

# JoBimText Tutorial 2015

## Practice Session

Eugen Ruppert

TU Darmstadt  
Language Technology Group



24.02.2015

# Outline

- 1 JoBimViz – JoBimText models in practice
- 2 Accessing JoBimText models with Java
- 3 Calculating DT with JoBimText
- 4 From DT to JoBimText model

Documentation: <http://jobimtext.org>

## JoBimViz – JoBimText models in practice – Outline

- 1 JoBimViz – JoBimText models in practice
  - JoBimText models – A quick recap
  - JoBimViz
- 2 Accessing JoBimText models with Java
  - IThesaurus Interface
  - WebThesaurus Interface
  - Practice with example project
- 3 Calculating DT with JoBimText
  - Virtual Machine
  - Hadoop File System (HDFS) basics
  - Generating Hadoop script
  - Download and Import of DT
- 4 From DT to JoBimText model
  - Sense clustering
  - ISA Pattern Extraction

## JoBimText models – Distributional Thesaurus (DT)

- DT is a graph with weighted edges
- self-similarities are included
- top N similar words for a given word
- JoBimText models contain DTs for Jos (terms) and Bims (features)

Jo1	Jo2	Similarity score
mouse	mouse	1000
mouse	Mouse	79
mouse	rat	58
mouse	mice	43

## JoBimText models – Jo–Bim scores and counts

- Jo–Bim scores indicate the significance of a Jo–Bim combination,  
based on a significance score (LMI, PMI, LL)
- frequency of the combination is also included
- the Jo–Bim table is usually pruned to remove very frequent (noisy) and infrequent (random) combinations

Jo	Bim	Significance	Count
mouse	Bim_@_Bim(knockout_@_line)	4003.36	247
mouse	Bim_@_Bim(oldfield_@_Thomasomys)	2090.82	129
mouse	Bim_@_Bim(gray_@_lemur)	1475.39	93
mouse	Bim_@_Bim(the_@_cursor)	1321.72	89

## JoBimText models – Jo and Bim counts

- Jo and Bim counts help to find out whether a term or feature occurred in the corpus, and how often it occurred
- unpruned data, shows the actual counts

Jo	Count
mouse	18637
mousetrap	253
mouse's	162
mousehole	85

## JoBimText models – Sense clusters

- Sense clustering performed by Chinese Whispers algorithm  
Biemann (2006)
- sense id and a list of “cluster terms”
- with ISA labels for the cluster terms

---

Jo	Sense ID	Cluster terms
mouse	1	musk, mule, roe, barking, ...
mouse	1	mammalian, murine, Drosophila, human, ...
mouse	2	rat, mice, frog, sloth, rodent, ...
...		
mouse	6	joystick, keyboard, monitor, simulation, ...

---

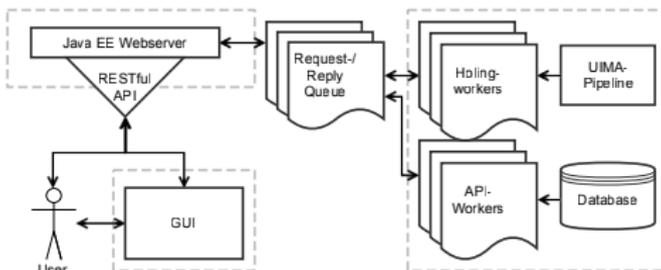
## JoBimViz – Overview

(formerly known as JoBimText web demo)

<http://maggie.lt.informatik.tu-darmstadt.de:10080/jobim/>

- web application
- offers access to JoBimText models
- sentence holing/parsing
- RESTful API with different output formats:  
JSON, TSV, XML

## JoBimViz – Architecture



- RESTful API
  - abstraction of database operations
  - message queue for robustness
  - useful for prototyping
- GUI
  - uses the RESTful API
  - demonstration of data

# JoBimViz Live Demo

JoBimText

Sentence Input field      Start Holing      Holing Operation Selector

Graph    List

Representation Options

RESTful API Selector

Darmstadt<sub>Jo</sub>  
Count: 6344   

Count Display

Holing/Parsing Result

Selectors for Jos and Bims

He drives his jaguar to Darmstadt

Jo

Jo	Score
Darmstadt	1000
Kassel	72
Homburg	52
Cassel	44
Rotenburg	33

Similarity Display

Bims

Bim	Score	Count
Bim @_ Bim(Hesse...	2898.29	201
Bim @_ Bim(SV_@...	2530.20	143
Bim @_ Bim(Akallie...	1692.80	96
Bim @_ Bim(Hesse...	1688.02	109
Bim @_ Bim(Hesse...	1253.45	72

Jo-Bim Scores and Counts

CW    CW-finer

Sense 1: Kassel · Homburg · Cassel · Roten...

Sense Clusters

Visualization by the [Language Technology Group](#) at the TU Darmstadt.  
API description and documentation: [JoBim API description](#)

## JoBimViz – Available models

Name	Data	Holing Operation
wikipediaTrigram	Wikipedia EN 2014	Trigram holing
wikipediaStanford	Wikipedia EN 2014	Stanford parsing, dependency holing
trigram	En news 120M	Trigram holing
stanford	En news 120M	Stanford parsing, dependency holing
germanTrigram	De news 70M	Trigram holing
germanParsed	De news 70M	Mate-tools parsing, dependency holing

Identify holing type in URL from a RESTful API request:

[http://maggie.lt.informatik.tu-darmstadt.de:](http://maggie.lt.informatik.tu-darmstadt.de:10080/jobim/ws/api/wikipediaTrigram/jo/similar/Darmstadt)

[10080/jobim/ws/api/wikipediaTrigram/jo/similar/Darmstadt](http://maggie.lt.informatik.tu-darmstadt.de:10080/jobim/ws/api/wikipediaTrigram/jo/similar/Darmstadt)

## Accessing JoBimText models with Java – Outline

- 1 JoBimViz – JoBimText models in practice
  - JoBimText models – A quick recap
  - JoBimViz
- 2 Accessing JoBimText models with Java
  - IThesaurus Interface
  - WebThesaurus Interface
  - Practice with example project
- 3 Calculating DT with JoBimText
  - Virtual Machine
  - Hadoop File System (HDFS) basics
  - Generating Hadoop script
  - Download and Import of DT
- 4 From DT to JoBimText model
  - Sense clustering
  - ISA Pattern Extraction

## IThesaurus Overview

- general Interface to retrieve data from JoBimText models
- methods for JBT model access
- can access multiple data sources:  
Database, DCA, DCAlight, JoBimViz  
→ data sources can be changed without affecting other code  
e.g. developing with JoBimViz as data source, then switching  
to DCA for efficiency

# IThesaurus Methods I

- Counts: `getTermCount(TERM)`, `getContextsCount(CONTEXTS)`
- Similarities:  
`getSimilarTerms(TERM)`, `getSimilarContexts(CONTEXTS)`,  
`getSimilarTermScore(TERM, TERM)`
- Term-Contexts counts and scores:  
`getTermContextsCount(TERM, CONTEXTS)`,  
`getTermContextsScore(TERM, CONTEXTS)`,  
`getTermContextsScores(TERM key)`,  
`getContextsTermScores(CONTEXTS key)`
- Sense clusters and ISAs:  
`getSenses(TERM)`, `getIsas(TERM)`, `getSenseCUIs(TERM)`

Javadoc API:

<http://maggie.lt.informatik.tu-darmstadt.de/jobimtext/doc/org.jobimtext/>

# IThesaurus configuration and access I

- Construction with a configuration file

```
IThesaurusDatastructure<String, String> dt;
```

```
// different data sources possible
```

```
dt = new WebThesaurusDatastructure(  
    "conf_web_wikipedia_trigram.xml");  
dt.connect();
```

```
dt = new DatabaseThesaurusDatastructure(  
    "conf_mysql_wikipedia_trigram.xml");  
dt.connect();
```

# IThesaurus configuration and access II

## Configuration for JoBimViz:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>  
<webThesaurusConfiguration>  
  <protocol>http</protocol>  
  <server>maggie.lt.informatik.tu-darmstadt.de</server>  
  <port>10080</port>  
  <path>/jobim/</path>  
  <dataset>wikipediaTrigram</dataset>  
</webThesaurusConfiguration>
```

## IThesaurus configuration and access III

### Configuration for MySQL database (excerpt):

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<databaseThesaurusConfiguration>
  <tables>
    <tableSimilarTerms>LMI_1000_1200</tableSimilarTerms>
    ...
  </tables>

  <dbUrl>jdbc:mysql://SERVERNAME/wikipedia_trigram?useUnicode=true</dbUrl>
  <dbUser>USER</dbUser>
  <dbPassword>PASSWORD</dbPassword>
  <jdbcString>com.mysql.jdbc.Driver</jdbcString>

  <similarTermsQuery>select word2, count from $tableSimilarTerms
    where word1=? order by count desc </similarTermsQuery>
  ...
</databaseThesaurusConfiguration>
```

## WebThesaurusInterface vs. IThesaurus interface

- WebThesaurusInterface: same methods as the IThesaurus interface
- realized as access to the RESTful API
- additionally offers sentence holing:  
transforms sentence into Jos and Bims

```
WebThesaurusInterface<String, String> dtWeb;  
dtWeb = new WebThesaurusDatastructure(  
    "conf_web_wikipedia_trigram.xml");  
dtWeb.connect();  
  
dtWeb.getSentenceHoling("this is a sentence");
```

## Example Eclipse Project

- example project to access JoBimText models
- demonstration of the available methods
- JoBimViz and MySQL configuration files
- download project from the tutorial page:  
`http://maggie.lt.informatik.tu-darmstadt.de/  
jobimtext/2015/02/jobimtext-tutorial/`
- unpack and import in Eclipse

# IThesaurus – Try it out!

- use the `WebApiStart.java` class as a starting point
- try some tasks, for example:
  - 1 Determine the term with the highest term count in a sentence!
  - 2 Find the top 5 similar words for “Darmstadt”!
  - 3 What are the typical contexts for the term “university”?
- all methods are exemplified in `WebApiExample.java`

## IThesaurus – Try it out! – Results

Results for the Wikipedia Trigram model:

- 1 Term Frequencies for the sentence “this is a sentence”:  
a:261,241, is:242,366, this:202,507, sentence:39,239
- 2 Top 5 similar terms for “Darmstadt”:  
Darmstadt, Kassel, Homburg, Cassel, Rotenburg
- 3 Typical contexts for “university”:  
Bim\_@\_Bim(and\_@\_professor), Bim\_@\_Bim(a\_@\_professor),  
Bim\_@\_Bim(public\_@\_located)

## IThesaurus – complex example: contextualization

- contextualization, identifying the correct sense in context
- `WebApiContextualizationExample.java` in example project
- Example sentence: “The **mouse** button is stuck”
- Idea:
  - perform sentence holing
  - get the Bim from the holing output
  - get all senses for the target term
  - for each term from a sense cluster:
    - retrieve the `TermContextsCount(term, Bim)`
    - add count to results
  - the sense cluster with the highest count is the “correct” sense in this context

## Calculating DT with JoBimText – Outline

- 1 JoBimViz – JoBimText models in practice
  - JoBimText models – A quick recap
  - JoBimViz
- 2 Accessing JoBimText models with Java
  - IThesaurus Interface
  - WebThesaurus Interface
  - Practice with example project
- 3 **Calculating DT with JoBimText**
  - Virtual Machine
  - Hadoop File System (HDFS) basics
  - Generating Hadoop script
  - Download and Import of DT
- 4 From DT to JoBimText model
  - Sense clustering
  - ISA Pattern Extraction

## VirtualBox VM for Hadoop operations

- VM with preinstalled Hadoop & JoBimText
- Download: [http://sourceforge.net/projects/jobimtextgpl.jobimtext.p/files/hadoop-VM/Hadoop-VM.zip/download](http://sourceforge.net/projects/jobimtextgpl/jobimtext.p/files/hadoop-VM/Hadoop-VM.zip/download)
- Requirements:
  - about 4 GB of HD space
  - at least 4GB of memory
  - 64 bit infrastructure
  - installation of VirtualBox:  
<https://www.virtualbox.org/>

## Access to VM

- username: hadoop-user
- password: hadoop (same for root user)
- SSH server configured on port 3022:  
`ssh -p 3022 hadoop-user@127.0.0.1`

## Hadoop FS basics

Task	Command
read directory	<code>hadoop fs -ls</code> <code>hadoop fs -du [-h]</code>
create directory	<code>hadoop fs -mkdir folder</code>
delete directory	<code>hadoop fs -rm -r folder</code>
copy file to HDFS	<code>hadoop fs -put FILE folder</code>
delete file	<code>hadoop fs -rm FILE</code>
read contents of a folder	<code>hadoop fs -text folder/*</code>

<http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html>

## Hadoop FS basics – Practice

- create folder on HDFS  
`hadoop fs -mkdir wikipedia_1M`
- download and unpack wikipedia\_1M dataset:  
`https://sourceforge.net/projects/jobimtext/files/data/dataset/wikipedia\_sample\_1M.gz/download`
- upload file from client directly to HDFS  
`cat wikipedia_1M | ssh -p 3022 hadoop-user@127.0.0.1 "hadoop fs -put - wikipedia_1M/corpus.txt"`
- read text  
`hadoop fs -text wikipedia_1M/*`
- download dataset, e.g. after computation  
`ssh -p 3022 hadoop-user@127.0.0.1 "hadoop fs -text dataset-folder/p*" > dataset.txt`

## Generating Hadoop script

- Python script to generate the Hadoop operations shell script
- running the script:

```
cd jobimtext_pipeline_0.1.2/  
python generateHadoopScript.py
```
- getting help:

```
python generateHadoopScript.py -h
```
- minimal parameters:

```
python generateHadoopScript.py (-hl HOLING | -nh)  
dataset
```

## Generating Hadoop script – Parameters and options I

Holing options:

- hl set holing operation, e.g. “trigram” or “stanford”  
selection of holing operations depends on the on the JoBimText pipeline (ASL or GPL)  
The GPL licensed pipeline contains more holing operations (e.g. Stanford dependency holing)
- nh do not perform holing operation; to be used, when using a custom holing operation, or using pre-holed data
- savecas stores the binary CAS after holing operation, useful for parsing large corpora; ‘parse once, use often’
- lang document language, set in the CAS; required for Stanford parser and some other DkPro components

## Generating Hadoop script – Parameters and options II

JoBimText computation options:

- sig significance measure (LMI, PMI, LL, Freq) for word feature ranking, default: LMI
- sc similarity scoring function of two terms, default: one

Options:

- one adds a constant '1' for each shared feature
- scored adds  $1/|common\ terms\ for\ feature|$  for each feature
- log\_scored adds  $1/\log(|common\ terms\ for\ feature|)$  for each feature

## Generating Hadoop script – Parameters and options III

- af append features to the final DT, default: false  
compiles evidence for similarity score in the DT
- nb no Bim DT is computed, compute only the Jo DT
- fm format, for compatibility with former holing  
functions, v1, v2 or v3

## Generating Hadoop script – Parameters and options IV

Pruning options:

(two values can be specified, e.g. -f 2,3; the first value is used for the Jo DT, the second for the Bim DT)

- f minimal feature count, default: 2
- w minimal word count, default: 2
- wf minimal word-feature count, default: 0
- s minimal significance score, default: 0.0
- p choose the top p ranked features for each term, default: 1000

## Generating Hadoop script – Parameters and options V

- wpmfmax maximal number of words for a feature; features with higher wpmf count are discarded, default: 1000
- wpmfmin minimal number of words for a feature; features with lower wpmf count are discarded, default: 2
- ms minimal similarity, default: 2; removes noisy DT entries
- l maximal number of similar terms for a term, default: 200



## Distributional Thesaurus – Result data

Which folders to download after computation?

word count dataset...WordCount

feature count dataset...FeatureCount

unpruned term-feature scores and counts dataset...FreqSigLMI

Jo DT pruned term-feature scores and counts

dataset...FreqSigLMI...PruneContext\_SETTINGS

similarity graph

dataset...FreqSigLMI...PruneContext\_SETTINGS  
...SimCount\_SETTINGS\_SimSortlimit

Bim DT pruned term-feature scores and counts

dataset...FreqSigLMI...PruneContext\_BIM\_SETTINGS

similarity graph

dataset...FreqSigLMI...PruneContext\_BIM\_SETTINGS  
...SimCount\_SETTINGS\_SimSortlimit

## Distributional Thesaurus – DB Import

- use the createTables.py to generate MySQL commands for table creation and data import
- `python createTables.py dataset p sig_measure simsort_limit [path]`
  - `dataset` name of the dataset, e.g. wikipedia\_trigram
  - `p` number of features per word, e.g. 1000
  - `sig_measure` significance measure, e.g. LMI
  - `simSORT_limit` number of DT entries, e.g. 200
  - `path` optional: absolute path to the dataset folder; when used, import commands will be printed
- Example: `python createTables.py wikipedia_trigram 1000 LMI 200 /path`

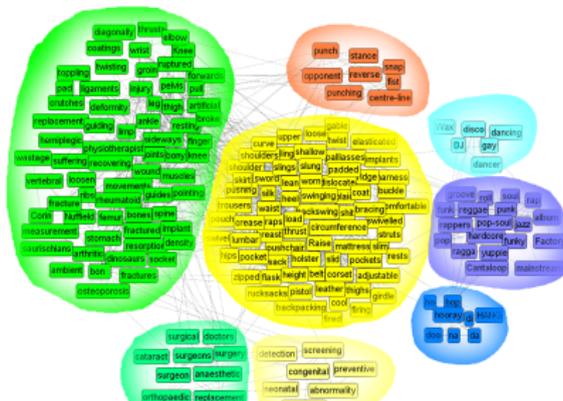
## From DT to JoBimText model – Outline

- 1 JoBimViz – JoBimText models in practice
  - JoBimText models – A quick recap
  - JoBimViz
- 2 Accessing JoBimText models with Java
  - IThesaurus Interface
  - WebThesaurus Interface
  - Practice with example project
- 3 Calculating DT with JoBimText
  - Virtual Machine
  - Hadoop File System (HDFS) basics
  - Generating Hadoop script
  - Download and Import of DT
- 4 From DT to JoBimText model
  - Sense clustering
  - ISA Pattern Extraction
  - Sense Labeling

## Sense clustering - Overview

- Chinese Whispers, Biemann (2006)
- unsupervised graph clustering algorithm
- Word Sense Induction (word sense clustering)
- Documentation:

<http://maggie.lt.informatik.tu-darmstadt.de/jobimtext/components/chinese-whispers/>



## Sense clustering – input/output data

- input: DT file
  - format: (word1, word2, similarity)
  - example DT: `https://sourceforge.net/p/jobimtext/code/HEAD/tree/trunk/org.jobimtext.examples.oss/similarity/SimSortlimit`
- output:
  - format: (word, sense\_id, list-of-sense-terms)
  - example sense clustering:

mouse	0	cat,dog,rat
mouse	1	keyboard,joystick

## Sense clustering – Options and arguments

- required arguments:
  - i input file
  - o output file
- optional arguments
  - a weighting algorithm: 1=constant, lin=linear or log=logarithmic (default: 1)
  - N number of top DT entries to consider (default: MAX)
  - n number of top edges to consider within entry (default: MAX)
  - ms minimal similarity (default: 1)
  - mr maximal cluster rank (default: MAX)
  - mc minimal cluster size (default: 1)

```
Execution: java -cp lib/org.jobimtext-0.1.2.jar:lib/* org.jobimtext.sense.ComputeSenseClusters -a  
1 -N 200 -n 100 -mc 3 -ms 5 -mr 100 -i DT_FILE -o OUTPUT_FILE
```

## ISA Pattern Extraction

PattaMaika

[http://maggie.lt.informatik.tu-darmstadt.de/  
jobimtext/components/pattamaika/](http://maggie.lt.informatik.tu-darmstadt.de/jobimtext/components/pattamaika/)

- UIMA pipeline (OpenNLP components)
- UIMA RUTA for pattern identification
- Hearst Patterns in RUTA, Hearst (1992) and Klaussner and Zhekova (2011):

```
(_NP (COMMA _NP)* ("and" | "or") "other" _NP{->TEMP})  
{-PARTOF(PATTERN)-> CREATE(PATTERN, "x"=TEMP)};
```

- Matches “She likes *cats*, *dogs* and other *animals*”

## Running PattaMaika

- Hadoop shell script creation:  
`python generatePattamaikaHadoopScript.py dataset  
[-q queue-name]`
- execution by running the shell script
- detailed instructions:  
`http://maggie.lt.informatik.tu-darmstadt.de/jobimtext/  
documentation/pattern-extraction-with-pattamaika/`

## PattaMaika – Results

- wikipedia\_1M dataset: ca. 200,000 patterns, 6,000 with frequency  $> 1$
- most frequent ISA patterns:

Pattern	Frequency
English ISA language	27
English ISA languages	26
China ISA countries	23
China ISA country	23
Australia ISA countries	22
Australia ISA country	22
Canada ISA countries	21
Canada ISA country	21
United_States ISA countries	19
United_States ISA country	19
India ISA country	18

## Sense Labeling – Overview

- ISA labeling of sense clusters
- Input data: ISA patterns, sense clusters
- Execution:

```
java -cp path/to/org.jobimtext.pattamaika-*.jar  
org.jobimtext.pattamaika.SenseLabeller pattern-file  
sense-cluster-file output-file [minimum_score]
```

- detailed instructions and examples:

```
http://maggie.lt.informatik.tu-darmstadt.de/jobimtext/  
documentation/sense-labelling/
```

## Sense Labeling – Input and output data

- patterns:
  - mouse ISA animal 15
  - cat ISA animal 10
  - dog ISA animal 20
  - dog ISA pet 5
- sense cluster:
  - mouse 0 cat,dog,rat
  - mouse 1 keyboard,joystick
- result:
  - mouse 0 cat,dog,rat animal:60, pet:5
  - mouse 1 keyboard,joystick product:20, input\_device:2

# Thank you!

Thank you for your attention! Good efforts with JoBimText models!

Questions?

## References

- Biemann, C. (2006). 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*, New York, USA.
- Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proc. COLING-1992*, Nantes, France, pp. 539–545.
- Klaussner, C. and D. Zhekova (2011). Lexico-syntactic patterns for automatic ontology building. In *RANLP Student Research Workshop*.