ASV Toolbox – A Modular Collection of Language Exploration Tools

Chris Biemann, Uwe Quasthoff, Gerhard Heyer, Florian Holz

NLP Group, Department of Computer Science
University of Leipzig, PF 100920
D-04009 Leipzig/Allemange
{biem|quasthoff|heyer|holz}@informatik.uni-leipzig.de

Abstract

ASV Toolbox is a modular collection of tools for the exploration of written language data both for scientific and educational purposes. It includes modules that operate on word lists or texts and allow to perform various linguistic annotation, classification and clustering tasks, including language detection, POS–tagging, base form reduction, named entity recognition, and terminology extraction. On a more abstract level, the algorithms deal with various kinds of word similarity, using pattern–based and statistical approaches. The collection can be used to work on large real-world data sets as well as for studying the underlying algorithms. Each module of the ASV Toolbox is designed to work either on a plain text files or with a connection to a MySQL database. While it is especially designed to work with corpora of the Leipzig Corpora Collection, it can easily be adapted to other sources.

1. Introduction

Natural Language Processing – in German: Automatische Sprachverarbeitung (ASV) – is increasingly based on statistical and pattern-based methods. These methods are mainly derived from computer science disciplines such as information retrieval, artificial intelligence and machine learning. Applying them to natural language text opens up new perspectives of discovering and describing structure in language which does not necessarily match the traditional linguist's (or computational linguist's) categories. To facilitate better understanding of the structure of natural language in the context of such methods and approaches, an explorative access to the statistical properties of language is indispensable. This is similar to research in physics: Carrying out well-defined physical experiments provides insights into the applicability of existing hypotheses and allows to deepen the understanding of further physical structures and the need for appropriate mathematical models.

Currently, in natural language processing a number of tools and environments are available that partially fulfil this need. Especially we want to mention the following sets of tools.

The IMS Stuttgart offers tools for morphological analysis, tagging and chunking as well as clustering (http://www.ims.uni-stuttgart.de/tcl/).

For the data and text mining algorithms in general, WEKA offers a collection of machine learning algorithms (Witten and Frank, 2000). WEKA contains tools for data preprocessing, classification, regression, clustering, association rules, and visualization. However, apart from very constrained examples, no specific language data are provided.

To apply data and text mining algorithms to natural language processing and related product development, LingPipe offers information extraction and data mining tools (Baldwin and Carpenter, 2003).

In order to combine such tools, GATE (Cunningham et al., 2002) provides infrastructure tools like IBM's Unstructured Information Management Architecture (UIMA). The original design principles behind the system have been to avoid tying developers into any particular theory, to support modular construction of language processing systems and to make it easy to swap components in and out.

Another collection of natural language processing tools is the Natural Language Toolkit (NLTK) (Bird and Loper, 2004). It provides several modules all written in Python for performing different kinds of NLP tasks – namely tokenization, tagging, parsing and formal semantic analysis – and comes with corpus data, too. Besides the main goals of simplicity, consistency, extensibility and modularity one aspect of the NLTK is educational so that it comes with course plans and python guidance but the algorithms are not highly optimized due to clear and easily understandable implementations.

However, the increasing availability of statistical natural language processing methods and large text corpora for many languages now leads to the somewhat paradoxical situation that more than ever language data and algorithms for processing them are available, but most of them are incompatible with each other in terms of data structures and interfaces.

The ASV Toolbox (Natural Language Processing Toolbox) presented here is intended to to allow the exploitation of the potential of large corpora collections and language independent processing methods for advancing the experimental study of languages, the learning of methods to do so, and the development of language processing technologies. It takes a modular approach to language data and algorithms and is based on

(1) a strict detachment of language processing algorithms from particular linguistic data and languages, thus encouraging the collection of language data and corpora, and the development of processing algorithms independently from each other;

(2) language independent processing algorithms, thus allowing for a comparison of languages with respect to certain statistical parameters and other computed features; and

(3) a modular software design, thus enabling algorithms to enrich language data and to re-use enriched data and features as well as adding new algorithms.

At present, the toolbox is set up to use data of the Leipzig Corpora Collection (LCC), a collection of text corpora based on a set size and format (Quasthoff et al., 2006), but can be easily adapted to other data formats. The algorithms it comprises are based on publications by researchers at the Natural Language Processing Division (ASV) at the Computer Science Department of the University of Leipzig (cf. also (Heyer et al., 2006)). The modular software design is kept open for adding new algorithms. Its main use at present is to share processing know-how and to support the teaching of natural language processing, helping students to better understand statistical properties of languages by experimenting with particular algorithms.
The compatibility of the tools with the data format of available text corpora for many languages as well as the compatibility of the tools with each other is a distinguishing feature of the toolbox in comparison to projects and tools like TCL, WEKA, LingPipe, or GATE. In particular, the tools can be iteratively applied to improve and annotate data, resulting in a processing chain of tools.

2. ASV Toolbox

At the moment, the ASV Toolbox 1.0 comprises 12 independent tools, also available as single modules. All tools are written in Java 1.5 and available as open source code as well as compiled binaries (http://wortschatz.uni-leipzig.de /˜cbiemann/software/toolbox/). Additionally to the GUI version, they can also be run with a command line call and the documentation provides examples how to integrate the functionality into custom Java programs. Most tools come with a variety of sample datasets and operate on data directly specified in the GUI, on plain text files and usually on MySQL databases. For users of the Leipzig Corpora Collection (LCC) the parameters are pre-set for a quick start (database format available at http://corpora.informatik.uni-leipzig.de/download.html). The Sections 3. and 4. give a short overview of the currently available tools and their usage with the GUI. Additional examples and more detailed instructions can be found on the ASV Toolbox website (see above).
The architecture of the ASV toolbox is modular. Accordingly, there is no need to install all available modules. It is also easily possible to implement new modules to work with the core framework. These modules can make use of several methods shared across the entire framework such as database and file access, or configuration.
The modules can also be used stand-alone, either using the GUI or the command line interface or the modules' methods directly in another Java program. Therefore the modules are provided as jar files.
The used data formats are designed to allow the reuse of results easily. To achieve this, a unified table format is used for most purposes. The main assumption is that every entity, be it a word, a sentence or a document, can be provided with and later identified by a unique number, Correspondingly the first column of any table stores an identification number, whereas the following column stores other identification numbers that represent annotations, term weights, related or any other information. The only other type of tables is one that maps the identification numbers to the corresponding strings, such as words. The tables are stored in a MySQL database or in a tab-separated file format.
The GUI of every module is organised in panels. Every tool has a welcome panel with a tool description. Then there are different panels for data input, processing and configuration. Panels which are useful for many tools like database configuration are provided by the framework.
In the following, the presently available modules in the ASV Toolbox framework are briefly described. Where appropriate, publications are cited that describe the corresponding algorithm in more detail.

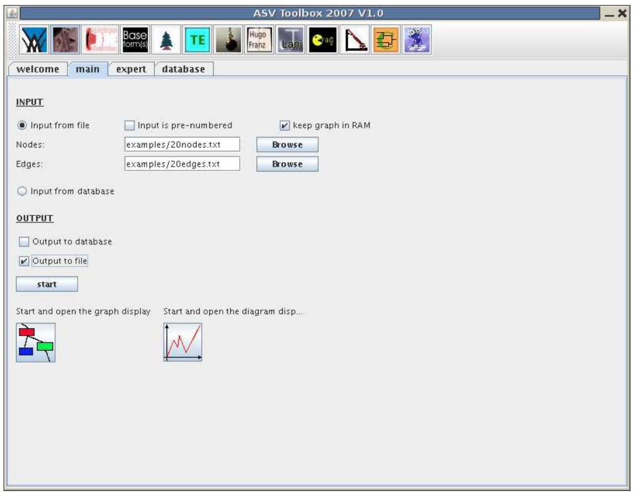
3. General Purpose Tools

3.1. Chinese Whispers Multi-purpose Graph Clustering

Chinese Whispers is a graph clustering algorithm (Biemann, 2006a) which has linear time complexity in the number of edges, which is especially beneficial for sparse graphs, i.e. where the number of egdes is far below the possible number of edges. It has been used for language separation (Biemann and Teresniak, 2005), unsupervised POS tagging (Biemann, 2006b) and word sense induction (Biemann, 2006a). CW is a clustering algorithm that clusters undirected, weighted graphs. The output is a non-hierarchical fuzzy partitioning of the graph. Its application is not bound to language data; the algorithm can partition arbitrary undirected, weighted graphs of any sizes. Best results are obtained on graphs with the recently described small world structure (Watts and Strogatz, 1998; Ferrer i Cancho and Sole, 2001; Steyvers and Tenenbaum, 2005).
The implementation of this algorithm is suitable to be applied on very large data sets (graphs with several millions nodes were tested successfully). For smaller data sets a force-directed graph layout based visualisation has been included. It both helps to browse the data as well as understand the Chinese Whispers algorithm by providing a step-by-step visualisation of its iterations.
A graph is specified by a list of nodes and a list of edges. Nodes have IDs and labels, edges consist of two node ids and an edge weight. The input data can be loaded either from files or from a database. The output can be stored in files or a database, too (see Fig. 1). The clustering result can also be visualized in an interactive manner, where for instances subgraphs can be chosen and extracted for a detailed view. See Figure 2 for the German subgraph from a seven languages separation task (Biemann and Teresniak, 2005).

3.2. Multi-purpose Word Classifier: Pretree Tool

This implementation of Compact Patricia Tree (CPT, pretree) classifiers proved to be useful for morphology-related tasks in e.g. (Witschel and Biemann, 2005; Bordag, 2006) and is used by various other tools in ASV Toolbox. The tool provides possibilities to train and evaluate classifiers that use beginnings or endings of strings as features. An important property of this classifier is that it reproduces the training set classification to 100%. Therefore, pretrees are capable of storing an exception list, while generalizing on unseen examples.



INPUT

[✓ Input from file] Input is pre-numbered [✓ keep graph in RAM]

Nodes: examples/20nodes.txt Browse

Edges: examples/20edges.txt Browse

Input from database

OUTPUT

Output to database

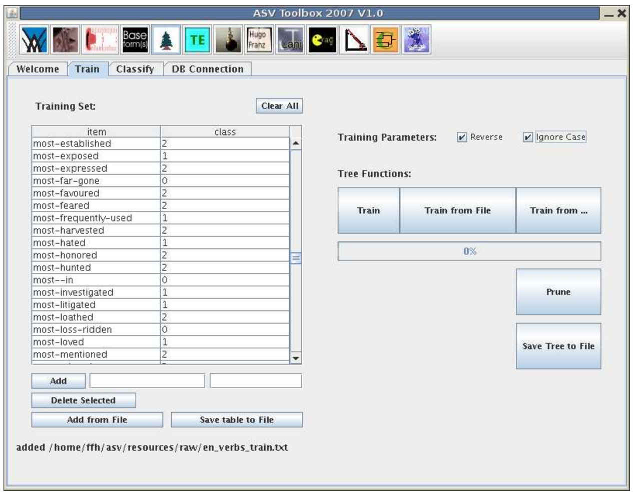
[✓ Output to file]

start

Start and open the graph display

Start and open the diagram disp...

Figure 1: The Chinese whispers main panel



Training Set:

item	class
most-established	2
most-exposed	1
most-expressed	2
most-far-gone	0
most-favoured	2
most-feared	2
most-frequently-used	1
most-harvested	2
most-hated	1
most-honored	2
most-hunted	2
most--in	0
most-investigated	1
most-litigated	1
most-loathed	2
most-loss-ridden	0
most-loved	1
most-mentioned	2

Training Parameters: [✓ Reverse] [✓ Ignore Case]

Tree Functions:

0%

added /home/ffh/asv/resources/raw/en_verbs_train.txt

Figure 3: The pretree train panel

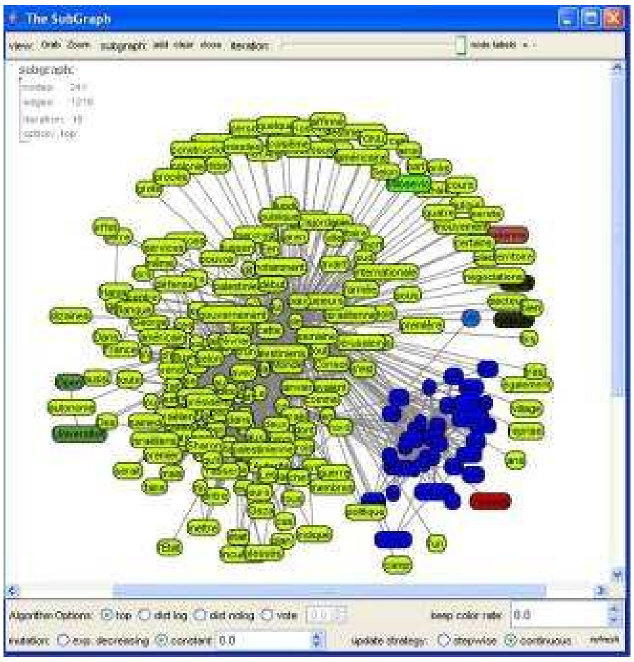
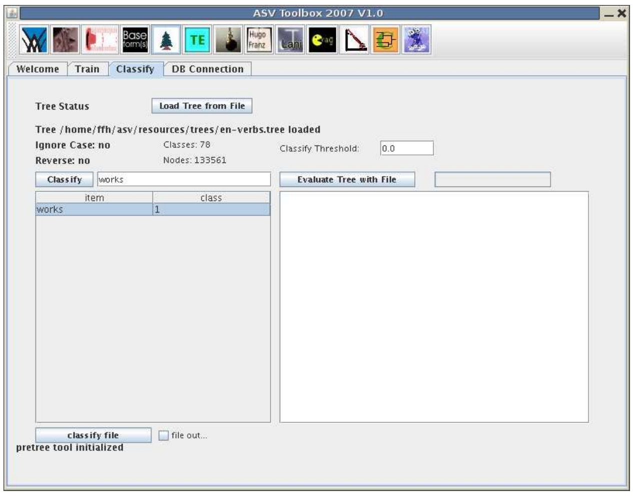


Figure 2: The visualization of a a subgraph after a clustering



Tree Status

Tree /home/ffh/asv/resources/trees/en-verbs.tree loaded

Ignore Case: no Classes: 78 Classify Threshold: 0.0

Reverse: no Nodes: 133561

Classify: works

item	class
works	1

file out...

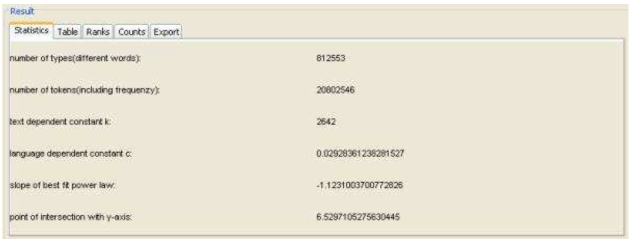
pretree tool initialized

Figure 4: The pretree classify panel

Pretrees can be trained using data from databases, files or directly given input. Trained pretrees can be saved directly or pruned beforehand. The tool allows also to use pretrees for classification (see Fig. 4).

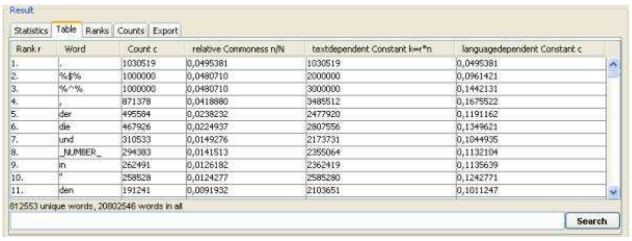
3.3. Zipf's Law Visualization for Corpora

For an introduction to quantitative linguistics, this tool enables to visualize Zipfian distributions (Zipf, 1949) for word frequency lists and documents. Various parameters are computed, rank-frequency lists are browseable and the plot can be exported in various formats. The tool uses input from a file, a database or from plain text input. It extracts the words from the given input, calculates the frequencies and other values of the words and presents the results in tables and in a diagram. The output panel contains several subpanel which report the following results.
On the "Statistics" subpanel, the overall figures for the corpus are summarized (see Fig. 5); the "Table" subpanel lists all word types with their frequency (see Fig. 6). The "Ranks" subpanel provides a similar view as the "Table" subpanel, but for a selected subset of ranks. The "Count" subpanel gives a comparison of the real data and the fitted Zipfian distribution. The results can be exported to an open document table or a csv file. The diagram can be saved as a



number of types(different words):	812553
number of tokens(including frequenzy):	20802546
text dependent constant k:	2642
language dependent constant c:	0.02928361238281527
slope of best fit power law:	-1.1231003700772826
point of intersection with y-axis:	6.5297105275630445

Figure 5: The Zipf distribution analysis tool statistic panel


Result

Statistics | Table | Ranks | Counts | Export

Rank r	Word	Count c	relative Commoness n/N	textdependent Constant k=r*n	languagedependent Constant c
1.	.	1030519	0,0495381	1030519	0,0495381
2.	%$%	1000000	0,0480710	2000000	0,0961421
3.	%^%	1000000	0,0480710	3000000	0,1442131
4.	,	871378	0,0418880	3485512	0,1675522
5.	der	495584	0,0238232	2477920	0,1191162
6.	die	467926	0,0224937	2807556	0,1349621
7.	und	310533	0,0149276	2173731	0,1044935
8.	_NUMBER_	294383	0,0141513	2355064	0,1132104
9.	in	262491	0,0126182	2362419	0,1135639
10.	"	258528	0,0124277	2585280	0,1242771
11.	den	191241	0,0091932	2103651	0,1011247

812553 unique words, 20802546 words in all

Search

Figure 6: The Zipf distribution analysis tool table panel

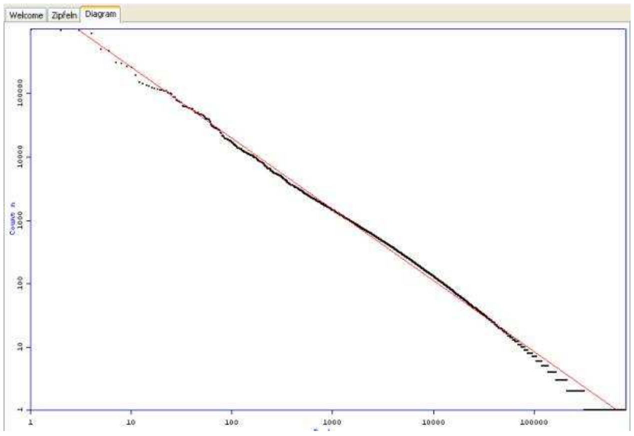


Figure 7: A fitted Zipfian distribution

png file and is displayed in the diagram panel (see Fig. 7).

3.4. Hierarchical Agglomerative Clustering

This basic implementation of hierarchical agglomerative clustering allows clustering elements represented as vectors using various norms and distance measures (Heyer et al., 2006). As example configuration, words can be clustered by their common significant co-occurrences (available from LCC in 15 languages). The result can be exported as XML file or in dendrogram picture format (see Fig. 8).

To determine the distance between clusters based on their member elements, the following distance functions are selectable:single, complete, average, average group and centroid linkage and Ward's method. As distance functions between two vectors the following function are selectable: L1- and L2-norm, Dice and Jaccard coefficient and the cosine.

The clustering tool can read feature vectors from files and from a database.

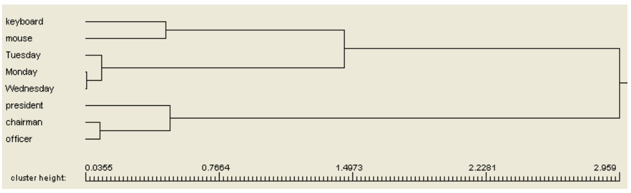
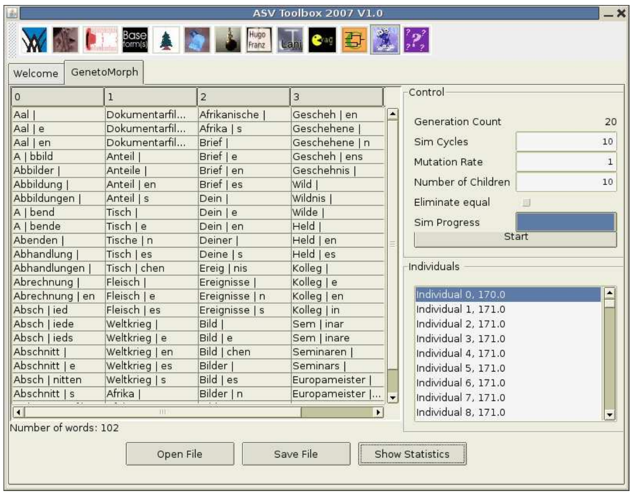


Figure 8: A sample clustering of words for English


ASV Toolbox 2007 V1.0

Welcome | GenetoMorph

0	1	2	3
Aal |	Dokumentarfil...	Afrikanische |	Gescheh | en
Aal | e	Dokumentarfil...	Afrika | s	Geschehene |
Aal | en	Dokumentarfil...	Brief |	Geschehene | n
A | bbild	Anteil |	Brief | e	Gescheh | ens
Abbilder |	Anteile |	Brief | en	Geschehnis |
Abbildung |	Anteil | en	Brief | es	Wild |
Abbildungen |	Anteil | s	Dein |	Wildnis |
A | bend	Tisch |	Dein | e	Wilde |
A | bende	Tisch | e	Dein | en	Held |
Abenden |	Tische | n	Deiner |	Held | en
Abhandlung |	Tisch | es	Deine | s	Held | es
Abhandlungen |	Tisch | chen	Ereig | nis	Kolleg |
Abrechnung |	Fleisch |	Ereignisse |	Kolleg | e
Abrechnung | en	Fleisch | e	Ereignisse | n	Kolleg | en
Absch | ied	Fleisch | es	Ereignisse | s	Kolleg | in
Absch | iede	Weltkrieg |	Bild |	Sem | inar
Absch | ieds	Weltkrieg | e	Bild | e	Sem | inare
Abschnitt |	Weltkrieg | en	Bild | chen	Seminaren |
Abschnitt | e	Weltkrieg | es	Bilder |	Seminars |
Absch | nitten	Weltkrieg | s	Bild | es	Europameister |
Abschnitt | s	Afrika |	Bilder | n	Europameister |...

Number of words: 102

Control

Generation Count	20
Sim Cycles	10
Mutation Rate	1
Number of Children	10
Eliminate equal	☐
Sim Progress	

Start

Individuals

Individual 0, 170.0
Individual 1, 171.0
Individual 2, 171.0
Individual 3, 171.0
Individual 4, 171.0
Individual 5, 171.0
Individual 6, 171.0
Individual 7, 171.0
Individual 8, 171.0

Open File | Save File | Show Statistics

Figure 9: The Genetomorph panel

3.5. Genetic Morphology Analysis: Genetomorph

This tool provides a genetic algorithm that is able to detect morphological regularities in word lists (Kazakov, 1997; Kazakov, 2001). A fitness function that minimizes the cost of describing morphological rules is optimized, individual solutions can be browsed and the progress until convergence is visualized in a plot. Sample data is available for German nouns and adjectives.

The algorithm is initialized with a list of words containing morphological marks. These marks can be set randomly but can also represent given morphological knowledge.

4. Linguistic Tools

4.1. Similar Words: Levenshtein Distance

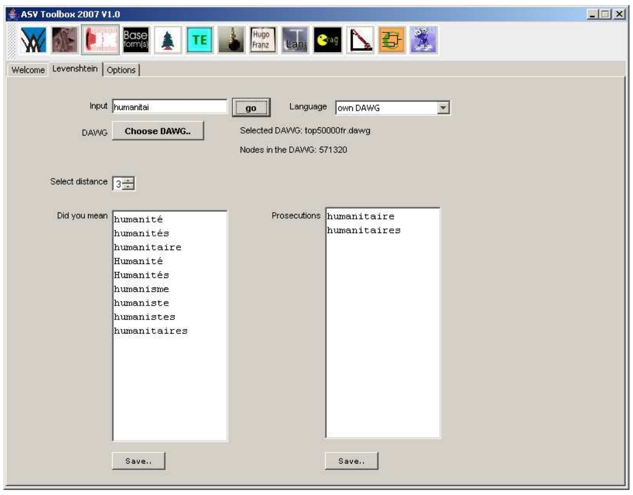
Based on a Directed Acyclic Word Graph (DAWG) implementation, this tool allows efficient basic spell checking by offering words from a given word list with Levenshtein distances (Levenshtein, 1965). As resources, we provide the top frequent 50.000 words for 15 languages. Building DAWGs from custom word lists is possible.

For example, the Italian wordlist returns for the misspelled input word "spagetti" the correct spelling "spaghetti" with distance 1 and offers "spetti", "soggetti", "panetti" and "paletti" with distance 2 (cf. Fig. 10 with the French example "humanitai").

There are two kinds of output. One list contains all words that have the specified maximum amount of Levenshtein distance, sorted in ascending order. The other contains all possible continuations of the entered (possibly partial) word (see Fig. 10).

4.2. Baseform Reduction and Compound Noun Splitting

A number of languages extensively use morphology and compounding, e.g. Germanic languages, Korean, Greek and Finnish. Compared to languages such as English, where (noun) compounds are expressed using several tokens and morphological changes of word strings occur only rarely, this leads to a tremendous increase in vocabulary



Input: humanitai

Language: own DAWG

DAWG

Selected DAWG: top50000fr.dawg

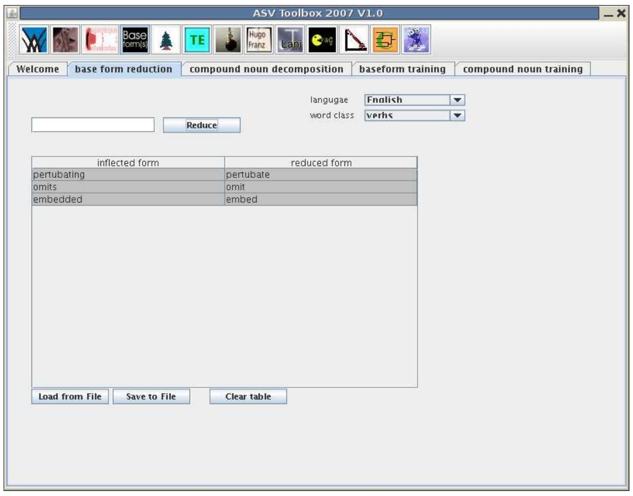
Nodes in the DAWG: 571320

Select distance: 3

Did you mean:
humanité
humanités
humanitaire
Humanité
Humanités
humanisme
humaniste
humanistes
humanitaires

Prosecutions:
humanitaire
humanitaires

Figure 10: The Levenshtein tool panel

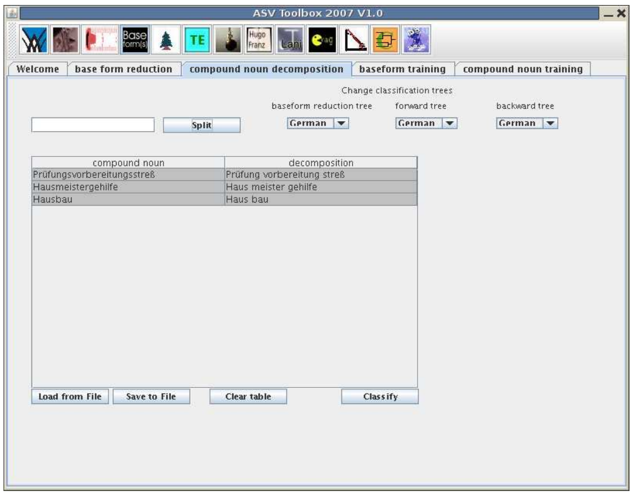


langugae: English

word class: verbs

inflected form	reduced form
pertubating	pertubate
omits	omit
embedded	embed

Figure 11: The base form reduction panel



Change classification trees

baseform reduction tree: German

forward tree: German

backward tree: German

compound noun	decomposition
Prüfungsvorbereitungsstreß	Prüfung vorbereitung streß
Hausmeistergehilfe	Haus meister gehilfe
Hausbau	Haus bau

Figure 12: The compound noun decomposition panel

size. In applications, this results in sparse data, challenging a number of NLP applications. For IR experiments with German, Braschler et al. report that decompounding results in higher text retrieval improvements than stemming (Braschler and Ripplinger, 2003).

Most string classification problems can be solved by using a Compact Patricia Tree (CPT, see Sect. 3.2.). Accordingly, this module, designed to perform baseform reduction and compound splitting, comprises an efficient implementation of a CPT and is bundled with data for base form reduction for English, German and Norwegian, as well as German noun compound splitting.

Applied to real-world problems, such as the baseform reduction, precision and recall depend mostly on the size and variety of the training data and achieves F-values in the high 90% range. CPTs have sucessfullt applied to other tasks including name gender classifications or as a generalization module in an unsupervised morpheme segmentation algorithm (Bordag, 2007). Unsupervised approaches of learning compound splits are described in (Larson et al., 2000; Monz and de Rijke, 2002; Holz and Biemann, 2008). In (Holz and Biemann, 2008) CPTs are built up for splitting and for periphrasing compounds.

To facilitate experimentation with training or unclassified data, the GUI is split into two parts: base form reduction and compound noun decomposition.

The base form reduction panel is designed to reduce word form to their base forms. For this, the language and the part of speech must be provided (see Fig. 11).

The noun compound decomposition panel is designed to specifically split compounds into their consecutive parts and additionally reduce the parts to their base forms.

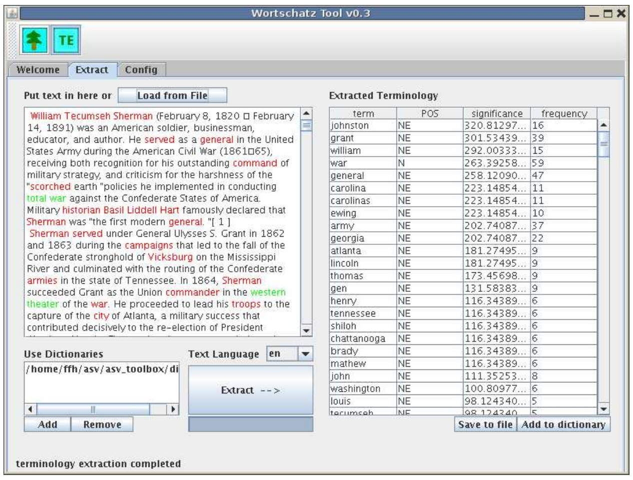
As parameter one of the provided languages or own data in form of a CPT has to be chosen (see Fig. 12). Compound decomposition is implemented by recursively applying CPT classifiers to split parts from the beginning and the end of the word. Once the parts are identified, they are reduced to base form by applying a POS-independent base form reducer.

4.3. Terminology Extraction

Comprising the non-interactive part of (Witschel, 2004)'s terminology extraction method, this tool extracts terminologically relevant terms and phrases from documents by comparing them to a background corpus and given POS-tag patterns. Currently, it is available for English, Finnish and German.

There are several approaches to extract index terms from documents, e.g. statistical (Salton et al., 1975; Cohen, 1995), linguistical (Bourigault, 1992) and hybrid (Daille et al., 1994). One important statistical method is the so-called differential analysis which measures the extent to which the frequency of a word w in the given text deviates from its frequency in general usage (Witschel, 2004). The latter frequency is determined using a reference corpus, i.e. a large and well-balanced collection of documents in the given language. The termhood of w is quantified using a statistical significance measure reflecting the significance of the frequency deviation.

The implemented pattern-based methods are based on part-of-speech (POS) information. Here POS-tags are used both



Put text in here or

William Tecumseh Sherman (February 8, 1820 □ February 14, 1891) was an American soldier, businessman, educator, and author. He served as a general in the United States Army during the American Civil War (1861□65), receiving both recognition for his outstanding command of military strategy, and criticism for the harshness of the "scorched earth "policies he implemented in conducting total war against the Confederate States of America. Military historian Basil Liddell Hart famously declared that Sherman was "the first modern general. "[1]
Sherman served under General Ulysses S. Grant in 1862 and 1863 during the campaigns that led to the fall of the Confederate stronghold of Vicksburg on the Mississippi River and culminated with the routing of the Confederate armies in the state of Tennessee. In 1864, Sherman succeeded Grant as the Union commander in the western theater of the war. He proceeded to lead his troops to the capture of the city of Atlanta, a military success that contributed decisively to the re-election of President

Use Dictionaries: /home/ffh/asv/asv_toolbox/di

Text Language: en

Extracted Terminology

term	POS	significance	frequency
johnston	NE	320.81297...	16
grant	NE	301.53439...	39
william	NE	292.00333...	15
war	N	263.39258...	59
general	NE	258.12090...	47
carolina	NE	223.14854...	11
carolinas	NE	223.14854...	11
ewing	NE	223.14854...	10
army	NE	202.74087...	37
georgia	NE	202.74087...	22
atlanta	NE	181.27495...	9
lincoln	NE	181.27495...	9
thomas	NE	173.45698...	9
gen	NE	131.58383...	9
henry	NE	116.34389...	6
tennessee	NE	116.34389...	6
shiloh	NE	116.34389...	6
chattanooga	NE	116.34389...	6
brady	NE	116.34389...	6
mathew	NE	116.34389...	6
john	NE	111.35253...	8
washington	NE	100.80977...	6
louis	NE	98.124340...	5
tecumseh	NE	98.124340...	5

terminology extraction completed

Figure 13: The terminology extraction panel

to restrict the output units to certain word classes - it is often argued that most technical terms are nouns - and to extract multiword units (or phrases) in a shallow way. This is done by extracting sequences of words that appear frequently together and follow certain POS patterns, e.g. noun + noun (”terminology extraction”).
For using the extraction tool, either a text can be directly written into the text area or be loaded from file. It is possible to build and use an own terminology dictionary by saving the extracted terminology as or adding to a dictionary.
At the “Config” Panel the technical parameters can be set. Generally, the predefined default settings should work rather well for medium-sized texts (up to 10,000 words), for larger texts, some of the thresholds may need to be adjusted.

4.4. Gazetteer Bootstrapping: The Pendulum

For building gazetteers for generalized Named Entity Recognition, this tool provides a bootstrapping framework that grows small initial gazetteers using a set of rules and a customizable regular expression tagging. In the case of person names, this search-and verification methodology is able to extract e.g. some 40,000 names starting from a list of 20 with high precision from large plain text corpora, see (Quasthoff et al., 2002).
For using the tool a database which contains the following three tables is needed: a table with words, a table with sentences and table which connect the words and sentences tables. The bootstrapping process can be adjusted with several parameters which can be loaded from a configuration file or set on the configurations panels.
There are five panels where settings can be made. On the “File Management” panel it can be chosen which output files for different item classes and which configuration files to use. On the “Parameters and Settings” panel the database connection and the parameters for the bootstrapping algorithm can be configured. On the “Rules and Patterns” panel the extraction rules and patterns can be added, deleted and saved. On the “Input Items” panel already classified items are to be given to start the bootstrapping. Additional back-

```
<person pattern="TIT PU VN NN">Dr.
Angela Merkel</person> hat <person
pattern="VN NN">Gerhard Schröder
</person> im Amt abgelöst .
```

Figure 14: Sample NER markup on German

ground knowledge about other items can be given. These will not be listed in the item list at the end. On the “Tag System” panel the tag encoding and the regular expression tagging can be configured. Regular expression tagging means that regular expressions for character strings can be used for finding candidates for new items.

4.5. Named Entity Recognition: NameRec

NameRec is a gazetteer- and rule-based Named Entity Recognition tool that make heavy use of gazetteers (e.g. built by the Pendulum as described in the previous section). The extraction patterns and rules are freely configurable. The tool marks plain text with NER markup. Resources are available for German for person names with professions; a sample markup is given in Fig. 14. The configuration panels are very similar to the panels of the pendulum and follow the same design.

4.6. Probabilistic Language Identification at Sentence Level

This state-of-the-art word-based language identification program allows identifying the language at sentence level using frequency word lists of the languages which are to be distinguished (Biemann and Teresniak, 2005). It can be used to identify chunks of foreign language in a corpus. At the moment, 25 languages are supported; the number of languages is easily extensible by providing frequency lists either from a word list file or a database. As input and output either directly given text, a text file or a database can be used. There are no other parameters to be specified.

4.7. POS-tagging with a ViterbiTagger

This simple tagger implementation is based on tag trigrams and tag distributions for words. Not as powerful as a full HMM implementation (Rabiner, 1989), it also uses the viterbi algorithm (Viterbi, 1967) and comes with a morphological back-off component (realized with Pretree, see Sect. 3.2.) and is capable of training tagger models on very large annotated texts in various formats. Further, it allows tagging previously tagged text with a second (e.g. semantic) tag. The format of the tagger model is readable as plain text, which could prove useful for educational purposes. An evaluation framework is included that also deals with evaluating on different tag sets for Gold standard and test. Supervised models are provided for English, German and Finnish, unsupervised tagger models for resource-scarce languages or domain-specific applications are available at http://wortschatz.uni-leipzig.de/˜cbiemann/software/unsupos.html.

5. Your Tool

The “Your Tool” tool is an empty frame which allows to easily implement new tools for the ASV Toolbox framework. There are already four existing classes in the package `de.uni_leipzig.asv.toolbox.yourTool`.

6. Conclusion

We introduced ASV Toolbox, a collection of algorithms suited for natural language processing, data browsing, and training as well as for educational and scientific purposes. Several of the algorithms, such as the Chinese Whispers, are the result of very recent research and are unique to this collection. The standardized graphical user interface helps getting started with complicated algorithms in an intuitive and easy way. Most algorithms are designed to be applicable on very large data sets effortlessly.

The open and modular architecture enables users to employ the implementations in custom projects as well as to add more tools. This Toolbox is not intended to replace any other toolbox, or other existing algorithm implementations. Instead, it is not only meant to be complementary, but due to the simple programming interfaces potentially compatible. Most modules can read data from direct input, from plain text files and from databases, which further stimulates seamless integration of further corpora.

As such, the ASV Toolbox is a contribution to standardize, simplify and operationalize language algorithm distribution and integration.

7. References

Breck Baldwin and Bob Carpenter. 2003. Lingpipe. http://www.alias-i.com/lingpipe/index.html.

C. Biemann and S. Teresniak. 2005. Disentangling from babylonian confusion – unsupervised language identification. In *Proceedings of CICLing-2005, Computational Linguistics and Intelligent Text Processing, Mexico City, Mexico and Springer LNCS 3406*.

C. Biemann. 2006a. Chinese whispers – an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the HLT-NAACL-06 Workshop on Textgraphs*.

C. Biemann. 2006b. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the COLING/ACL-06 Student Research Workshop 2006, Sydney, Australia*.

Steven Bird and Edward Loper. 2004. Nltk: The natural language toolkit. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (Demonstration Track)*, pages 214–217, http://nltk.sourceforge.net/.

S. Bordag. 2006. Two-step approach to unsupervised morpheme segmentation. In *Proceedings of the Unsupervised segmentation of words into morphemes - Challenge 2005, Venice, Italy*.

S. Bordag. 2007. Unsupervised and knowledge-free morpheme segmentation and analysis. In *Proceedings of the Working Notes for the CLEF Workshop 2007, Budapest, Hungary*.

D. Bourigault. 1992. Surface grammatical analysis for the extraction of terminological noun phrases. In *Proceedings of Coling92*, pages 977–981.

M. Braschler and B. Ripplinger. 2003. Stemming and decompounding for german text retrieval. In *Proceedings of ECIR, LLNCS 2633*, pages 177–192.

J.D. Cohen. 1995. Highlights: language and domain independent automatic indexing terms for abstracting. *Journal of the American Society for Information Science*, 46(3):162–174.

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, http://gate.ac.uk/documentation.html.

B. Daille, E. Gaussier, and J. Langé. 1994. Towards automatic extraction of monolingual and bilingual terminology. In *Proceedings of Coling94*, pages 515–521.

Ramon Ferrer i Cancho and Ricard V. Sole. 2001. The small world of human language. In *Proceedings of The Royal Society of London. Series B, Biological Sciences.*, pages 268(1482): 2261–2265.

Gerhard Heyer, Uwe Quasthoff, and Thomas Wittig. 2006. *Text Mining: Wissensrohstoff Text – Konzepte, Algorithmen, Ergebnisse*. W3L-Verlag.

F. Holz and C. Biemann. 2008. Unsupervised and knowledge-free learning of compound splits and periphrases. In A. Gelbukh, editor, *Proceedings of the CICLing 2008, LNCS 4919*, pages 117–127. Springer.

Dimitar Kazakov. 1997. Unsupervised learning of naïve morphology with genetic algorithms. In A. van den Bosch, W. Daelemans, and A. Weijters, editors, *Workshop Notes of the ECML/MLnet Workshop on Empirical Learning of Natural Language Processing Tasks*, pages 105–112, Prague, Czech Republic, April.

Dimitar Kazakov. 2001. Unsupervised learning of word segmentation rules with genetic algorithms and inductive logic programming. *Machine Learning*, 43:121–162, April-May.

M. Larson, D. Willett, J. Köhler, , and G. Rigoll. 2000. Compound splitting and lexical unit recombination for improved performance of a speech recognition system for german parliamentary speeches. In *Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP)*.

Vladimir I. Levenshtein. 1965. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady Akademii Nauk SSSR*, 163(4):845–848.

Ch. Monz and M. de Rijke. 2002. Shallow morphological analysis in monolingual information retrieval for dutch, german, and italian. In *CLEF 2001: Revised Papers from the Second Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems*, pages 262–277.

U. Quasthoff, C. Biemann, and C. Wolff. 2002. Named entity learning and verification: Expectation maximisation in large corpora. In *Proceedings of CoNNL-2002, Taipei, Taiwan*.

U. Quasthoff, M. Richter, and C. Biemann. 2006. Corpus portal for search in monolingual corpora. In *Proceedings of the LREC*.

Lawrence R. Rabiner. 1989. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77 (2), pages 257–286.

G. Salton, A. Wong, and C.S. Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Mark Steyvers and Josh Tenenbaum. 2005. The large scale structure of semantic networks: Statistical analyses and a model of semantic growth. *Cognitive Science*, 29(1):41–78.

Andrew J. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.

D. J. Watts and S. H. Strogatz. 1998. Collective dynamics of ’small-world’ networks. *Nature*, 393:440–442.

F. Witschel and C. Biemann. 2005. Rigorous dimensionality reduction through linguistically motivated feature selection for text categorisation. In *Proceedings of NODALIDA*.

Hans Friedrich Witschel. 2004. *Text, Wörter, Morpheme – Möglichkeiten einer automatischen Terminologie-Extraktion*. Ergon Verlag.

I.H. Witten and E. Frank. 2000. *Weka: Practical Machine Learning: Tools and Techniques with Java Implementations*. Morgan Kaufmann, http://www.cs.waikato.ac.nz/˜ml/weka/.

George Kingsley Zipf. 1949. *Human Behaviour and the Principle of Least-Effort*. Addison-Wesley, Cambridge MA edition.