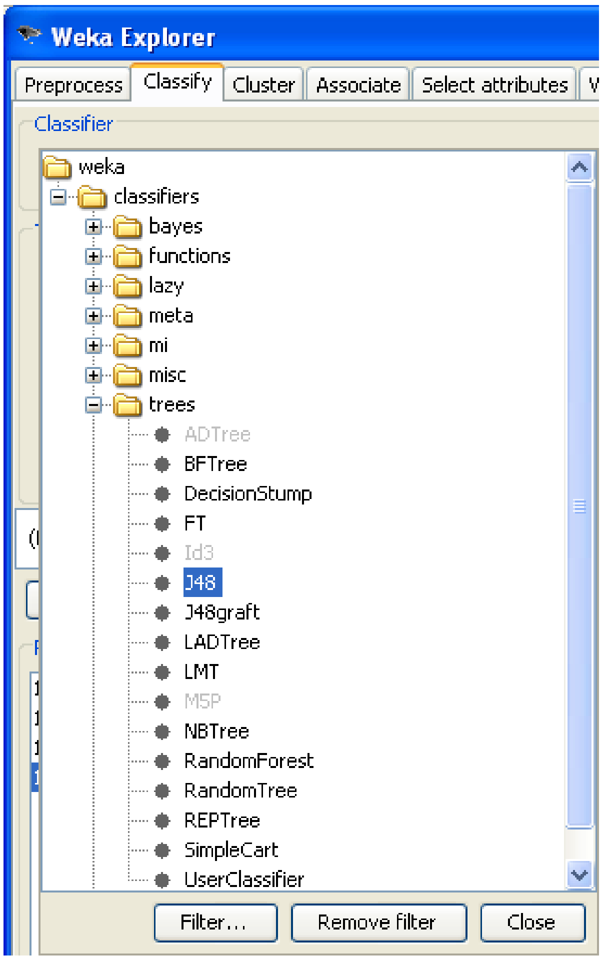
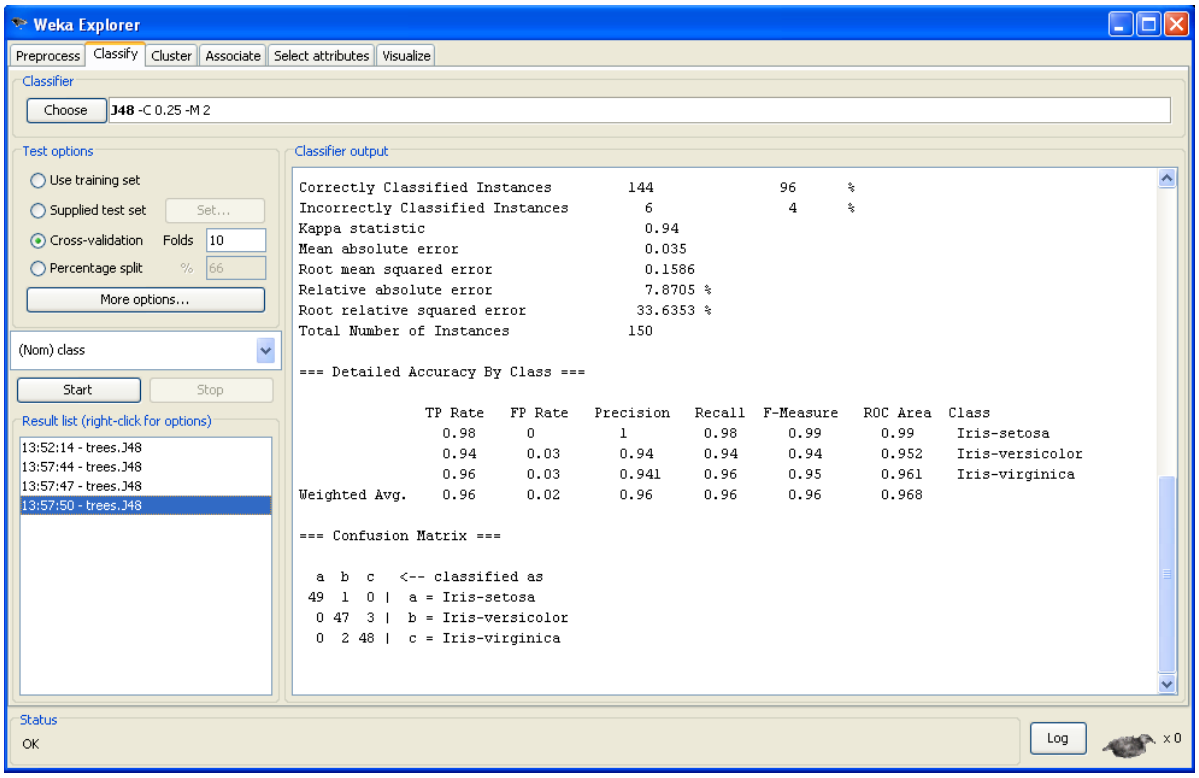


Figure 3.12: WEKA screenshot: Select one classifier from the combobox



Classifier: J48 -C 0.25 -M 2

Test options:
- ( ) Use training set
- ( ) Supplied test set
- [✓ Cross-validation] Folds 10
- ( ) Percentage split % 66

(Nom) class

Result list (right-click for options)
- 13:52:14 - trees.J48
- 13:57:44 - trees.J48
- 13:57:47 - trees.J48
- 13:57:50 - trees.J48

Classifier output

```
Correctly Classified Instances         144               96      %
Incorrectly Classified Instances         6                4      %
Kappa statistic                          0.94
Mean absolute error                      0.035
Root mean squared error                  0.1586
Relative absolute error                  7.8705 %
Root relative squared error             33.6353 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.98      0         1         0.98      0.99      0.99     Iris-setosa
                 0.94      0.03      0.94      0.94      0.94      0.952    Iris-versicolor
                 0.96      0.03      0.941     0.96      0.95      0.961    Iris-virginica
Weighted Avg.    0.96      0.02      0.96      0.96      0.96      0.968

=== Confusion Matrix ===

  a  b  c   <-- classified as
 49  1  0 |  a = Iris-setosa
  0 47  3 |  b = Iris-versicolor
  0  2 48 |  c = Iris-virginica
```

Status
OK

Figure 3.13: WEKA screenshot: Textual representation of the output

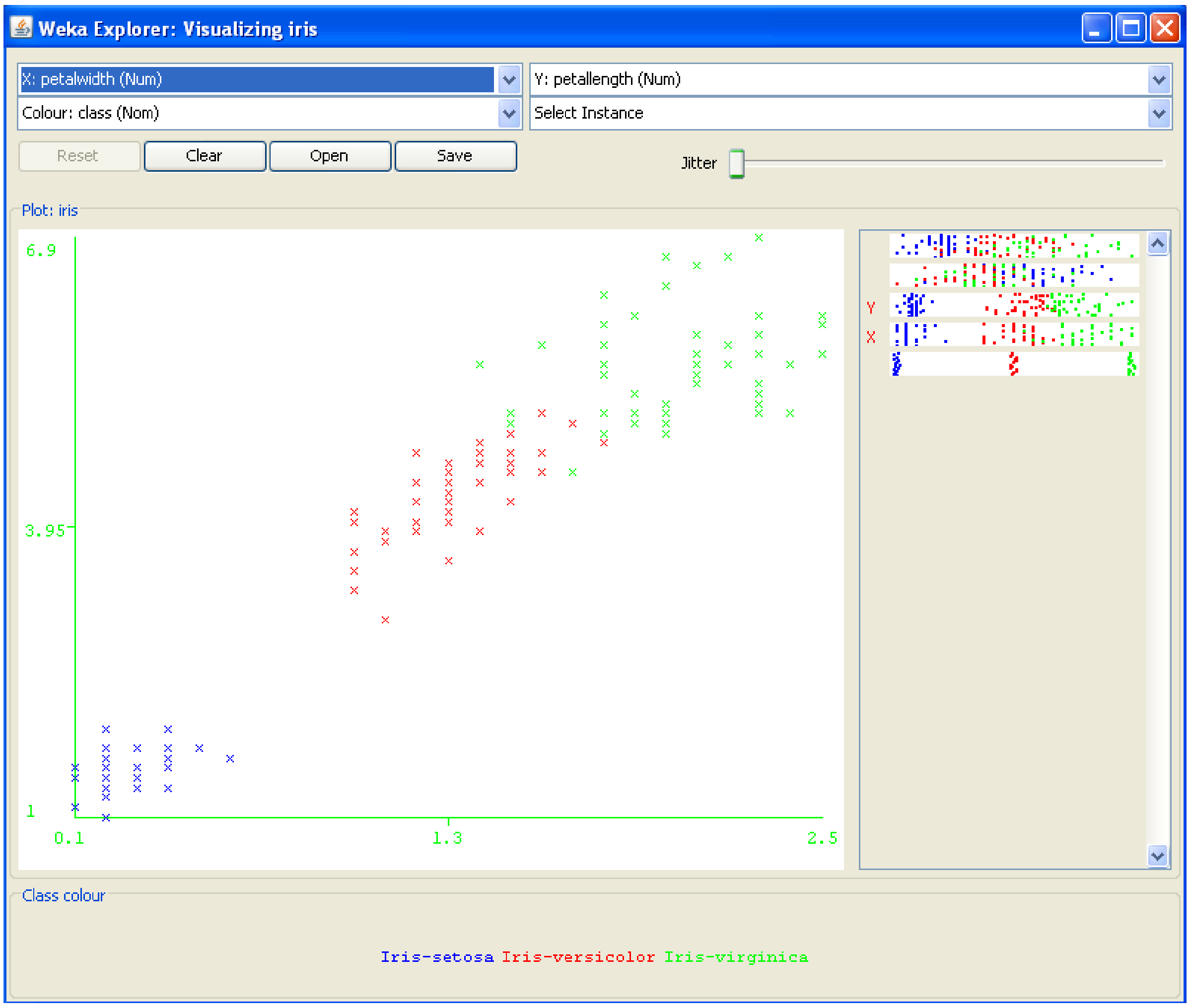


Figure 3.14: WEKA screenshot: Grafical representation of the output

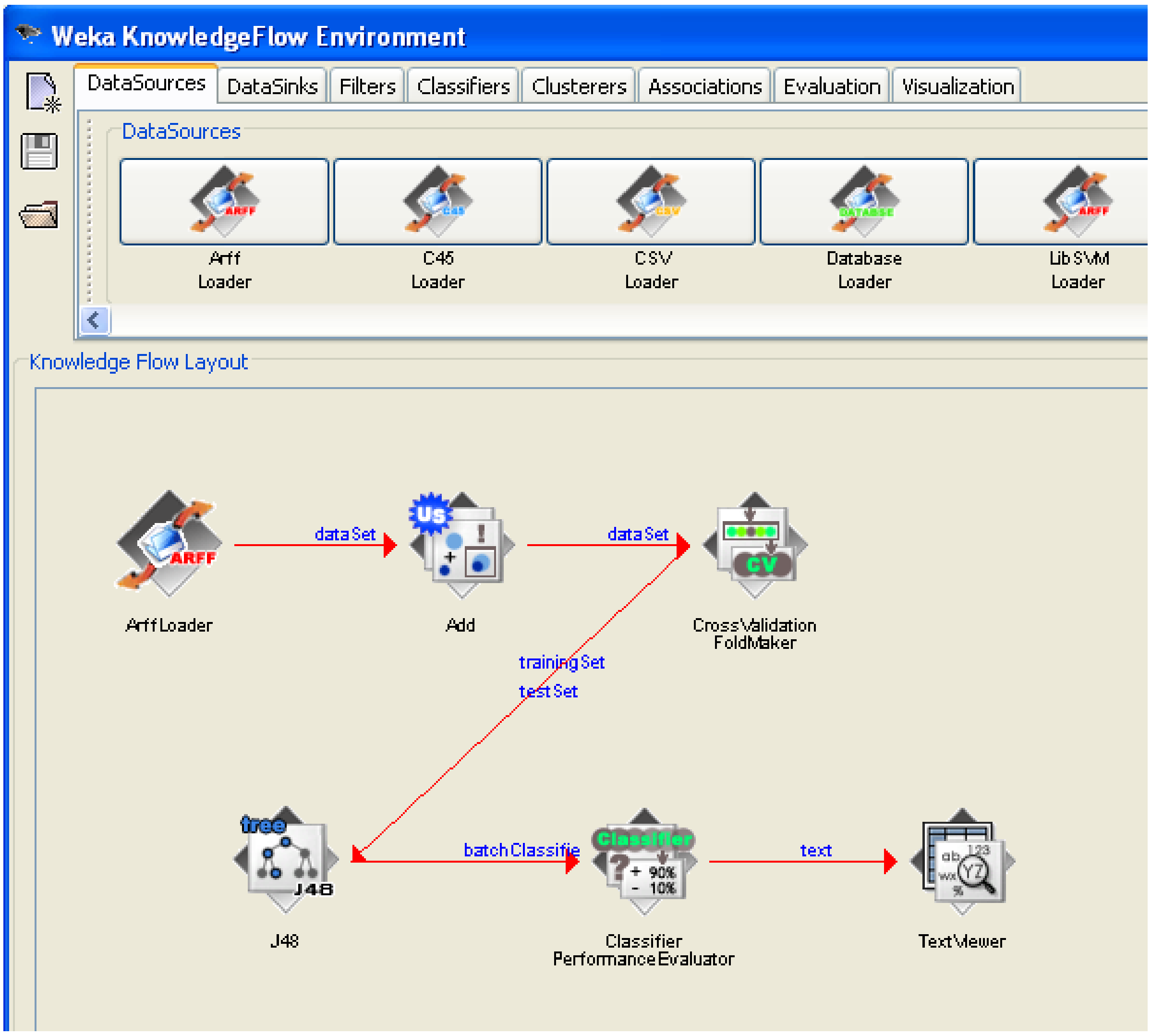


Figure 3.15: WEKA screenshot: Grafical design of an problem

- interacts with Google search via Google Web APIs
- web crawler downloading relevant web documents according to the index list returned by Google search
- parsing HTML documents (Htmlparser library)
- sentence boundary detection (using maximum entropy classifier trained on WSJ corpus)
- word tokenization
- POS tagging (using CRFTagger[18])
- phrase chunking (using CRFChunker)

The CRFTagger in JWebPro is different from POS tagging for pure text/natural language (eg. in JTextPro) because Web documents are more noisy and less grammatical. The project team adapted their tagger to this requirements over time. [16]

This toolkit tries to provide a framework to build applications in the domain of NLP and information retrieval which processes data from the world wide web. If online functionality is not required JTextPro would be the right choice.

JWikiDoc, an extension of JWebPro is included in this project. It is *"a tool for crawling and downloading Wikipedia documents"* [16]. This tool is useful *"for building Web data collections/corpora"* [16] from Wiki pages and its functionality covers removing HTML tags, navigation links and noisy text. The number of retrieved Wiki pages can be adjusted by setting maximum number of retrieved documents or the maximum hyperlink depth.

The ground lying Java-based CRFTagger for English itself is built upon FlexCRFs which is designed to achieve high tagging speed. *"The model was trained on sections 01..24 of WSJ corpus and using section 00 as the development test set (accuracy of 97.00%)."* [17]

FlexCRF itself is written in C/C++ using STL[19] library and is designed to deal with large datasets and millions of features. Furthermore it supports first-order and second-order Markov CRFs. The package includes a parallel version of FlexCRF which allows users to train conditional random fields on massively parallel processing systems supporting the Message Passing Interface[20].

[18]Conditional Random Field Tagger

[19]The Standard Template Library is a generic C++ library that provides many of the basic algorithms and data structures of computer science

[20]MPI is an high performance interface for massively parallel machines where many parallel working processes communicates with each other to solve a problem faster than each process on its own

To work with JWebPro, a Google API key is required. While Google do not issue new keys, it is not possible to use the Google SOAP API. At application startup, the Google key has to be suplied with an parameter file (option.txt) otherwise it will not work.

The website[21] of JWebPro provides a brief documentation in form of a readme file which is included in the JWebPro download package too. Some links to related projects like JWikiDoc and JTextPro are also available on the website.

It was not possible to launch JWebPro because of the missing Google API key. Instead I used JTextPro which is responsible for text processing inside of JWebPro. The download is available on the same web site which covers the sources and the compiled library *jtextpro.jar*. The included readme file explains the structure of the command line parameters. The application can be used by typing:

```
java -Xmx512m -classpath ./lib/jtextpro.jar jtextpro.JTextProcessor ./models ./
    samples/input.txt}\\ within the JTextPro folder.

Loading sentence segmentation model ...
Reading options ...
Reading options completed!
Reading the context predicate maps ...
Reading context predicate maps (19803 entries) completed!
Reading the context predicate maps ...
Reading label maps (2 entries) completed!
Reading dictionary ...
Reading dictionary (19803 entries) completed!
Reading features ...
Reading 22200 features completed!
Loading sentence segmentation model completed!

Loading POS tagging model ...
Reading options ...
Reading options completed!
Reading the context predicate maps ...
Reading context predicate maps (211800 entries) completed!
Reading label maps ...
Reading label maps (45 entries) completed!
Reading dictionary ...
Reading dictionary (211800 entries) completed!
Reading features ...
Reading 428699 features completed!
```

[21] http://jwebpro.sourceforge.net/

```
Loading POS tagging model completed!

Loading phrase chunking model ...
Reading options ...
Reading options completed!
Reading the context predicate maps ...
Reading context predicate maps (240114 entries) completed!
Reading label maps ...
Reading label maps (23 entries) completed!
Reading dictionary ...
Reading dictionary (240114 entries) completed!
Reading features ...
Reading 548751 features completed!
Loading phrase chunking model completed!
```

The output file is called like the input file with the additional extension *.out.* and looks like
Why/WRB/B-ADVP Is/VBZ/O Microsoft/NNP/B-NP Afraid/NNP/I-NP of/IN/B-PP Google/NNP/B-NP
In comparison to the input
Why Is Microsoft Afraid of Google
it contains POS annotations. Information about these annotations can be found on the web site of the University of Leeds[22].

3.7 Automatic Content Extraction (ACE)

This is not a NLP toolkit as described in the other sections. *"The objective of the NIST (National Institute of Standards and Technology) ACE series of evaluations is to develop human language understanding technology that provides automatic detection and recognition of key information about real-world entities, relations, and events that are mentioned in source data"* [18]. Input sources include text, audio and image data whereas text can be written in Arabic, Chinese and English. Until now, detection of real-world objects was limited to documents while coreferencing these objects across documents was impossible. *"This focus is changing in 2008 with the ACE evaluation scaling up to cross-document and cross-language global integration and reconciliation of information"* [18].

The Automatic Content Extraction (ACE) program, a new effort to stimulate and benchmark research in information extraction, presents four challenges [19]

[22]http://www.scs.leeds.ac.uk/ccalas/tagsets/brown.html

- Recognition of entities, not just names - entity detection and tracking (EDT)
- Recognition of relations - relation detection and characterization (RDC)
- Event extraction - event detection and characterization (EDC)
- Extraction is measured not merely on text, but also on speech and on OCR input. The lack of case and punctuation, including the lack of sentence boundary markers, poses a challenge to full parsing of speech.

"The LDC (Linguistic Data Consortium) is an open consortium of universities, companies and government research laboratories. It creates, collects and distributes speech and text databases, lexicons, and other resources for research and development purposes. The University of Pennsylvania is the LDC's host institution." [20]

The NIST website[23] do not provide software components to start with NLP operations, but it provides evaluation data which can be used to test your NLP algorithms. The results of your test can be submitted to NIST which compares the reference output to your output and calculates the difference.

3.8 TermExtractor

It is an online webservice that is developed by the Linguistic Computing Laboratory (LCL) which is part of the Computer Science Department of the University of Roma "La Sapienza". *"The group works in the areas of the semantic web (the web of next generation), computational linguistics, e-learning, and information retrieval"* [21].

"TermExtractor is a software package for the extraction of relevant terms consensually referred in a specific domain" [21]. Application input can be a corpus of domain documents that will be parsed and plausible terms are extracted. *"Two entropy-based measures, called Domain Relevance and Domain Consensus, are then used to select only those terms which are relevant to the domain of interest or consensually referred throughout the documents. This is achieve with the aid of a set of contrastive corpora from different domains"* [21].

Furthermore, TermExtractor is able to recognize terms in many types of document (txt, pdf, ps, dvi, tex, doc, rtf, ppt, xls, xml, html/htm, chm, wpd) and text layouts: title, bold, italic, underlined, colored, capitalized, smallcaps. The text layout is used to assign to terms with selected layouts a greater importance.

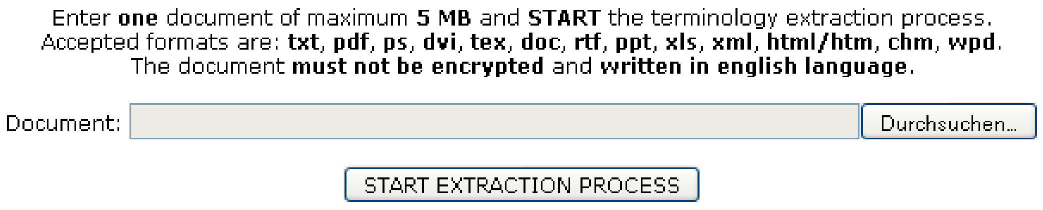
[23]http://www.nist.gov/speech/tests/ace/

The system is intended to collect the ”common terminology”. If a term is only referred in one document, it is not considered. The larger the the corpus is, the better TermExtractor can work. It uses statistical measures, and, in absence of a statistically significant evidence, a term cannot be extracted! Three main measures are of significance:

- Domain Relevance: the term must be frequent in the corpus you submit, and not frequent or never found in other reference corpora (this allows you to skip terminology which is not specific of this domain, eg the term world wide web is found also in other domains). In one of the options, you can change the ”default” contrastive domains used for this analysis.
- Domain Consensus: within the domain to be analysed, the frequency of a term must be evenly distributed across documents, simulating the consensus that a term must gain before being accepted as a true domain term.
- Lexical cohesion: this is a statistical measure that computes the likeliness that a multi word string is really a terminological string and not a sequence of unrelated terms.

TermExtractor is available on the website[24] and requires a free registration to use it. In case you do not want to register, you can only submit 1 document with a maximum size of 5 megabytes. Once registered you can start to upload archives of documents to extract terminology from. Instructions for using TermExtractor are available online and will provide additional information on the statistical measures[25] used by TermExtractor. While TermExtractor is an online web service no further (installation, how-to) documentation is provided.

Using TermExtractor is very simple because it is an web service which provides an text input field on the web site.

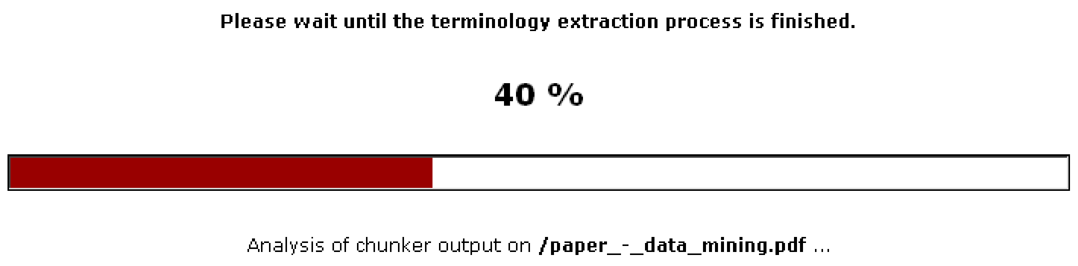

Enter **one** document of maximum **5 MB** and **START** the terminology extraction process.
Accepted formats are: **txt**, **pdf**, **ps**, **dvi**, **tex**, **doc**, **rtf**, **ppt**, **xls**, **xml**, **html/htm**, **chm**, **wpd**.
The document **must not be encrypted** and **written in english language**.

Document:

Figure 3.16: TermExtractor screenshot: Text input field on the web site

While not registered, I could only submit one document to the web service. After submitting, the web site shows a progress bar indicating the status of the operation.

[24]http://lcl2.di.uniroma1.it/termextractor
[25]http://lcl2.di.uniroma1.it/termextractor/help/termextractor_measures.pdf


Please wait until the terminology extraction process is finished.

40 %

Analysis of chunker output on **/paper_-_data_mining.pdf** ...

Figure 3.17: TermExtractor screenshot: Progress bar showing the status of term extraction

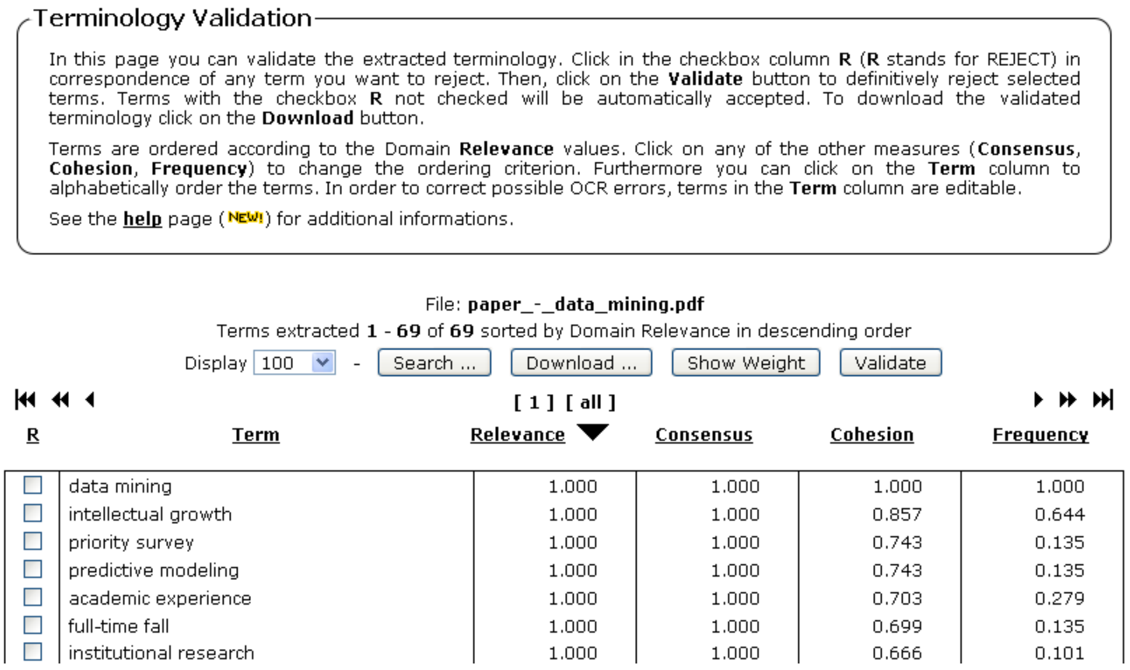
After finishing the extraction process, the results are presented on a new page which contains a short explanation and information for each term:

- Relevance
- Consensus
- Cohesion
- Frequency

3.9 TreeTagger

The TreeTagger is a closed source tool for annotating text with POS and lemma information which has been developed at the Institute for Computational Linguistics of the University of Stuttgart by Helmut Schmid. It supports the languages German, English, French, Italian, Dutch, Spanish, Bulgarian, Russian, Greek, Portuguese, Chinese and old French texts and can be adapted to other languages if manually tagged corpora are available. It can also be used as a chunker (tool which breaks input file into parts) for English, German and French only.

"It differs from other probabilistic taggers in the way the transition probabilities are estimated" [22]. *"The TreeTagger is a Markov Model tagger which makes use of a decision tree to get more reliable estimates from contextual parameters."* [23] *"The resulting tagger achieves higher accuracy than a standard trigramm tagger. Due to an efficient implementation, the tagger is able to tag up to 10.000 tokens per second on a SPARC10 workstation. Thus, the TreeTagger is a fast and high-quality tool for the annotation of corpora with POS information"* [22].


Terminology Validation

In this page you can validate the extracted terminology. Click in the checkbox column **R** (**R** stands for REJECT) in correspondence of any term you want to reject. Then, click on the **Validate** button to definitively reject selected terms. Terms with the checkbox **R** not checked will be automatically accepted. To download the validated terminology click on the **Download** button.

Terms are ordered according to the Domain **Relevance** values. Click on any of the other measures (**Consensus**, **Cohesion**, **Frequency**) to change the ordering criterion. Furthermore you can click on the **Term** column to alphabetically order the terms. In order to correct possible OCR errors, terms in the **Term** column are editable.

See the **help** page (NEW!) for additional informations.

File: **paper_-_data_mining.pdf**

Terms extracted **1 - 69** of **69** sorted by Domain Relevance in descending order

R	Term	Relevance	Consensus	Cohesion	Frequency
☐	data mining	1.000	1.000	1.000	1.000
☐	intellectual growth	1.000	1.000	0.857	0.644
☐	priority survey	1.000	1.000	0.743	0.135
☐	predictive modeling	1.000	1.000	0.743	0.135
☐	academic experience	1.000	1.000	0.703	0.279
☐	full-time fall	1.000	1.000	0.699	0.135
☐	institutional research	1.000	1.000	0.666	0.101

Figure 3.18: TermExtractor screenshot: The result of the extraction process

The TreeTagger project startet in the year 1993 and was funded by the Ministry of Science and Research of the Land Baden-Württemberg. Documentation to this project is originated in *"a revised version of a paper which was presented at the International Conference on New Methods in Language Processing, 1994, Manchester, UK"* [22]. Executeables (no source code) are available for windows, linux, mac-osx and sparc systems and its licence grants you (the licensee) the right to use the TreeTagger software (the system) for evaluation, research and teaching purposes. Any other usage of the system (in particular for commercial purposes) is forbidden [24]. Furthermore a windows GUI is available for download separately. The last GUI update was accomplished in March 2008. The TreeTagger software was last updated on 18th of July 2008 (timestamp of the treetagger executeable) and comes with a brief description of the command line parameters in a readme file.

This closed source tool comes with perl scripts and binaries for different platforms that allows you a quick installation. After downloading the platform package, the tagging scripts, the install script and the language specific parameter file into the same directory (do not unzip), provide the installer script with the required file mode (chmod +x install-tagger.sh) and execute it. This will extract the downloaded files and establish a working environment for TreeTagger. Working with the software is quite easy. The directory *<TreeTagger directory>/cmd* contains a perl script for each NLP operation available in

TreeTagger. Starting the POS tagger looks like (test.txt contains text from a NY Times news article):

```
>./cmd/tree-tagger-english test.txt
  reading parameters ...
  tagging ...
Google NP  <unknown>
is  VBZ be
one CD  one
of  IN  of
a DT  a
number  NN  number
of  IN  of
companies NNS company
devising VBG devise
ways  NNS way
to  TO  to
control VB control
the DT the
demand  NN  demand
for IN for
electric  JJ  electric
power NN power
as  IN  as
an  DT  an
alternative NN alternative
to  TO  to
building VBG build
more  JJR more
power NN power
plants NNS plant
. SENT  .
The DT the
company NN company
has VBZ have
developed VBN develop
...
```

The output is split into three columns: First with the original word from the input file, followed by the POS annotation in the second column. The last column contains the lemma of the input word. Because TreeTagger is closed source, I can not provide any source snippets.

3.10 OpenCalais

OpenCalais encapsulates NLP technology within an online web service that is operated by Thomson Reuters after the acquisition of ClearForest. *"Calais is a rapidly growing toolkit of capabilities that allow you to readily incorporate state-of-the-art semantic functionality within your blog, content management system, website or application."* [25]

Figure 3.19: OpenCalais logo

Usage is free of charge, but some costly licence models with advanced features are available too. The service handles plain text as an input that should be semantically tagged with the help of NLP techniques. The result of the web service is either HTML text where all identified people, places, events & facts are highlighted or an RDF file (Resource Description Framework) which is a standard for describing resources on the web. To get a quick overview of the capabilities of Calais, their homepage (http://www.opencalais.com/) provides an online application[26] called *DocumentViewer* which processes text from an HTML text area. Furthermore a browser plugin for Internet Explorer and Firefox called *Gnosis* is available to tag every page visited and presents the results in a sidebar of your browser.

The Calais web service is the core API for many applications. A lot of effort is spent to bring these tools to the developers to enable them building great applications for other users of the community. The following illustration shows tools provided to the Calais community.

Some interesting tools provided by Calais for developers are [25]:

- *Calais web service*
- *Calais Marmoset* allows your web site to provide microformat metadata to other metadata crawlers
- The *Calais Pipes Service* enables Yahoo! Pipes users to enrich their RSS feeds with semantic metadata

[26]http://sws.clearforest.com/calaisViewer/

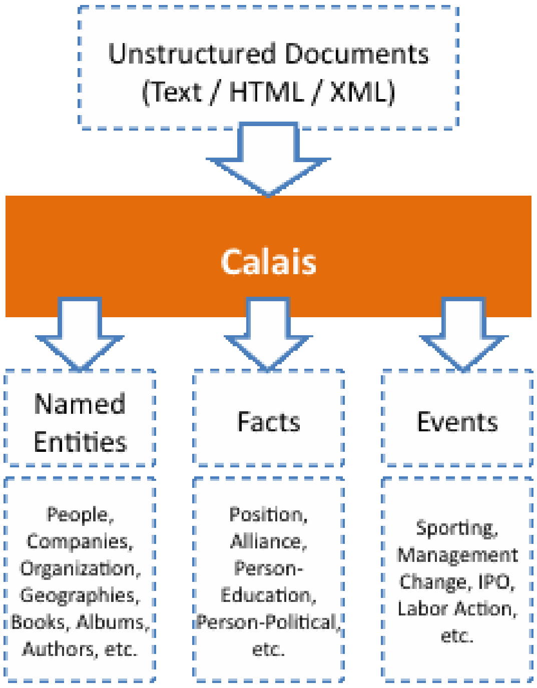

Unstructured Documents (Text / HTML / XML)
Calais
Named Entities
Facts
Events
People, Companies, Organization, Geographies, Books, Albums, Authors, etc.
Position, Alliance, Person-Education, Person-Political, etc.
Sporting, Management Change, IPO, Labor Action, etc.

Figure 3.20: OpenCalais workflow

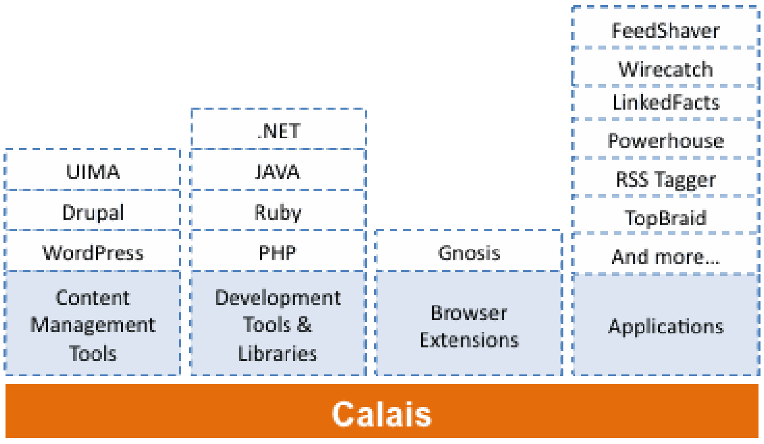


Figure 3.21: OpenCalais contents

- The *Tagaroo WordPress plugin* supercharges your WordPress blog with automated tag suggestions and Flickr image searching and incorporation
- *Gnosis* is a Firefox (3.0 and 2.0) and IE plugin that automatically analyzes content as you read it and provides you with a variety of tools to explore the people, companies, places and things youŕe reading about
- The *Submission Tool* performs semantic analysis of files supplied by the user via a series of submission screens
- The *Microsoft Office SharePoint Server 2007 (MOSS 2007) Open Calais Integration* automatically adds semantic metadata and intelligent search capabilities to content pages on MOSS 2007 content management web sites.
- *Microsoft Popfly* allows users to easily create mashups that include information from various sources. The ”Calais Popfly Block” shows how Microsoft Popfly users can enrich their Twitter-based mashups, and possibly others, with semantic metadata provided by OpenCalais

Some applications using OpenCalais have arised recently. A couple of them are official solutions from the Thompson Reuters company, others are community driven:

- Thinkpedia (community) - *”Thinkpedia is a new way to navigate and explore the contents of Wikipedia. It is what we call a ”Visual Wiki”. Thinkpedia uses SemanticProxy in order to extract semantics out of Wikipedia articles, which are then shown in an interactive visualization which is created with Thinkmap.”* [26]
- SemantalyzR (community) *”...is an easy to use tool for analyzing web pages for semantic data. On our page analysis you'll find keyword links to search pages for that keyword. These search pages are made up from results of various services on the web.”* [25] Some prominent supporters are Yahoo!, Flickr, Twitter and Wikipedia.
- Calais Tagaroo Wordpress plugin and Phase2Technologys Drupal plugin for automatically tagging your blog.
- LinkedFacts.com *”...helps by creating references to informations on things mentioned in your articles, such as background infos on people, neighborhood news on cities, company profiles, images, bookmarks, search results, and much more.”* [27]
- SemanticProxy *”...translates the content of any URL on the web to its semantic representation in RDF, HTML or Microformats.”* [9]

Calais did not provide an open source product, but introduced three ways to access their services. Dependent on what you need, 3 licence models are available:

- Open Calais web service allowes users *”to submit up to 40,000 transactions per day at a maximum rate of four transactions per second”*[25].

- Calais Professional provides identical functionality as Open Calais, but with a high-performance SLA[27] which defines *"a daily transaction limit of 100,000 transactions and an enhanced 20 transactions per second rate"* [25]. Although the transaction volume can be extended to a maximum of 2.000.000 per day.
- ClearForest[28] On-Premise Solutions installs Calais technology inside your company and allows the buyer to modify the metadata generation engine *"to create new entities and relations and to incorporate user lexicons"* [25].

The OpenCalais website[29] provides access to a gallery of tools, a forum and a blog containing tons of information about this web service. You need to register if you want to use the SOAP / REST API to access the webservice. Information about using these programming interfaces are on the website too.

While OpenCalais provides a tool called Document Viewer[30] where you can paste the text that should be annotated, I want to describe the soap interface. To work with the API you need to register[31] for free to obtain a API key. More information about the web service parameters is available on the tools section[32] of the web site. The required web service description is provided too[33]. So I startet to create a light soap client to utilize the web service. With the following steps this task was completed in a little while:

- Create new Java project in Eclipse (3.4)
- Create new Web Service Client (provide the url to the WSDL file to autogenerate the required source files)
- Create a Java class utilizing the request

The simple java class required to do the soap request looks like:

```java
public class SOAPQuery {

  public static void main(String[] args)
  {
    try
    {
      Calais service = new CalaisLocator();
      CalaisSoap call = service.getcalaisSoap();

```

[27]Service Level Agreement: formal definition between two parties about the level of service
[28]provides text-driven business intelligence solutions, aquired by Reuters in April 2007
[29]http://www.opencalais.com/
[30]http://sws.clearforest.com/calaisviewer/
[31]OpenCalais registration http://www.opencalais.com/apps/register
[32]OpenCalais documentation: http://www.opencalais.com/calaisAPI
[33]OpenCalais WSDL: http://api.opencalais.com/enlighten/?wsdl

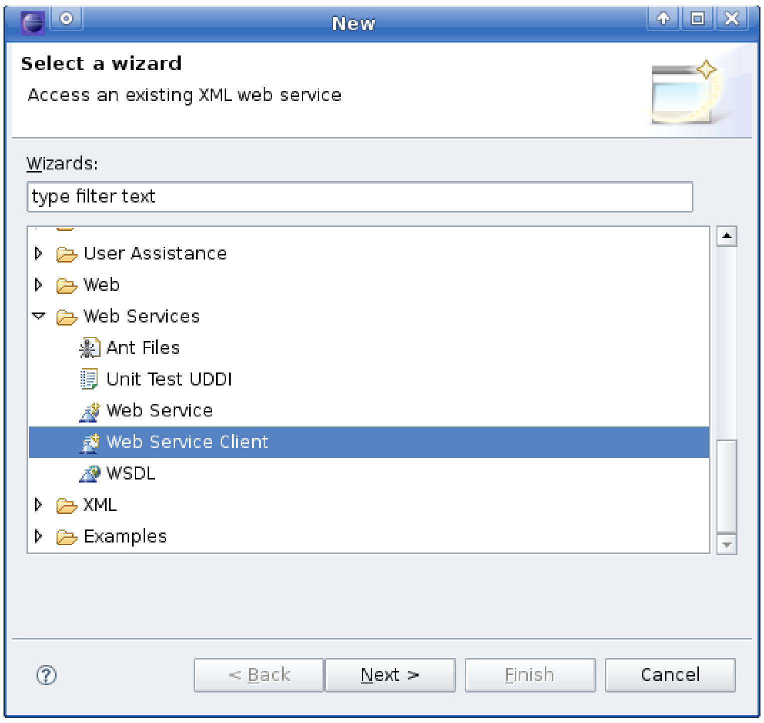


Figure 3.22: OpenCalais screenshot: Generating a web service client in Eclipse 3.4

```java
      // supply parameters
      java.lang.String licenseID = "7hw523rj236rw97mnjj7qw5u";

      java.lang.String content = "Ludwig van Beethoven , 16 December 1770[1] -
         26 March 1827) was a German composer and pianist. He was a crucial
         figure in the transitional period between the Classical and Romantic
         eras in Western classical music, and remains one of the most acclaimed
          and influential composers of all time." + "Born in Bonn, then in the
         Electorate of Cologne in western Germany, he moved to Vienna in his
         early twenties and settled there, studying with Joseph Haydn and
         quickly gaining a reputation as a virtuoso pianist. Beethoven's
         hearing gradually deteriorated beginning in his twenties, yet he
         continued to compose, and to conduct and perform, even after he was
         completely deaf.";

      java.lang.String paramsXML = "<c:params xmlns:c=\"http://s.opencalais.com
         /1/pred/\" xmlns:rdf=\"http://www.w3.org/1999/02/22-rdf-syntax-ns#\">"
          + "<c:processingDirectives c:contentType=\"text/txt\" c:
         enableMetadataType=\"GenericRelations\" c:outputFormat=\"Text/Simple
         \">" + "</c:processingDirectives>" + "<c:userDirectives c:
         allowDistribution=\"true\" c:allowSearch=\"true\" c:externalID=\"17
         cabs901\" c:submitter=\"ABC\">" + "</c:userDirectives>" + "<c:
         externalMetadata>" + "</c:externalMetadata>" + "</c:params>";

      // do the call and read the result
      java.lang.String result = call.enlighten(licenseID, content, paramsXML);

      // print output to std.out
      System.out.println("Result = " + result);
    }
    catch (Exception ex)
    {
      ex.printStackTrace();
    }

  }

}
```

As you can see, the soap service provides a function called *enlighten* which takes three parameters:

- licenceID - the key obtained from OpenCalais after registration

- content - a string which contains the text that should be annotated (the first two paragraphs of the Wikipedia entry about Beethoven)
- paramsXML - a XML parameter file[34] containing some processing information (e.g. the output format *Text/Simple*; other possibilities are *XML/RDF* and *Text/Microformats*)

The return of the web service looks as follows:

```
Result = <!-- Use of the Calais Web Service is governed by the Terms of Service
     located at http://www.opencalais.com. By using this service or the results
     of the service you agree to these terms of service. -->
 <!--City: Vienna, Cologne,
Country: Germany,
Person: Ludwig van Beethoven, Joseph Haydn,
--><OpenCalaisSimple>
  <Description>
    <allowDistribution>true</allowDistribution>
    <allowSearch>true</allowSearch>
    <calaisRequestID>3b10002c-6a30-4654-8bbe-17bdddba1f10</calaisRequestID>
    <externalID>17cabs901</externalID>
    <id>http://id.opencalais.com/2aLe3hwFIQMn-TDOg41ePQ</id>
    <about>http://d.opencalais.com/dochash-1/13099a0c-6ff1-3fb2-a12f-
        a936545aa726</about>
  </Description>
  <CalaisSimpleOutputFormat>
    <Person count="8" relevance="0.905">Ludwig van Beethoven</Person>
    <City count="1" relevance="0.197">Vienna</City>
    <City count="1" relevance="0.197">Cologne</City>
    <Country count="1" relevance="0.197" normalized="Germany">Germany</Country>
    <Person count="1" relevance="0.197">Joseph Haydn</Person>
    <Topics />
  </CalaisSimpleOutputFormat>
</OpenCalaisSimple>
```

The output first contains a summary of all recognised entities grouped by the kind of entity. Then some meta information about the request is displayed. At the end the details to the entities are presented in the simple text output format.

[34]OpenCalais XML input parameters: http://www.opencalais.com/APIcalls#inputparameters

3.11 Apache UIMA

The Apache Unstructured Information Management Applications (UIMA) are part of a framework which has been developed by IBM and is intended to analyze unstructured information like text, video and audio data *"in order to discover knowledge that is relevant to an end user"*[28]. The applications are written in Java or C++ and are affiliated with each other. To enable interaction between the components, these have to implement interfaces defined by the framework, which manages and controls data flow among the components. Furthermore UIMA can wrap components as network services, which allows the application to spread over a network of computers to achieve its goals.

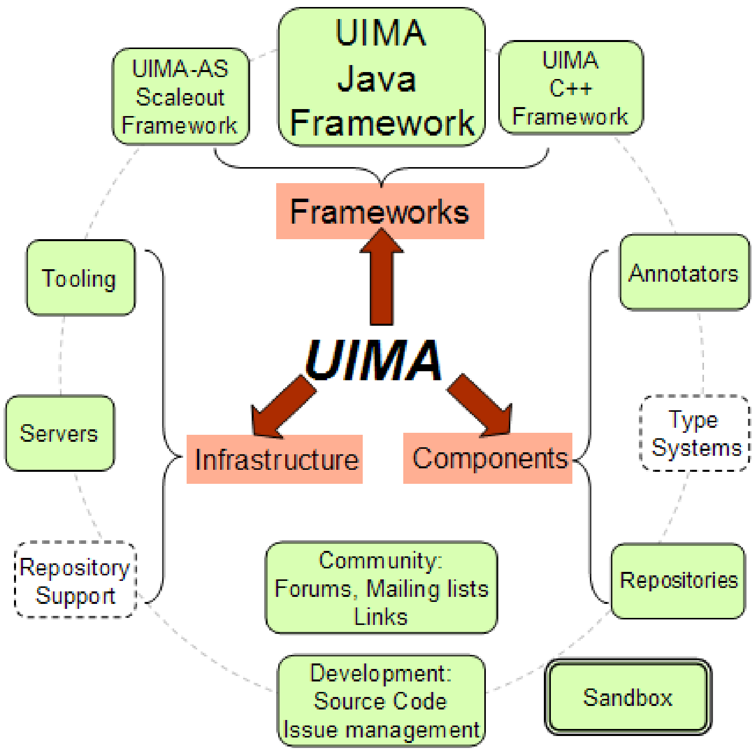


Figure 3.23: The UIMA framework [28]

While UIMA is an empty framework, it provides a basis to develop and combine different analytic components. The major field of application is text search. For sophisticated search applications, unstructured text has to be processed with common tagging techniques like tokenization, lemmatization and POS detection. Furthermore UIMA provides analysis components for entity and relation detection.

The UIMA framework is a open source project and available under the Apache Licence for download. Implementations are available written in Java and in C++ for Linux and Windows. For developers, Apache also supplies an plugin for simple annotator development in the Eclipse Java IDE. An annotator is the part of the program that analyses the input and returns an annotated document. Additionally to Java and C++ annotators can be written in Perl, Python and TCL. Some **annotators** are already available for download to do the real work of extracting information from unstructured data:

- Whitespace Tokenizer - uses simple withespace segmentation to annotate token and sentences.
- Snowball[35] - provides a stemming algorithm for several languages
- Regular Expression - detects entities based on regular expressions like email addresses, URLs, phone numbers, . . .
- Tagger - implementation of a Hidden Markov Model tagger
- Java Bean Scripting Framework (BSF[36]) - an interfaces which interprets annotators in scripting languages like Beanshell[37], Rhino Javascript[38], Jython[39] and JRuby[40].
- Dictionary - looks up a simple dictionary to create annotations
- OpenCalais - wraps the OpenCalais web service and returns its results
- Concept Mapper - is a individually configurable dictionary based annotator

Several user manuals and guides, online javadoc as well as getting-started documents are provided on the website[41] to show users how they can install, setup and customize the software. A well kown project using the UIMA components is U-compare[42] which has recently received the UIMA Award 2008. It is an text mining/NLP system which includes the world largest UIMA component repository. *"U-Compare allows users to build complex NLP workflows via an easy drag-and-drop interface, and makes visualization and comparison of the outputs of these workflows simple"* [29]. Although UIMA has been moved to the Apache project web site in June 2008, some packages remain on the IBM web site[43] (e.g. SemanticSearch and IBM UIMA Adapter).

[35]Snowball http://snowball.tartarus.org/
[36]BSF http://jakarta.apache.org/bsf
[37]Beanshell http://www.beanshell.org/
[38]Rhino Javascript http://www.mozilla.org/rhino
[39]Jython http://jython.sourceforge.net/Project/index.html
[40]JRuby http://jruby.codehaus.org/
[41]http://incubator.apache.org/uima/
[42]http://www.u-compare.org/index.html
[43]IBM UIMA web site: http://www.alphaworks.ibm.com/tech/uima

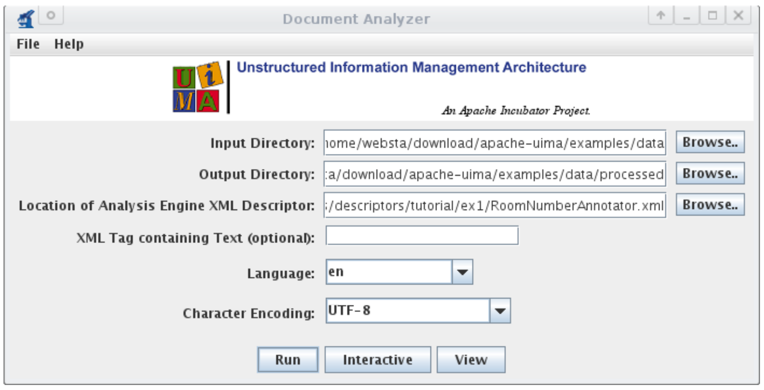
Apache UIMA is available for download on the web site and includes all required libraries, documentation, examples and Eclipse plugins. To get in touch with the programming interface, Apache provides a good tutorial which describes the provided examples. Furthermore the UIMA examples could be imported into Eclipse, which made it easy to handle the annotator classes and the corresponding descriptor files. UIMA works with so called annotators which are responsible for identify relevant patterns in a document and attach a feature structure (type and set of attribute-value pairs) describing the matched tokens in more detail (metadata). One or more Annotators are combined into an Analysis Engine (AE) and forms a analysis logic for an document. All annotations created are represented in the UIMA Common Analysis Structure (CAS). *"The CAS is the central data structure through which all UIMA components communicate"* [29].

The first example to start with in the tutorial is a room number annotator:

```java
/**
 * Example annotator that detects room numbers using Java 1.4 regular expressions.
 */
public class RoomNumberAnnotator extends JCasAnnotator_ImplBase {
  private Pattern mYorktownPattern = Pattern.compile("\\b[0-4]\\d-[0-2]\\d\\d\\b");
  private Pattern mHawthornePattern = Pattern.compile("\\b[G1-4][NS]-[A-Z]\\d\\d\\b");

  /**
   * @see JCasAnnotator_ImplBase#process(JCas)
   */
  public void process(JCas aJCas) {
    // get document text
    String docText = aJCas.getDocumentText();
    // search for Yorktown room numbers
    Matcher matcher = mYorktownPattern.matcher(docText);
    while (matcher.find()) {
      // found one - create annotation
      RoomNumber annotation = new RoomNumber(aJCas);
      annotation.setBegin(matcher.start());
      annotation.setEnd(matcher.end());
      annotation.setBuilding("Yorktown");
      annotation.addToIndexes();
    }
    // search for Hawthorne room numbers
    matcher = mHawthornePattern.matcher(docText);
    while (matcher.find()) {
      // found one - create annotation
      RoomNumber annotation = new RoomNumber(aJCas);
      annotation.setBegin(matcher.start());
      annotation.setEnd(matcher.end());
      annotation.setBuilding("Hawthorne");
      annotation.addToIndexes();
    }
  }
}
```

An application called DocumentViewer provided with the package helps you to test the annotator. The main screen offers some input fields to supply the directory of the input documents, the output directory and the location of the annotator description.


Document Analyzer
File Help
Unstructured Information Management Architecture
An Apache Incubator Project
Browse..
Run
Interactive
View

Input Directory: ıome/websta/download/apache-uima/examples/data
Output Directory: :a/download/apache-uima/examples/data/processed
Location of Analysis Engine XML Descriptor: :/descriptors/tutorial/ex1/RoomNumberAnnotator.xml
XML Tag containing Text (optional):
Language: en
Character Encoding: UTF-8

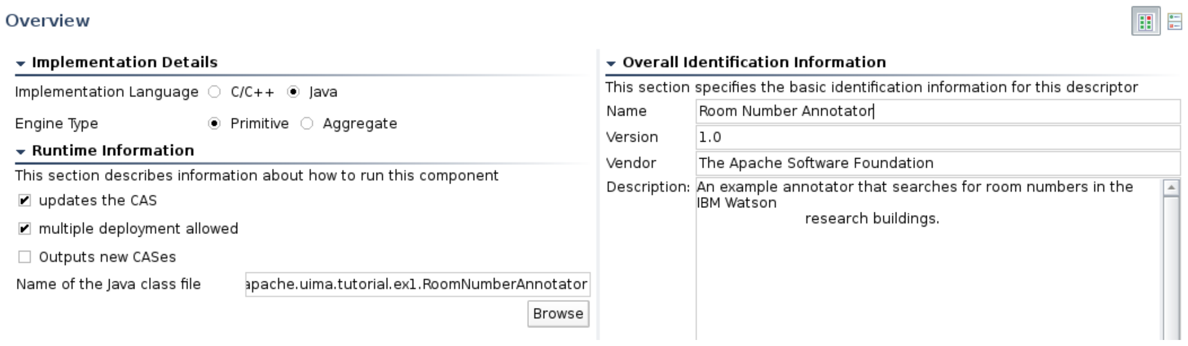
Figure 3.24: UIMA screenshot: The main window

The description must be provided to UIMA at runtime and includes the following information[29]:

- Name, description, version, and vendor
- The annotator's inputs and outputs, defined in terms of the types in a Type System Descriptor
- Declaration of the configuration parameters that the annotator accepts

This file can be comfortable be edited by using the Component Descriptor Editor available for Eclipse.

After providing the required information to the Document Analyzer, you can start interactive mode (a dialog opens where you can input your text) or run the analyse process on the supplied documents. When analysis finishes a dialog shows you all analysed documents.


Overview

Implementation Details

Implementation Language ○ C/C++ [✓ Java]

Engine Type [✓ Primitive] ○ Aggregate

Runtime Information

This section describes information about how to run this component

[✓] updates the CAS

[✓] multiple deployment allowed

☐ Outputs new CASes

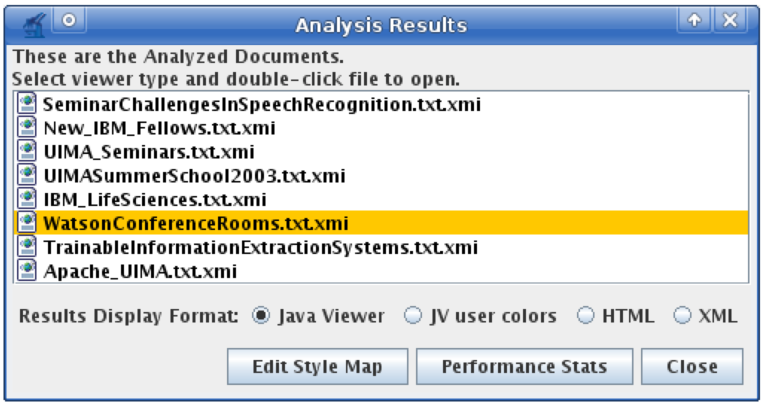
Name of the Java class file: apache.uima.tutorial.ex1.RoomNumberAnnotator

Overall Identification Information

This section specifies the basic identification information for this descriptor

Name	Room Number Annotator
Version	1.0
Vendor	The Apache Software Foundation
Description:	An example annotator that searches for room numbers in the IBM Watson research buildings.

Figure 3.25: UIMA screenshot: The component descriptor editor in Eclipse


Analysis Results

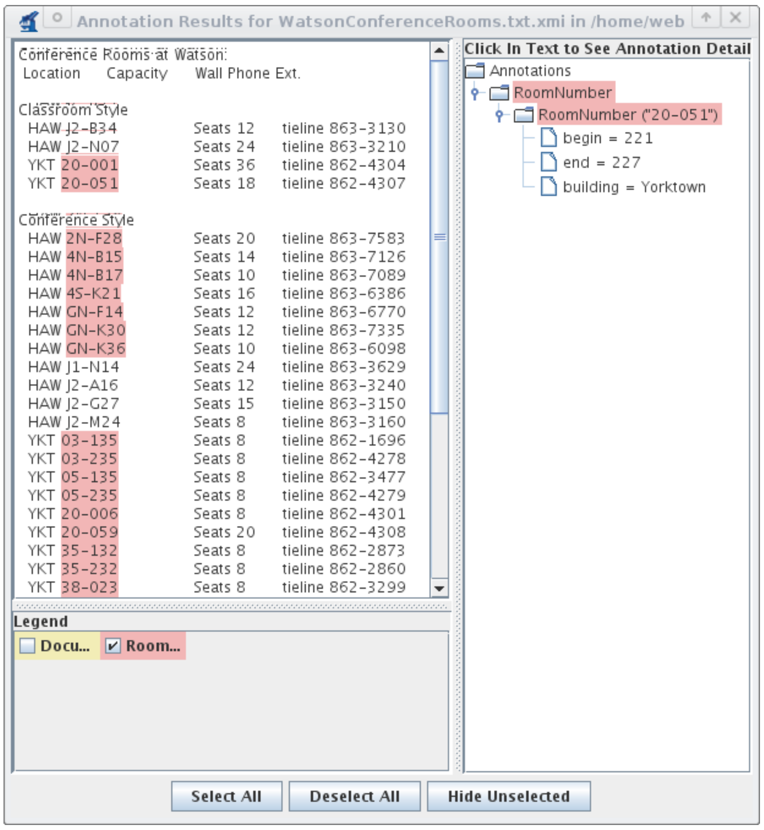
These are the Analyzed Documents.
Select viewer type and double-click file to open.

- SeminarChallengesInSpeechRecognition.txt.xmi
- New_IBM_Fellows.txt.xmi
- UIMA_Seminars.txt.xmi
- UIMASummerSchool2003.txt.xmi
- IBM_LifeSciences.txt.xmi
- WatsonConferenceRooms.txt.xmi
- TrainableInformationExtractionSystems.txt.xmi
- Apache_UIMA.txt.xmi

Results Display Format: [✓ Java Viewer] ○ JV user colors ○ HTML ○ XML

Figure 3.26: UIMA screenshot: After analysis, select a document for display

By double clicking one, the annotator output is visualised in a further dialog where you can control the annotations.


Annotation Results for WatsonConferenceRooms.txt.xmi in /home/web
Click In Text to See Annotation Detail
Legend
Docu...
Room...
Select All
Deselect All
Hide Unselected

```
Conference Rooms at Watson:
 Location      Capacity       Wall Phone Ext.

Classroom Style
  HAW J2-B34                    Seats 12      tieline 863-3130
  HAW J2-N07                    Seats 24      tieline 863-3210
  YKT 20-001                    Seats 36      tieline 862-4304
  YKT 20-051                    Seats 18      tieline 862-4307

Conference Style
  HAW 2N-F28                    Seats 20      tieline 863-7583
  HAW 4N-B15                    Seats 14      tieline 863-7126
  HAW 4N-B17                    Seats 10      tieline 863-7089
  HAW 4S-K21                    Seats 16      tieline 863-6386
  HAW GN-F14                    Seats 12      tieline 863-6770
  HAW GN-K30                    Seats 12      tieline 863-7335
  HAW GN-K36                    Seats 10      tieline 863-6098
  HAW J1-N14                    Seats 24      tieline 863-3629
  HAW J2-A16                    Seats 12      tieline 863-3240
  HAW J2-G27                    Seats 15      tieline 863-3150
  HAW J2-M24                    Seats 8       tieline 863-3160
  YKT 03-135                    Seats 8       tieline 862-1696
  YKT 03-235                    Seats 8       tieline 862-4278
  YKT 05-135                    Seats 8       tieline 862-3477
  YKT 05-235                    Seats 8       tieline 862-4279
  YKT 20-006                    Seats 8       tieline 862-4301
  YKT 20-059                    Seats 20      tieline 862-4308
  YKT 35-132                    Seats 8       tieline 862-2873
  YKT 35-232                    Seats 8       tieline 862-2860
  YKT 38-023                    Seats 8       tieline 862-3299
```

```
Annotations
  RoomNumber
    RoomNumber ("20-051")
      begin = 221
      end = 227
      building = Yorktown
```

Figure 3.27: UIMA screenshot: DocumentAnalyzer showing the annotations

To change the annotation behavior, you only need to change the annotator source code. Different examples are provided for sentence tokenizer, entity recognition and others.

3.12 LingPipe

LingPipe is a Java toolkit for linguistic analysis of human language developed by Alias-i which started 1995 as a collaboration of graduate students at the University of Pennsylvania. It supports HTML, XML and plain text and provides a command line for your analysis. Some demos also includes a GUI or a web interface but a GUI is generally not included. Alias-i distributes this toolkit under its Royalty Free License Version 1 which *"governs the copying, modifying, and distributing of the computer program or work con-*

taining a notice stating that it is subject to the terms of this License and any derivative works of that computer program or work" [30].

A feature overview of LingPipe's information extraction and data mining tools [30]:

- *track mentions of entities (e.g. people or proteins)*
- *link entity mentions to database entries*
- *uncover relations between entities and actions*
- *classify text passages by language, character encoding, genre, topic, or sentiment*
- *correct spelling with respect to a text collection*
- *cluster documents by implicit topic and discover significant trends over time*
- *provide part-of-speech tagging and phrase chunking*

Alias-i provide an extensive online documentation with comprehensive tutorials for all NLP tasks possible with LingPipe eg. classification, named entity recognition, clustering, POS tagging, senctence detection, ... Furthermore a javadoc as well as setup instructions are available in the website[44]. For further reading, a LingPipe blog[45] is accessable at WordPress.com.

To test this implementation free downloading of the core package[46] and for a POS test you need a corpus too. I decided to download the Brown corpus[47] (link is also available on the Lingpipe web site). After downloading these packages, extract them and start training the POS tagger on the Brown corpus. Detail information is available in the tutorial section[48] on the web site. For my tests I used the ant build file in the directory */lingpipe-directory/demos/tutorial/posTags* and adapted the *data.dir* and *data.pos.brown* parameter to train it with the Brown corpus: *ant -f build.xml model-brown*. The output of this operation is a model file and the following processing information:

```
ant -f build.xml model-brown

Buildfile: build.xml
compile:
model-brown:
     [java] n-gram=8
     [java] num chars=256
     [java] lambda fact=8.0
     [java] corpus parser=BrownPosCorpus
```

[44]http://alias-i.com/lingpipe/
[45]LingPipe blog http://lingpipe-blog.com/
[46]Lingpipe core package http://alias-i.com/lingpipe/web/download.html
[47]http://prdownloads.sourceforge.net/nltk/nltk-data-0.3.zip
[48]http://alias-i.com/lingpipe/demos/tutorial/posTags/read-me.html

```
    [java] corpus file=/home/websta/download/lingpipe-3.7.0/nltk-data-0.3/
       brown.zip
    [java] model file=../../models/pos-en-general-brown.HiddenMarkovModel

BUILD SUCCESSFUL
Total time: 47 seconds
```

The parameters N-GRAM, NUM_CHARS and LAMBDA are for the Hidden Markov Model (HMM) to define *"how many characters to use as the basis for the model, the total number of characters, and an interpolation parameter for smoothing"*[30].

After generation of the HMM I started POS tagging of an english text from NY Times. The script uses input from standard input where I pasted the text and presents the following output to standart out:

```
ant -f build.xml run-brown
Buildfile: build.xml

compile:

run-brown:
    [java] Reading model from file=../../models/pos-en-general-brown.
       HiddenMarkovModel
    [java]
    [java]
Google is one of a number of companies devising ways to control the demand for
   electric power as an alternative to building more power plants. The company
    has developed a free Web service called PowerMeter that consumers can use
   to track energy use in their house or business as it is consumed.
    [java] INPUT>
    [java] FIRST BEST
    [java] Google_np is_bez one_cd of_in a_at number_nn of_in companies_nns
       devising_vbg ways_nns to_to control_vb the_at demand_nn for_in
       electric_jj power_nn as_cs an_at alternative_nn to_in building_vbg
       more_ap power_nn plants_nns ._. The_at company_nn has_hvz developed_vbn
        a_at free_jj Web_np service_nn called_vbd PowerMeter_np that_cs
       consumers_nns can_md use_vb to_to track_vb energy_nn use_nn in_in
       their_pp\$ house_nn or_cc business_nn as_cs it_pps is_bez consumed_vbn
       ._.
    [java]
    [java] N BEST
    [java] #  JointLogProb      Analysis
    [java] 0   -605,213 Google_np  is_bez one_cd  of_in  a_at  number_nn
       of_in  companies_nns devising_vbg ways_nns to_to control_vb  the_at
```

```
        demand_nn  for_in  electric_jj  power_nn  as_cs  an_at  alternative_nn
        to_in  building_vbg more_ap  power_nn  plants_nns ._.   The_at
        company_nn  has_hvz developed_vbn a_at  free_jj  Web_np  service_nn
        called_vbd PowerMeter_np that_cs  consumers_nns can_md  use_vb  to_to
        track_vb  energy_nn  use_nn  in_in  their_pp\$ house_nn  or_cc
        business_nn  as_cs  it_pps is_bez consumed_vbn ._.
    [java] 1  -605,549 Google_np  is_bez one_pn  of_in  a_at  number_nn
        of_in  companies_nns devising_vbg ways_nns to_to control_vb  the_at
        demand_nn  for_in  electric_jj  power_nn  as_cs  an_at  alternative_nn
        to_in  building_vbg more_ap  power_nn  plants_nns ._.   The_at
        company_nn  has_hvz developed_vbn a_at  free_jj  Web_np  service_nn
        called_vbd PowerMeter_np that_cs  consumers_nns can_md  use_vb  to_to
        track_vb  energy_nn  use_nn  in_in  their_pp\$ house_nn  or_cc
        business_nn  as_cs  it_pps is_bez consumed_vbn ._.

   <other sentences>

    [java]
    [java] CONFIDENCE
    [java] #  Token        (Prob:Tag)*
    [java] 0  Google           0,811:np     0,119:uh     0,067:nn     0,001:
        nps    0,001:np\$
    [java] 1  is               0,999:bez    0,000:pp\$\$   0,000:np
        0,000:vbz    0,000:dt
    [java] 2  one              0,549:cd     0,434:pn     0,015:vbn    0,001:
        rb     0,000:jj
    [java] 3  of               1,000:in     0,000:rb     0,000:vbz    0,000:
        nn     0,000:rp
    [java] 4  a                1,000:at     0,000:np     0,000:nn     0,000:
        np\$    0,000:pn\$
    [java] 5  number           1,000:nn     0,000:jj     0,000:vbn    0,000:
        vb     0,000:np
    [java] 6  of               1,000:in     0,000:nn     0,000:jj     0,000:
        nns\$   0,000:nn\$

  <other tokens>

    [java]
    [java]
    [java] INPUT>
    [java] Java Result: 130
```

The output is divided into three parts. In the first part, input text is split up into sentences which are split up into tokens. Each token is labeled with syntactic labels showing the tokens word-category disambiguation. This kind of output is called the ”first-best”. Second part contains the ”nth best” ouput, generates a score for the annotation of each token of the sentence and displays the score next to the annotated tokens followed by the second best score for the same tokens with slightly other annotation. Finally, the third part of the output presents a table that shows the confidence of the correctness of the assigned annotation e.g. the tagger has a 81.1% confidence that ”Google” is a proper noun in singular form. With a propability of 11.9% it could also be a interjection.

Key points of the code behind this operation are a regular expression tokenizer, generated and used by the statements

```
HiddenMarkovModel hmm = (HiddenMarkovModel) <__ObjectInputStream__>.readObject
    ();
HmmDecoder decoder = new HmmDecoder(hmm);
static TokenizerFactory TOKENIZER_FACTORY = new RegExTokenizerFactory("(-|'|\d
    |\p{L})+|\S");
Tokenizer tokenizer = TOKENIZER_FACTORY.tokenizer(<charArray>,<startIndex>,<
    endIndex>);
String[] tokens = tokenizer.tokenize();
```

The first output is generated by the statement:

```
String[] tags = decoder.firstBest(tokens);
        System.out.println("\nFIRST BEST");
        for (int i = 0; i < tokens.length; ++i)
            System.out.print(tokens[i] + "_" + tags[i] + " ");
```

The nth best is calculated by:

```
Iterator nBestIt = decoder.nBest(tokens);
for (int n = 0; n < MAX_N_BEST && nBestIt.hasNext(); ++n)
{
  ScoredObject tagScores = (ScoredObject) nBestIt.next();
    double score = tagScores.score();
    String[] tags = (String[]) tagScores.getObject();
    System.out.print(n + " " + format(score) + " ");
    for (int i = 0; i < tokens.length; ++i)
      System.out.print(tokens[i] + "_" + pad(tags[i],5));
    System.out.println();
}
```

To get the confidence output, these statements are required:

```java
TagWordLattice lattice = decoder.lattice(tokens);
for (int tokenIndex = 0; tokenIndex < tokens.length; ++tokenIndex)
{
  ScoredObject[] tagScores = lattice.log2ConditionalTags(tokenIndex);
  System.out.print(pad(Integer.toString(tokenIndex),4));
  System.out.print(pad(tokens[tokenIndex],15));
    for (int i = 0; i < 5; ++i)
  {
    double logProb = tagScores[i].score();
    double conditionalProb = Math.pow(2.0,logProb);
    String tag = (String) tagScores[i].getObject();
        System.out.print(" " + format(conditionalProb) + ":" + pad(tag,4));
  }
}
```

3.13 Orange

This toolkit is an open source C++ data mining software which provides techiques for preprocessing, modelling and data exploration. It is developed by the University of Ljubljana and licenced under GPL. Orange can be accessed either directly by command line, *"through python scripts or through GUI objects called Orange Widgets"* [31]. Existing, time-critical components are written in C++ but a scripting interface for python is included in a way that the user can test some ideas interactively. Additional components can be therefore be written in C++ whereas python is used as a glue language.

Some yet implemented features of Orange include [31]:

- *Orange can read from and write to tab-delimited files and C4.5 files, and supports also some more exotic formats.*
- *Preprocessing covers feature subset selection, discretization, feature utility estimation for predictive tasks.*
- *Predictive modelling includes classification trees, naive bayesian classifer, k-NN, majority classifier, support vector machines, logistic regression, rule-based classifiers (e.g., CN2).*
- *Ensemble methods including boosting, bagging, and forest trees.*
- *Data description methods provides various visulizations (in widgets), self-organizing maps, hierarchical clustering, k-means clustering, multi-dimensional scaling, and other.*

- *Model validation techniques, that include different data sampling and validation techniques (like cross-validation, random sampling, etc.), and various statistics for model validation (classification accuracy, AUC, sensitivity, specificity, ...).*

This project provides an extensive online documentation for all features, a forum as well as example python scripts and data sets on its website[49].

To install orange, your system has to meet the requirements (see installation instructions on the web site). There is no precompiled package available, but a subversion repository where you can download the latest revision. After downloading you have to build the system with the supplied make file. In my tests I tried to classify a dataset containing *"votes for each of the U.S. House of Representatives Congressmen on the 16 key votes; a class is a representative's party"*[31]. The classification should find characteristica of republican and democratic votes and determine the party on the basis of the 16 key votes. The data set looks like (file *voting.tab*):

```
republican n y n y y y n n n y  y y y n y
republican n y n y y y n n n n n y y y n
democrat   y y   y y n n n n y n y y n n
democrat n y y n   y n n n n y n y n n y
democrat y y y n y y n n n n y  y y y y
```

To use the orange python package you only have to import it:

```
import orange
data = orange.ExampleTable("voting")   // file called voting.tab
classifier = orange.BayesLearner(data)
print "Possible classes:", data.domain.classVar.values
print "Probabilities for democrats:"
for i in range(5):
  p = classifier(data[i], orange.GetProbabilities)
  print "%d: %5.3f (originally %s)" % (i+1, p[1], data[i].getclass())
```

This part of code prints out the probability that a person with the supplied votes is member of the republican or of the democratic party.

```
Possible classes: <republican, democrat>
Probabilities for democrats:
1: 0.000 (originally republican)
2: 0.000 (originally republican)
3: 0.005 (originally democrat)
4: 0.998 (originally democrat)
5: 0.957 (originally democrat)
```

[49]http://www.ailab.si/orange/

Important for this example is the correct installation of the right versions that are required to build the package.

4 Classification

Here I'll try to pack the implementations from previous chapter into an table to compare them. Important criteria would be availability (open source/costs), implementing algorithms, usability, references and so on.

This section shows the characteristics of the investigated implementions in a clearly arranged table. The relevance of the criteria and their importance are discussed below. The next section (5) recycles this table and summarises the analysis of the toolkits. The criteria are:

- web service or available for download
- open source
- programming language (java, pyhton, c/c++, other)
- licence model
- documentation (installation guides, how-tos, tutorial, API documentation)
- examples
- system requirements (additional software packages)
- setup
- usability (generating results with the help of documentation)
- GUI available
- visualisation
- community activity (how active is the community - forum, mailing lists)

Our paper only evaluates free available software toolkits in contrast to an evaluation[1] of fourteen desktop data mining tools[32] which tested tools ranging from 75 USD to 25,000 USD. While costs for aquirement should be low whereas quality should reach a maximum. Setting up relevant characteristics to reach the maximum is a very subjective process but we are the opinion, that our twelve parameters are a good start separate the wheat from the chaff.

[1] http://www.dataminninglab.com/TOOLCOMPARISON/tabid/58/Default.aspx

	allocation			progr. language					documentation				
	web service	download package	open source	Java	Python	C/C++	other	licence model	installation manual	tutorial, how-to	book	examples	additional
Rapidminer		✓	✓	✓				AGPL	+	++			[1][2][3][4]
NLTK		✓	✓		✓			GPLv2	++	++	✓	++	[1][4][6]
GATE		✓	✓	✓				LGPL	++	++		+	[4][6]
OpenNLP		✓	✓	✓				Apache v2	○	○			
WEKA		✓	✓	✓				GPL	++	++	✓		[2][6]
JWebPro		✓	✓	✓				GPL	○	○		○	
Term Extractor	✓									○			
TreeTagger		✓					[10]	[11]	○	–			
Open Calais	✓									++		++	[3][12]
Apache UIMA		✓	✓	✓		✓	✓	Apache	+	++		+	[2][3][6]
LingPipe		✓	✓	✓				RFL[13]	++	++			[12]
Orange		✓	✓		✓	✓		GPL	++	+			[3]

	system requ.							
	Windows	Unix / Linux	MacOS X	setup	usability	GUI	visualisation	community activity
Rapidminer	√	√	√	++	++	√	++	++[5]
NLTK	√	√	√	++	++	[15]	[15]	++
GATE	√	√	√	++	+	++	○	++
OpenNLP	√[7]	√[7]	√[7]	+	○	[15]	[15]	+
WEKA	√	√	√	++	+	++	+	++[5]
JWebPro	√	√	√	+	+			Mid'2007[8]
Term Extractor	√[9]	√[9]	√[9]		++	○	[15]	–
TreeTagger	√[9]	√[9]	√[9]		++	[15]	[15]	Mid'2007[8]
Open Calais	√[9]	√[9]	√[9]		++	+	○	++
Apache UIMA	√	√	√	++	+	+	○	++
LingPipe	√	√	√	++	+	[15]	[15]	+[5]
Orange	√[14]	√[14]	√[14]	○	++	[15]	[15]	+

[1]Courses, trainings
[2]Wiki
[3]Forum
[4]Movie
[5]Community edition, company supported
[6]Mailing list
[7]Apache Ant
[8]No indication on the website found, timestamp of the binaries
[9]Browser based solution
[10]Perl
[11]Permission for evaluation, research and teaching purposes
[12]Blog
[13]Royality free licence v1
[14]Additional packages required for building and visualisation
[15]No graphical interface available

5 Conclusion

At the beginning of this thesis, my knowledge about the diversity of implementation in the fields of IE and especially tagging was limited to two or three software packages. During literature research an immense number of available tools I had troubles to select the most interesting implementations available. But soon I discovered, that some packages were massively reused by others (eg. WEKA, OpenCalais) whereas others reuses other basic tools (eg. OpenNLP, Rapidminer, GATE). When I studied the available manuals, how-tos and forum entries I registered that the prevalence of the tools massively depends on the community supporting it. Some implementations are commercial projects that have a free community edition. These packages are very mature and have very good documentation (eg. Rapidminer, OpenCalais, LingPipe).

After all, it is not possible to present an overall winner at this point. The useage and the experience with other IE toolkits and programming languages will definitely have a strong impact on to selection process.

Some suggestions that you should think about before selecting your implementation:

- Keep in mind, that some of the tools provide plenty of interfaces to import and export different formats which would reduce sunk costs if you are unsatisfied with your first choice.
- The implementations with GUI (Rapidminer, WEKA, OpenCalais, GATE) are more suitable for unexperienced users. So the users do not have to learn a programming language first but can concentrate on their IE tasks.
- Start with a toolkit that is well documented and that has a active community. It could become very frustrating if noone answers your posts in the forum.

Bibliography

[1] 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.

[2] Sándor Dominich. *The Modern Algebra of Information Retrieval (The Information Retrieval Series).* Springer, 1 edition, April 2008.

[3] G. Salton. Automatic phrase matching. In *D. G. Hays, Readings in Automatic Language Processing*, pages 169–188, New York, NY, USA, 1966. American Elsevier Publishing Co.

[4] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing.* MIT Press, Cambridge, MA, USA, 1999.

[5] Brown corpus homepage. http://khnt.aksis.uib.no/icame/manuals/brown/, last visited 2009-02-16.

[6] Rapidminer homepage. http://rapid-i.com/content/blogcategory/38/21/lang,en/, last visited 2008-12-29.

[7] Open source business foundation, award 2008 - impressionen und gewinner. http://www.osbf.de/de/node/1199, last visited 2008-12-29.

[8] Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. Yale: Rapid prototyping for complex data mining tasks. In Lyle Ungar, Mark Craven, Dimitrios Gunopulos, and Tina Eliassi-Rad, editors, *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940, New York, NY, USA, August 2006. ACM.

[9] Semanticproxy homepage. http://semanticproxy.opencalais.com/about.html, last visited 2009-01-01.

[10] Nitin Madnani. Getting started on natural language processing with python. *Crossroads*, 13(4):5–5, 2007.

[11] Natural language toolkit homepage. http://www.nltk.org/, last visited 2008-12-29.

[12] General architecture for text engineering homepage. http://gate.ac.uk/, last visited 2008-12-29.

[13] Open nlp homepage. http://opennlp.sourceforge.net/, last visited 2008-12-29.

[14] Opennlp maxent homepage. http://maxent.sourceforge.net/, last visited 2009-01-03.

[15] Weka homepage. http://www.cs.waikato.ac.nz/ ml/index.html, last visited 2009-01-04.

[16] Jwebpro - a java based web processing toolkit homepage. http://jwebpro.sourceforge.net/, last visited 2008-12-30.

[17] Crftagger homepage. http://crftagger.sourceforge.net/, last visited 2009-01-04.

[18] Kay Peterson Zhiyi Song Stephanie Strassel, Mark Przybocki and Kazuaki Maeda. Linguistic resources and evaluation techniques for evaluation of cross-document automatic content extraction. In European Language Resources Association (ELRA), editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may 2008.

[19] G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. The automatic content extraction (ace) program–tasks, data, and evaluation. *Proceedings of LREC 2004*, pages 837–840, 2004.

[20] The linguistic data consortium homepage. http://ldc.upenn.edu/, last visited 2009-01-05.

[21] Linguistic computing laboratory homepage. http://lcl2.uniroma1.it/home.jsp, last visited 2009-01-05.

[22] Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *In International Conference on New Methods in Language Processing, Manchester, UK*, pages 44–49. 1994.

[23] Helmut Schmid. Improvements in part-of-speech tagging with an application to german. In *In Proceedings of the ACL SIGDAT-Workshop*, pages 47–50, 1995.

[24] Treetagger homepage. http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/, last visited 2009-01-05.

[25] Open calais homepage. http://www.opencalais.com/, last visited 2008-12-30.

[26] Thinkpedia homepage. http://thinkpedia.cs.auckland.ac.nz/, last visited 2009-01-01.

[27] Linkedfacts homepage. http://www.linkedfacts.com/, last visited 2009-01-01.

[28] Apache uima homepage. http://incubator.apache.org/uima/, last visited 2009-02-02.

[29] U-compare homepage. http://www.u-compare.org/index.html, last visited 2009-02-02.

[30] Lingpipe homepage. http://alias-i.com/lingpipe/, last visited 2009-02-02.

[31] Orange homepage. http://www.ailab.si/orange/, last visited 2009-02-02.

[32] Ph. D, Brian Gomolka, Eric Schmidt, Marguerite Summers, and Kevin Toop. Evaluation of fourteen desktop data mining tools. In *IEEE International Conference on Systems, Man, and Cybernetics*, October 1998.

List of Figures