Commercially Empowered
Linked Open Data Ecosystems in Research

D 2.3 - Evaluation and Performance Report

Summary: Within the scope of work package 2, two web services have been developed which allows semantic enrichment and disambiguation of scientific articles as well as mind maps. These services have been named Enrichment Service and Disambiguation Service. In the case of scientific articles, which are typically stored as PDF documents, the documents are first parsed, analysed and extracted. Here the PDFs are split into their structure, which allows us to re-create the original table of contents. Furthermore a set of metadata, for example the author names, are extracted. The algorithms, which have been developed to build the Enrichment Service, have been evaluated against existing or newly created data-set and the results have been published. The task of the Disambiguation Service is to disambiguate entities discovered by the Enrichment Service as well as disambiguate table columns. The results of this work have been published.

Project Acronym	CODE
Grant Agreement number	296150
Project Title	Commercially Empowered Linked Open Data Ecosystems in Research
Date	2014-04-30
Nature	R (Report)
Dissemination level	PU (Public)
WP Lead Partner	Know-Center GmbH
Revision	final
Authors	Roman Kern, Stefan Zwicklbauer

Consortium:

SEVENTH FRAMEWORK PROGRAMME

Project Officer & Project Coordinators

Project Officer	Stefano Bertolo	European Commission DG CONNECT Unit G3 EUROFORUM 01/293 - 10 rue Robert Stumper, L-2557 Luxembourg stefano.bertolo@ec.europa.eu
Project Coordinator	Stefanie Lindstaedt	Know-Center GmbH Inffeldgasse 13, 8010 Graz, Austria +43 316 873-30800 (phone) +43 316 873-1030800 (fax) slind@know-center.at
Scientific Coordinator	Michael Granitzer	University of Passau Innstrasse 33a, D-94032 Passau +49(0)851-509-3305 michael.granitzer@uni-passau.de

Document Revision History

Revision	Date	Author	Organization	Description
1st Draft	2014-04-20	Roman Kern	Know-Center GmbH	Initial version describing the Enrichment Service and the introduction
2nd Draft	2014-04-22	Stefan Zwicklbauer	University of Passau	Description of the Disambiguation Service
3rd Draft	2014-04-29	Stefan Klampfl	Know-Center GmbH	Proofreading and general feedback
4th Draft	2014-04-30	Roman Kern	Know-Center GmbH	Integrated feedback
Final Version	2014-05-06	Roman Kern	Know-Center GmbH	Added conclusions

Contents

1 Introduction

This document summarises the evaluation activities conducted to assess the efficiency and effectiveness of the semantic enrichment and integration components. The two main components are the so called **Enrichment Service** and the **Disambiguation Service**. These are service components, which do not directly provide a user visible interface, but are used by the two partner systems, Mendeley and MindMeister, as well as by the other components of the CODE infrastructure.

The Enrichment Service addresses two main technological challenges: i) the **parsing and processing of PDF files** in order to extract structure (e.g. tables) and meta-data, and ii) the **extraction of named entities and relations** between them. In both cases the final output consists of factual information, where in the case of tables the facts can be directly inferred from the structure, and in the case of relationship between entities, the facts are stated textual information. In order to demonstrate the functionality of the Enrichment Service a demo application and a showcase Web application have been developed, which also allow to assess the quality and performance of the achieved results.

The challenges of the Disambiguation Service are two-fold as well: i) the **disambiguation of entities** (e.g. named entities found by the Enrichment Service), and ii) the **disambiguation of columns** found in tables. The combined output of the Disambiguation and the Enrichment Service are used in multiple different scenarios within the CODE project.

In terms of domains we focused on two selected fields. At first the **biomedical domain** has been chosen, in particular due to the availability of existing resources. Examples for this are the CALBC, GENIA and BioNLP initiatives. As the second domain, the **field of computer science** have been chosen. This has been motivated by the fact that here the resources are scarce, while many of the people involved in the project are directly associated with this domain.

2 Enrichment Service

This section gives an overview of the evaluation conducted in regard to the algorithms implemented within the Enrichment Service. Most of the algorithms have been disseminated as scientific publications, hence most of the reported performance numbers are taken from the respective evaluation section of these studies. These publications are grouped into papers which serve as a technological bases and papers that have been published out of the CODE project. The algorithmic base of the Enrichment Service is found in these papers:

- **TeamBeam Meta-Data Extraction from Scientific Literature** [KJHG12] - This paper describes and algorithmic approach for extraction of core meta-data from PDFs, that contain scientific articles and has been awarded with the best paper award. The proposed method serves as foundation of the PDF extraction algorithms. The achieved performance as reported by the paper has been improved during the course of the project.

The second group of papers report the progress within the project, have undergone a peer-review process and are published at various venues. They all carry the corresponding acknowledgement.

- **An Unsupervised Machine Learning Approach to Body Text and Table of Contents Extraction from Digital Scientific Articles** [KK13b] - This paper describes the extraction of structural elements out of PDF articles and has been awarded as the best paper of the TPDL 2013 conference. The focus on the conducted work has been to develop methods, that do not rely on training data, which usually is associated with two disadvantages. First, training data needs to be created, which in many cases is a tedious endeavour. Second, training data is typically bound to a specific domain or sub-field, which requires training models for each of these domains.

- **Extraction of References Using Layout and Formatting Information from Scientific Articles** [KK13a] - This paper focuses on the extraction of references out of scientific articles. The followed approach has been to take the current state-of-the-art in the field, analyse its properties and to reimplement the algorithm. Finally the algorithm has been enhanced by integrating more features, raising the overall performance.

- **Unsupervised document structure analysis of digital scientific articles** [KK14] - This journal paper summarises previous work and additionally describes previously unpublished algorithms. The newly introduced algorithms focus on the identification of the layout structure within scientific articles. A combination of supervised and unsupervised approach is presented, using the respective advantages for different tasks.

The last group of papers have been assembled within the project, and are currently under submission.

- **Towards a Marketplace for the Scientific Community: Accessing Knowledge from the Computer Science Domain** - While in the biomedical domain many resources and training data have been developed, such resources are lacking in the domain of Computer Science. Therefore much effort has been invested within the project to rectify the situation. This paper summarises the

main results of this effort, consisting of an ontology for the Computer Science domain and its application. Therefore a number of scientific articles has been manually annotated. Thereby the work has been steered into a direction, studying the suitability of such work in order to arrive at valuable marketplace.

- **A Comparison of two Unsupervised Table Recognition Methods from Digital Scientific Articles** - Out of the task associated with the extraction of structure from scientific articles, one of these tasks is of particular interest within the project. This is the task of identification and extraction of tables, as tables are expected to be an important source for factual information. This paper describes two approaches of unsupervised table extraction methods and compares these with the current state-of-the-art and existing commercial solutions.

2.1 Applications

The Enrichment Service itself exposes its functionality via a **REST-like API**, but does not provide an end-user visible interface. Therefore in the course of the project a number of applications have been developed to demonstrate and showcase the underlying technologies.

2.1.1 Demo Application

To demonstrate the achievements after one project year, a **demonstration application** has been developed on top of the Enrichment Service. This web application should should give at a glance overview of some of the developed functionality. It does not provide rich interaction with the user. The demonstration application allows to upload a PDF, which is expected to contain a scientific article. While the application is well suited for friendly users with expert knowledge, it does not allow to track success factors.

2.1.2 CODE Annotator Tool

During the second project year another web application has been developed to showcase the underlying functionality, the **CODE annotator tool**. The development has been steered to allow a rich interaction with the user, simulating the scenario of users who wishes to extract and model specific knowledge. To accomplish this task, a tight integration with other tools from the project partners has been one of the goals. The typical expected work flow can be summarised as follows:

1. The user collects and manages scientific articles of interest in her **Mendeley** library, which then can be imported into the CODE annotator tool, typically as two different data-sets - one for the development and another one for the validation.

2. The user creates a model as **Mindmeister** mind map, which represents the main items of interest, in particular the entities (e.g. concepts) and relations between them. The mind map can then be imported into the CODE annotator tool as well.

3. The **search and annotation functionality** of the CODE annotator tool is used in order to manually specify the text passages within the scientific articles, which contain factual statements. Depending on the nature of the concepts, different approaches might be taken.

4. Given the manually annotated development data-set, **machine learning** algorithms may be applied in order to automatise the annotation process. Here the user is given an array of tools on how features crucial for the learning algorithms are shaped. During this process the performance of the learning can be evaluated using the validation data-set.

5. Finally, the user may choose to **share the model**. The model with the learnt representation of the domain can then be used by other tools. For example, the model can be fed to the Enrichment Service to provide user specific annotations.

Besides giving the user a platform to create user and domain specific models, the CODE annotator tool also provides mechanism to track the performance of the algorithms and to assess some of the key performance indicators of the project. These will be reported in the following sections.

2.2 Accessing Knowledge from the Computer Science Domain

Unlike the biomedical domain, in the field of computer science there is a lack of readily available corpora, where knowledge items are contained in a structured form. Therefore the task has been tackled to create such a resources, including the sub-tasks of modelling the knowledge, assessing the scope, integrating the state-of-the-art, develop the missing technological methods and disseminate the results.

2.2.1 Ontological Description

The first step is the knowledge modelling task, where entities and their relations are to be conceptualised. Figure 1 gives an overview of the ontological description of the computer science domain. For example, there are algorithms, corpora and performance measures. Given these concepts and their relations allows the identification and extraction of factual statements, e.g. what performance has been achieved by various algorithms on specific data-sets. Another example use case is the automatic extraction of funding information out of PDFs, which allows the aggregation of domain level statistics.

2.2.2 Manual Annotation

Given this ontological descriptions containing concepts and their relations, a number of scientific articles from the domain of computer science were manually annotated. The final annotated data-set contains more than 5,000 manually curated concepts and relations. Table 1 gives an detailed overview of the manually annotated concepts.

2.2.3 Semi-Automatic Performance

The first step starting from the manual annotations towards fully automated annotations is a heuristics driven approach. Here information extraction techniques are devised based on manually crafted rules, pre-assembled lists and trigger phrases. These methods should then serve as source of evidence for the complexity of the task and to provide a baseline figure. We focused on five difference concepts:

Algorithms The first concept are the named entities representing different algorithms. Here we followed a gazetteer based approach. We crawled Wikipedia to arrive at 1193 different names for algorithms, which were manually filtered to contain names not too specific or too general.

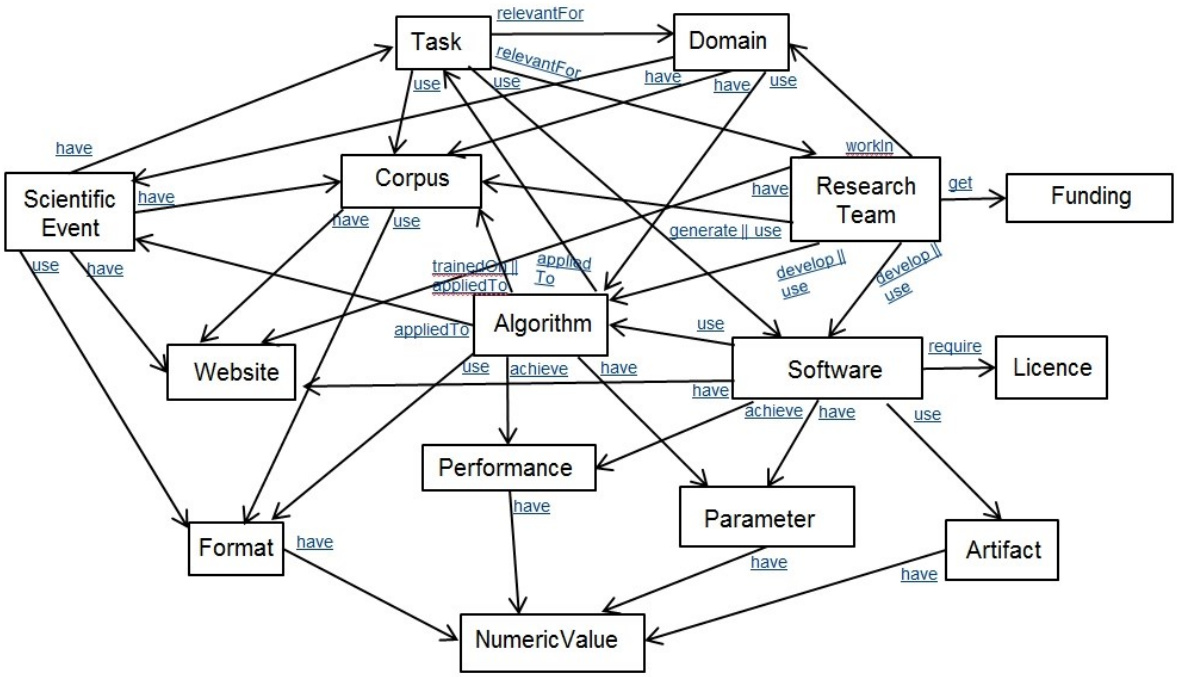

Figure 1: Ontological description of the computer science domain as modelled by the domain experts

Corpus The second concept are the names of corpora, for example data-sets or dictionaries. Again Wikipedia has been used as a source to populate the gazetteer list. The final number of entries in this white list is 105.

Performance The next concept are names of performance measures, e.g. precision or recall. The number of performance measures is a little less than for the other types of concepts, totalling at 85 contained in the respective gazetteer list.

Numeric Value The concept of numeric values goes beyond simple numbers, as we are in particular interested in the number as well as the corresponding unit. Furthermore the authors of scientific articles find innovative ways in how to write a numeric value, making the even task of extracting numbers a challenge.

Funding Information The final concept differs from the other types of concepts as it is represented as phrases of multiple words within the scientific articles. Here a simple gazetteer based approach is likely to miss many occurrences of this concept. Therefore we devised an method, which makes use of trigger phrases and noun phrases.

In table 2 the results of the semi-automatic annotation are summarised. As ground truth we used our manually annotated data-set. While the performance in terms of precision is satisfactory, the recall values are lower than the precision, with the exception of numeric values. These results highlight the complexity of the task. In particular the open classes like algorithm and corpus turned out to be a tough challenge for information extraction techniques.

Table 1: Ontological description of the computer science domain as modelled by the domain experts

Concept	#Instances	#Unique Instances
Domain	86	35
Corpus	187	51
Scientific Event	35	21
Algorithm	734	200
Numeric Value	991	593
Parameter	102	39
Task	522	200
Software	944	192
Performance	800	187
Website	87	29
Funding Information	37	30
Research Team	26	21
Artifact	93	38
Format	125	32
License	4	3

2.2.4 Fully Automatic Performance

In course of the semi-automating annotation of scientific articles from the computer science domain we also studied the effectiveness of fully automatic approaches. In the field of information extraction, algorithms like Maximum Entropy (ME) and Conditional Random Fields (CRF) are considered to be state-of-the-art. We therefore integrated these approaches into the Enrichment Service. We report the initial results of a CRF based classifier, which has been evaluated on the same data-set used for training. Therefore the reported numbers in table 3 cannot be considered as being representative for the performance in the real world. Instead, the figures represent the best case and thus can be considered as being the upper baseline given the size of such a training data-set and given the particular machine learning approach.

2.3 Accessing Knowledge from the General Domain

In the field of Natural Language Processing the task of Information Extraction has received a lot of attention. In particular a series of initiatives has been conducted to assess and improve the state-of-the-art in this domain. Here the CoNLL, the Conference on Natural Language Learning, has been organising so called shared tasks in this field. This conference series started in the mid 90ties of the past century and carried out named entity extraction tasks until 2003. In 2003 the last shared task on Language-Independent Named Entity Recognition has been conducted[1]. In this shared task they defined four

[1] http://www.clips.ua.ac.be/conll2003/ner/

Table 2: Results of the semi-automatic annotation for the five selected concepts, highlighting the difference between open and closed class entities.

Concept	IE Approach	Precision	Recall	F_1 - Score
Algorithm	Gazetteer	0.68	0.34	0.45
Corpus	Gazetteer	0.94	0.50	0.65
Performance	Gazetteer	0.80	0.49	0.61
Numeric Value	Pattern	0.76	0.87	0.81
Funding Information	Trigger Phrases	0.79	0.70	0.74

different classes of named entities: i) person names, ii) organisation names, iii) location names and iv) a miscellaneous category.

In table 4 the results from the 2003 shared task are summarised for the English sub-task. The participants have been free to use external corpora or additional tools. For example, the participants who have achieved the highest F_1 score utilised a combination of different machine learning algorithms, including maximum entropy. Furthermore they integrated the output of another family of named entity recognition tools into their feature set. These tools have been trained on data-sets different than the one supplied by the CoNLL organisers.

We replicated the best performing system from the CoNLL shared task, but we omitted some of the aspects. In particular, i) we did not use pre-existing named entity recognition components, ii) we did not combine the output of multiple algorithms and iii) we did not use any assembled gazetteer lists. The idea to combine as many components as possible appears to be a sound approach to achieve the highest possible performance figure for a shared task, but in a more practical scenario such an approach is less suited due to its excessive run-time. Instead we replicated much of the feature engineering of their system and replaced their family of algorithms with a single one. The purpose of this evaluation has therefore not been to achieve a optimised result for this specific data-set, but to have a reference to compare against.

In table 5 the performance of our named entity recognition pipeline on the CoNLL data-set 'testa' is shown. This data-set contains examples close to the training data-set and hence the performance is very good, especially given the knowledge lean approach, which does not make use of any pre-assembled lists of person or organisation names. Figure 2 shows how these numbers have been collected. The CODE annotator tool provides the functionality to train on one data-set, here the CoNLL training data-set, and to test the performance of the machine learning model on another data-set, here the 'testa' data-set.

In addition to the 'testa' data-set, there is also the CoNLL test data-set labelled 'testb'. This corpus contain examples less similar to the original training data set, therefore one can observe a drop in the performance numbers. This has also been the case for the participants of the original CoNLL shared task. The results for the Enrichment Service are summarised in table 6. In particular the named entity for the classes organisation and miscellaneous appear to be affected more than the other two classes.

The performance of named entity recognition pipeline of the Enrichment Service on the general

Table 3: Results from the CRF base classifier on the same data-set, which has been used for training. Therefore the presented numbers are not indicative for the performance of the classifier for unseen data.

Concept	**Precision**	**Recall**	**F_1**
Algorithm	0.81	0.81	0.81
Artifact	0.85	0.68	0.76
Corpus	0.9	0.81	0.85
Domain	0.81	0.55	0.65
Format	0.8	0.8	0.8
Funding Information	0.75	0.61	0.67
License	1	0.5	0.67
Numeric Value	0.9	0.84	0.87
Parameter	0.63	0.19	0.29
Performance	0.85	0.67	0.75
Research Team	0.89	0.62	0.73
Scientific Event	0.91	0.6	0.72
Software	0.87	0.86	0.86
Task	0.74	0.76	0.75
Website	0.82	0.83	0.83

Table 4: Results of the 2003 CoNLL shared task on named entity recognition for the English sub-task.

Team	**Precision**	**Recall**	**F_1**
FIJZ03	88.99	88.54	88.76
CN03	88.12	88.51	88.31
KSNM03	85.93	86.21	86.07
ZJ03	86.13	84.88	85.50
CMP03b	84.05	85.96	85.00
CC03	84.29	85.50	84.89
MMP03	84.45	84.90	84.67
CMP03a	85.81	82.84	84.30
ML03	84.52	83.55	84.04
BON03	84.68	83.18	83.92
MLP03	80.87	84.21	82.50
WNC03	82.02	81.39	81.70
WP03	81.60	78.05	79.78
HV03	76.33	80.17	78.20
DD03	75.84	78.13	76.97
Ham03	69.09	53.26	60.15
baseline	71.91	50.90	59.61

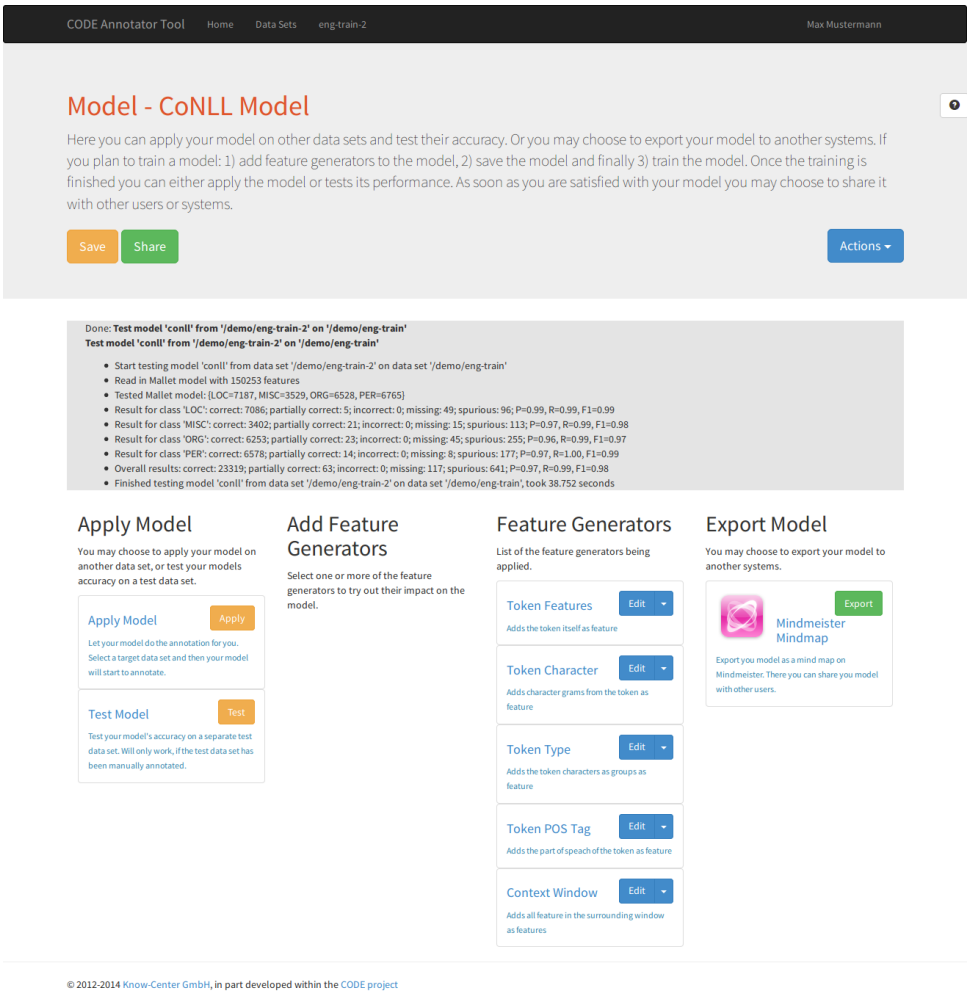

Figure 2: Screenshot of the CODE annotator tool, displaying the result of the evaluation run on the testa data-set of CoNLL.

domain has not been in the focus of the project. Therefore the achieved performance just serves as a reference point and as a demonstration of how the algorithmic will perform on domains other than the two selected for the project (the computer science and bio-medical domain).

2.4 Accessing Knowledge from the Bio-Medical Domain

We originally planned to conduct our evaluation experiments using the CALBC corpus. The feedback we received from our partners indicated that this data-set, in particular the contained concepts, might not be of interest to the expected target audience. For the example, the information that 'patient' or 'mice' are annotated as class 'living being' appeared to be too obvious and of little use for researchers investigating scientific articles.

Therefore we expanded our scope and searched for alternative data-sets, which are more in line

Table 5: Results of the Enrichment Service for the CoNLL shared task for sub-set labelled ``testa''.

Category	Precision	Recall	F_1
Person	0.90	0.92	0.91
Organisation	0.85	0.85	0.85
Location	0.91	0.93	0.92
Miscellaneous	0.91	0.84	0.88

Table 6: Results of the Enrichment Service for the CoNLL shared task for sub-set labelled ``testb''.

Category	Precision	Recall	F_1
Person	0.84	0.91	0.87
Organisation	0.80	0.78	0.79
Location	0.85	0.88	0.87
Miscellaneous	0.80	0.77	0.79

with the expectations. We decided to continue our evaluation using two difference corpora from the field of bio-medical research, namely i) the GENIA[2] corpus and ii) the BioNLP 2011[3] corpus.

2.4.1 Bio-Medical Named Entities

We selected the GENIA data-set as a base for the named entity recognition for the bio-medical domain. The corpus has been used in the 2004 bio-entity recognition task at BioNLP/NLPBA. In particular, the version 3.02 of the GENIA corpus has been used for this task. The training data-set of this corpus consists of 2,000 abstracts from MEDLINE, totalling at more than 20,000 sentences. The test set consists of 404 abstracts, which are similar to the test data-sets. The results of the participants is summarised in table 7.

Especially for entities from the bio-medical domain there are many specialised dictionaries and custom tailored feature engineering approaches [LG08]. In order to improve the performance even further, the preprocessing pipeline has been adapted for the bio-medical domain, e.g. by customising the part-of-speach tagger [TTK+05]. For the Enrichment Service we decided against optimising the algorithms for specific data-sets like GENIA. Instead we tried to allow to make the improvements of our information extraction pipeline applicable to other domains, as already highlighted in the previous sections. The results of the evaluation is presented in table 8.

[2] http://www.nactem.ac.uk/tsujii/GENIA/ERtask/report.html
[3] http://2011.bionlp-st.org/

Table 7: Results from the BioNLP/NLPBA shared task on entity recognition from the bio-medical domain.

Submission	**Precision**	**Recall**	**F_1**
Zho04	76.0	69.4	72.6
Fin04	71.6	68.6	70.1
Set04	70.3	69.3	69.8
Son04	67.8	64.8	66.3
Zha04	69.1	61.0	64.8
Roes04	67.4	61.0	64.0
Par04	66.5	59.8	63.0
Lee04	50.8	47.6	49.1
Baseline	52.6	43.6	47.7

Table 8: Detailed results of the CODE annotations for the GENIA data-set for the five classes, surpassing the results from the original shared task participants.

Class	**Precision**	**Recall**	**F_1**
DNA	0.72	0.71	0.71
RNA	0.67	0.76	0.71
Cell Line	0.56	0.65	0.60
Cell Type	0.84	0.69	0.75
Protein	0.73	0.82	0.77
Overall	0.74	0.77	0.75

Table 9: Results from the entity relation sub-task from the BioNLP initiative, combining the results for the two relation types (protein-component, subunit-complex).

Team	Precision	Recall	F_1
University of Turku	68	50.1	57.7
VIB - Ghent University	37	47.5	41.6
Concordia University	46.9	24.4	32
University Of Science, VNU, HCMC	23.3	15.7	18.7

Table 10: Results from the relation extraction of the CODE approach, assembled using the CODE annotator tool.

Relation Type	Precision	Recall	F_1
Protein-Component	0.68	0.76	0.72
Subunit-Complex	0.67	0.42	0.52

2.4.2 Bio-Medical Relations

Factual information is not only found in the occurrences of named entities, but in particular in the relationship between entities. Again, this area has been invested by the community of NLP research on bio-medical text. In order to develop and test our methods we re-used an already existing data-set. Here we have chosen the supporting task on entity relations from the 2011 BioNLP initiative[4]: ``The task concerns the detection of relations stated to hold between a gene or gene product and a related entity such as a protein domain or protein complex.''

In table 9 the results from the sub-tasks from four different teams is reported. As indicated by the generally low performance, this task has proven to be a hard problem.

For the extraction of relations we followed a supervised machine learning approach. This time we have selected the Maximum Entropy classification algorithm due to is low run-time requirements. In our initial evaluations we found our results to be in line with the current state-of-the-art in this field and presented in table 10. As our evaluation does not directly follow the strategy from the BioNLP shared task, the number should give an indicator for the performance of our approach, but do not allow a direct comparison. The evaluation has been conducted using the CODE annotator tool, which also allows to visually inspect the results.

[4] http://2011.bionlp-st.org/home/entity-relations

Table 11: Performance of the PDFBox library on three types of granularity: words, lines and blocks.

Granularity	Split	Merged	Not Found	Recognized
Blocks	1799 (21.65%)	1465 (17.63%)	4 (0.05%)	5040 (60.66%)
Lines	665 (1.02%)	1333 (2.04%)	10 (0.02%)	63353 (96.93%)
Words	1506 (0.27%)	1050 (0.19%)	71 (0.01%)	557385 (99.53%)

Table 12: Performance of the CODE PDF extraction pipeline on three types of granularity: words, lines and blocks.

Granularity	Split	Merged	Not Found	Recognized
Blocks	1508 (18.15%)	565 (6.80%)	165 (1.99%)	6070 (73.06%)
Lines	582 (0.89%)	457 (0.70%)	198 (0.30%)	64124 (98.11%)
Words	14650 (2.62%)	1205 (0.22%)	666 (0.12%)	543491 (97.05%)

2.5 Extraction from PDFs

Most of the published scientific articles today are stored in the Portable Document Format (PDF). This file format has been originally engineered to allow a consistent rendering of the document's content on various devices. This target can now be considered as being largely achieved given the ubiquitous nature of PDF documents. Unfortunately, the storage of structured information within PDF file, e.g. the sections and headings, has not been considered a priority. Basically PDF file just contain operations like draw a character with font at a specific location, in many cases even omitting word boundaries. In order to finally extract factual information out of this type of files, one needs to devise an extraction pipeline to extract text, layout and structure. Libraries like Apache PDFBox[5] already provide a technological base for these extraction tasks and even attempt to solve some of these extraction tasks.

2.5.1 Extracting Contiguous Text Blocks

We compare our text extraction algorithm against the text extracting component of the open-source PDFBox library. Here we look at three levels of granularity, namely i) word-level, ii) line-level iii) block-level. In table 11 the results for the PDFBox are reported agaist the GROTOAP data-set [TCR12].

The results from our CODE PDF extraction pipeline for the three different levels of granularity are reported in table 12. Generally the performance from our extraction pipeline can be considered to be an improvement over the baseline. Just for words the agreement between the data-set and the output of PDFBox is higher. Manual inspection revealed that in the ground truth punctuation characters are considered part of the preceding words, which is not in line with the output from our extraction pipeline.

[5] http://pdfbox.apache.org

Table 13: Contingency table of blocks labelled as meta-data evaluated on the GROTOAP dataset. Columns correspond to our labels, rows correspond to GROTOAP labels. Boldface entries denote equivalent labellings.

Type	Title	Journal	Author	Affiliation	Email	Abstract
Abstract	1	0	0	0	0	**195**
Body	0	0	0	0	0	3
Correspondence	0	0	0	0	**31**	0
Unknown	0	0	0	1	0	0
Copyright	0	0	0	0	6	0
Author	0	0	**102**	0	1	0
Title	**109**	0	1	0	0	0
Dates	0	0	0	0	1	0
Bib Info	0	1	0	0	0	0
Affiliation	1	0	0	**96**	1	0
Total	111	1	103	97	40	198

2.5.2 Block Labelling

The GROTOAP data-set, which has already been used for the text extraction, also contains labels for the blocks. The vocabulary for labelling these block differs from the semantics utilised by the CODE PDF extraction pipeline. Therefore we present the results in form of a contigency tables. See table 13 for the meta-data blocks and table 14 for all remaining blocks.

2.5.3 Body Text and Headings Extraction

The next series of evaluation test the performance of the extraction of the main text, which is considered to be the body text of the scientific articles. Additionally to the main text, the extraction of heading information is evaluated as well.

For the evaluation of the body text and heading extraction we used a data-set assembled from 1,000 articles randomly drawn from Pubmed[6], a collection of open-access research papers from the bio-medical domain. Here the baseline is supplied by SectLabel [LNK11], which is available as open-source from the ParsCit software package[7]. We included results from two different PDF parsing libraries as baseline for SectLabel, PDFBox and Poppler[8].

Main text performance is defined in terms of the relative number of insert and delete operations necessary to reproduce the ground truth text. Table 15 reports the performance of our CODE PDF extraction pipeline and the baseline results. Our PDF extraction pipeline makes use for the layout information, as well as formatting information, resulting in an improved performance in comparison to the

[6] http://www.ncbi.nlm.nih.gov/pubmed/
[7] http://wing.comp.nus.edu.sg/parsCit/
[8] http://poppler.freedesktop.org/

Table 14: Contingency table of blocks evaluated on the GROTOAP dataset. Columns correspond to our labels, rows correspond to GROTOAP labels. Boldface entries denote equivalent labellings.

Type	Decoration	Caption	Main	Heading	Table	Sparse	Unlabelled
Abstract	0	0	6	1	0	50	25
Body	6	25	**3707**	**1188**	8	377	278
Keywords	0	0	2	1	0	6	17
Correspondence	0	0	0	0	0	9	4
Figure Caption	0	**437**	0	0	2	106	45
Table Caption	2	**167**	0	0	8	14	5
Equation Label	10	0	0	0	0	183	0
Page Number	**819**	0	0	0	0	26	0
Unknown	3	3	76	66	345	366	377
Table	14	1	2	4	**3700**	1790	46
Copyright	5	0	7	0	0	75	90
Type	1	0	0	0	0	101	1
Author	0	0	0	2	0	4	3
Editor	0	0	0	0	0	18	1
References	2	0	**379**	37	0	375	516
Title	0	0	0	1	0	2	6
Figure	2	1	0	0	7	3706	89
Dates	0	0	0	0	0	7	64
Equation	0	0	2	32	0	491	3
Bib Info	**1158**	0	2	3	0	115	59
Affiliation	0	0	5	4	0	48	44
Total	2022	634	4188	1339	4070	7869	1673

baseline.

Table 15: Performance of the main text extraction compared to the output of ParsCit (SectLabel) measured as macro precision/recall/f_1. ``Raw'' indicates performance without hyphenation resolution.

Approach	Precision	Recall	F_1
CODE Main Text	0.950	0.961	0.945
CODE Main Text (raw)	0.947	0.961	0.944
ParsCit (PDFBox)	0.787	0.960	0.857
ParsCit (Poppler)	0.757	0.925	0.827

In addition to the evaluation of the main text, we conducted an evaluation on the extraction of headings. We used the same data-set as for the main text extraction, as well as using the same software stack as baseline. In table 16 the results of the headings evaluation is summarised. Again, the more sophisticated CODE PDF extraction pipeline delivers superior results.

Table 16: Performance of the headings extraction compared to the output of ParsCit (SectLabel) measured as macro precision/recall/f_1.

Approach	Precision	Recall	F_1
CODE Headings	0.837	0.768	0.779
ParsCit (PDFBox)	0.392	0.269	0.299
ParsCit (Poppler)	0.421	0.259	0.300

2.5.4 Table of Contents Extraction

Given the extracted headings one can reconstruct the table of contents. This is necessary for most of the articles, as they typically do not carry a corresponding meta-data[9]. We evaluated the table of contents extraction performance on the already used Pubmed data-set. This data-set also contains the reference table of contents, which can be transformed into a tree structure. For the evaluation we compare the extracted heading tree against the one taken from the ground truth by computing the tree edit distance [ZS89].

In table 17 the table of contents evaluation results are presented. The baseline is generated using the ParsCit algorithm, once using PDFBox as parsing component (ParsCit I) and poppler (ParsCit II). For the CODE PDF extraction pipeline three numbers are reported, representing three sources of errors: i) errors introduced by the table of content extraction (CODE I), ii) errors introduced via the block categorisation stage (CODE II) and iii) errors introduced by the block extraction stage (CODE III).

[9]This meta-data is just present for about 15% of all scientific articles across multiple domains.

Table 17: Evaluation of the table of contents extraction. Three configurations of the CODE PDF extraction pipeline are compared against two configurations of the baseline algorithm.

Measure	CODE I	CODE II	CODE III	ParsCit I	ParsCit II
Mean tree edit distance	0.25	2.15	5.18	13.59	13.42
Number of articles	308	308	633	633	633

Table 18: Performance of the table region detection on two different data sets, measured as the retrieval of individual characters.

Data Set	Precision	Recall
ICDAR 2013	0.825	0.918
Science Direct	0.842	0.952

2.5.5 Table Extraction

Tables are expected to contain factual information. Therefore tabular structures found in scientific articles were in the focus of our CODE PDF extraction pipeline. For the evaluation we used two different data sets, one from the general domain (ICDAR [GHOO13]) and one from the scientific domain (Science Direct). Good performance values on the former data set indicates a generalisation of our algorithms to the non-scientific domain.

Prior to extracting the actual information contained in the tables it is necessary to identify the region of a table within an article. Tables are primarily identified via their captions, from which the table region is then iteratively expanded using heuristics. Table 18 shows the performance of our table region detection algorithm on both data-sets, in terms of the overlap of characters.

For the actual table extraction we implemented two different approaches, using different clustering techniques. One method is based on clustering words into columns and rows based on their horizontal and vertical overlap. The other approach partitions the table content into rows and columns at selected minima of one-dimensional projection histograms. These minima indicate regions of white space within the table. The performance of the word clustering approach and the partitioning approach is presented in table 19 for the ICDAR data set.

For the Science Direct data set the performance of the two table extraction methods is shown in table 20. In comparison one can observe that the table extraction performance is slightly better for the general domain in contrast to the scientific domain. This can in part be explained by the more complex nature of many tables found in scientific articles.

Finally, we compare the CODE PDF extraction pipeline for table extraction with established, off-the-shelf tools, which are in part commercially available. Figure 3 gives an overview of the different approaches, including the region detection and two different measures for table extraction. From this figure one can infer that our approaches compare well in terms of identifying the relevant table region.

Table 19: Performance of the two table structure detection methods on the ICDAR data set, measured as the retrieval of individual table cells.

Method	Precision	Recall
Clustering	0.764	0.863
Partitioning	0.867	0.871

Table 20: Performance of the two table structure detection methods on the Science Direct data set, measured as the retrieval of individual table cells.

Method	Precision	Recall
Clustering	0.582	0.767
Partitioning	0.791	0.805

For the extraction of the table structure, one of the commercial tools (TET) achieves higher relative performance values, as the final number of extracted tables is just about half of the extracted tables from our approaches (68 vs. 125 tables in total for Science Direct). The other tool (pdfx [CPV13]) generally achieved lower performance values and yielded fewer extracted tables on our data sets (64 vs. 125. tables for Science Direct).

2.6 Runtime Behaviour

The architecture of the Enrichment Service has been deliberately chosen to reach a high degree of scalability. The service itself thereby is just a facade that redirects the individual requests to a separate worker instance. The worker and the service do not run an the same machine and while there is just one service, there are many worker instances. The number of workers thereby is not fixed in advance and thus their number may vary with the demand (we found that 512 workers is sufficiently enough).

The Enrichment Service is typically invoked in an asynchronous way and therefore the response time is not considered to be a critical issue. Still we analysed the different components of the Enrichment Service in order to identify candidates for runtime improvements. Therefore we used a profiler application and observed the runtime behaviour, see figure 4 for the overview of the extraction pipeline. Clearly, the parsing and the extraction of the PDF consumes the most of the time.

We investigating further and looked into the runtime distribution of the CODE PDF extraction pipeline. In figure 5 the individual parts of the PDF extraction are depicted. From this picture one can infer that the parsing itself and the extraction of the layout does not appear to contribute much to the overall run-time (in contrast to the block extraction and reading order detection).

In figure 6 the contribution of the individual detector components of the PDF Extraction pipeline are depicted. Judging from the measurements, the detection of the decoration (e.g. page numbers)

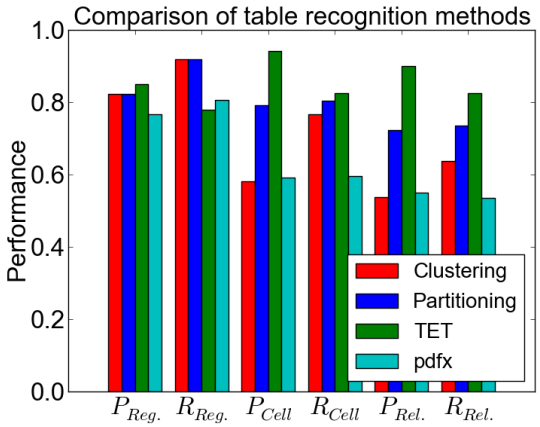

Figure 3: Comparison of the CODE PDF extraction pipeline with off-the-shelf offerings. $P_{Reg.}$ and $R_{Reg.}$ represent the precision/recall of the region detection, P_{Cell} and P_{Cell} compare the table extraction on individual cells, and $P_{Rel.}$ and $R_{Rel.}$ reflect the performance when using an alternative performance measure (used e.g. by the ICDAR shared task).

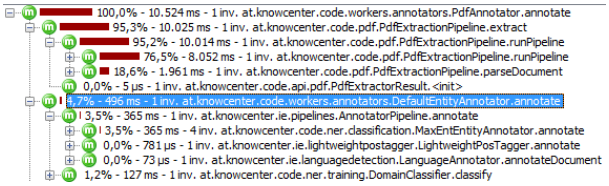

Figure 4: Output of the profile run to analyse the runtime behaviour of the Enrichment Service.

appear to contribute the most to the runtime behaviour.

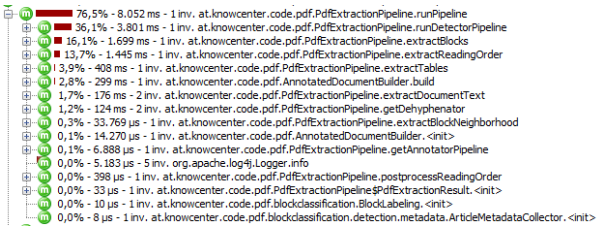

Figure 5: Profile run with the focus on the PDF Extraction pipeline.

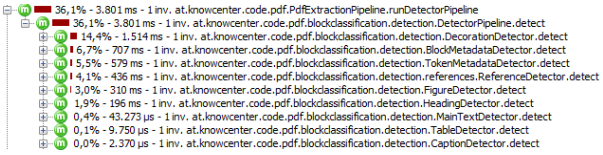

Figure 6: Details of the various detector components of the PDF Extraction pipeline.

3 Disambiguation Service

3.1 Approach

3.1.1 Standard Entity Disambiguation

Ambiguity in a text or name can arise from variations in how an entity may be referenced, from the existence of several entities with the same name or even from spelling mistakes in the name. Our approach assumes a retrieval based approach for disambiguating entities. Given a textual representation of an entity e_i, denoted as t_{e_i}, we return a ranked list R_i of possible entity candidates, i.e.

$$R_i = ranking(Kb, t_{e_i}) \tag{1}$$

Kb denotes the respective knowledge base containing all available entity candidates. In our work we distinguish between two types of knowledge bases depending on the kind of data used for disambiguation, namely an entity-centric knowledge base (e.g. ontologies or Wikipedia) or a document-centric knowledge base (e.g. annotated text corpora). *Entity-centric knowledge bases* describe every entity through a well-defined schema (e.g. database, ontology). Disambiguation algorithms mainly exploit suitable properties like entity names or its appropriate description if it is available. *Document-centric knowledge bases* consist of annotated documents where entities are described by their usage in text (i.e., their usage context). In contrast to entity-centric knowledge bases, there is no explicit schema or ontological description of entities. Instead, documents contain annotations that a particular textual representation t_{e_i} belongs to an entity e_i. This textual representation is often denoted as surface form in related work. When using the document-centric knowledge base, we assume that documents are sufficiently annotated on phrase or word level. Formally, we use the knowledge base definitions proposed in [ZSG13].
Each knowledge base demands its own disambiguation algorithm. We further integrated a LTR algorithm in both approaches and then combined the respective outcomes with a merging algorithm.

Disambiguation with an Entity-Centric Knowledge Base Entries in our entity-centric knowledge base contain standard attributes that are exhibited by all entities (i.e. name, description, link to web resource, type):

$$e_\text{i} = (ID, Name, Description, Link, Type, Occurrences) \tag{2}$$

All referenced surface forms for an entity and the respective amount of occurrences with this surface form are stored in *Occurrences*. Additionally, we store properties of those entities which have occurred in a given context range of the entity in our knowledge base. Table 21 shows an example of a TRNA-Protein stored in our knowledge base.
For disambiguation, we utilize an information retrieval based approach. In the following, we denote the surface form as t_{e_i} and a text of length W surrounding the surface form, namely the context, as $C = \{c_1, \ldots, c_W\}$. An attribute-based ranking function between a given text t and a knowledge base entity is denoted as $sim_x(t, e_i^a)$ with x being the similarity metric. The superscript in e_i^a defines attribute a of entity i. Hence, we compute the score S_{e_i} of an entity by comparing a surface form t_{e_i} with field *Name* (n) of all entities. Further, the surrounding text c is compared with field *Description* (d) of entities

Table 21: Example of an entity-centric knowledge base entry

Field	Content
ID	UNQ9A741
Name	Phenylalanyl-tRNA--protein transferase
Description	Functions in the N-end rule pathway of protein degradation where it conjugates Leu, Phe and, less efficiently, Met from aminoacyl-tRNAs to the N-termini of proteins...
Mainlink	http://www.uniprot.org/uniprot/Q9A741
Type	Caulobacter
Occurrences	aat:::3

in our entity-centric knowledge base:

$$S_{e_i} = sim_x(t_{e_i}, e_i^n) + \sum_{j=0}^{W} sim_x(c_j, e_i^d) \tag{3}$$

Entities whose names do not match with the surface form won't appear in the result list. The disambiguation result finally consists of the Top-N scored entities.

Disambiguation with a Document-Centric Knowledge Base Our document-centric knowledge base contains annotated documents. An annotation may consist of multiple references to entities from different domains or namespaces. We subdivide the content of a document d_i into the document title and the document title combined with the document text in order to increase the recall of the system. Furthermore, all disambiguated surface forms are stored in the field *Keywords*, while the field *Entities* contains all identifiers of referenced entities in a document. Additionally, *ID* depicts a unique document identifier. The structure of our document-centric knowledge base is denoted as follows:

$$d_i = (ID, Title, Titleandtext, Entities, Keyword) \tag{4}$$

Table 22 shows an example of a document stored in the document-centric knowledge base. The disambiguation algorithm is similar to a K-Nearest-Neighbor classification (K-NN) using majority voting. First, we select relevant documents and second, we count their containing entity references. The first task is similar to the approach using an entity-centric knowledge base. We query the T most relevant documents concerning the surface form and surrounding text (similar to equation 3, but querying the fields *Title* and *Titleandtext* instead). Again we require the existence of a similar surface form being present in a document to reduce noise in the retrieved results. The classification task entails counting the appearances of all referenced entities K in our document set T. The parameter T influences the overall results and must be determined empirically. As in K-NN based classification results, high K, which result of a high amount of documents T, are more robust against outliers but are more sensitive to the entity balance while low K's are less robust against outliers and computationally more efficient [Liu11]. Consequently, the result list R consists of the N most appearing entities in K. In contrast to disambiguation with an entity-centric knowledge base, the result quality strongly depends on

Table 22: Example of a document-centric knowledge base entry

Field	Content
ID	174996
Title	Antibody therapy for treatment of multiple myeloma
Abstract	Monoclonal antibody therapy antibody therapy has emerged as a viable treatment option for patients with lymphoma and some leukemias. It is now beginning to be...
TitleAndAbs	Antibody therapy for treatment of multiple myeloma. Monoclonal antibody therapy antibody therapy has emerged as a viable treatment option for patients with...
Keywords	Myeloma::43::50::diso:umls:C0026764:T191:diso

the amount of annotated disambiguated surface forms. For that reason a bootstrapping process with an entity-centric knowledge base and an integration of a document-centric knowledge base afterwards might be worthwhile.

Integration of Learning To Rank We modified both approaches by integrating Learning To Rank (LTR) algorithms. Basically, LTR algorithms learn feature weights to improve the result list in terms of relevant entities appearing on top of the ranking [Joa02]. The score of an entity e_i is a linear combination of all weighted feature values of the respective entity. Feature selection for the entity-based approach is difficult due to storing entities of several biomedical domains. Thus, we focus on general features like different similarity metrics. Table 23 shows an overview of our feature set for the entity-centric approach.

First, we restrict our result list by those entities whose names or synonyms do not match with the surface form. For this purpose, we choose the Jaro-Winkler distance, which is designed and best suited for short strings such as person names, to match surface form and entity names. If the Jaro-Winkler distance falls below a certain threshold, the entity will not appear in the final result set regardless of whether the other features exhibit high values. Furthermore, we distinguish between a Prior and a Sense Prior.

The *Prior* $p(e_i)$ describes the a-priori probability that an entity occurs. As usual in information retrieval, a logarithm is used for this feature to damp high values. A *Sense Prior* $p(e_i|t_{e_i})$ estimates the probability of seeing an entity with a given surface form [MJGSB11], but requires the complete analysis of already annotated documents. The remaining features compute the similarity between the surface form and the entity name as well as the entity description. Additionally, we determine the similarity between the surrounding context and the entity title/entity description. We apply the Vector Space Model with TF-IDF weights as well as the probabilistic model Okapi BM25 (4 + 2 features) for context similarity computation.

In terms of the document-centric knowledge base we integrated the same features as used for the entity-centric approach, except the prior features which are applicable for entities only (see table 23). To learn the weights of our feature set, we choose a ``pairwise'' LTR ([Joa02]) approach, similar to RankSVM [Scu10].

Table 23: Overview of LTR features. Sense Prior and Cooccurrence are only applicable if annotated documents are available.

Nr.	Feature
1	Jaro-Winkler similarity between surface form and entity names (Field *Name* and *Synonyms*)
2	Prior: Occurrences of an entity
3	Sense Prior: Occurrences of an entity with a specific surface form
4	Cooccurrences: Entity context alignment of entity and surface form
5	TF-IDF similarity between surface form and entity names (Field *Name* and *Synonyms*)
6	TF-IDF similarity between surface form and *Desc.*
7	TF-IDF similarity between context and entity names (Field *Name* and *Synonyms*)
8	TF-IDF similarity between context and *Desc.*
9	BM-25 similarity between surface form and *Desc.*
10	BM-25 similarity between context and *Desc.*

Combining both Approaches If we compute the ranked list with an entity-centric and document-centric knowledge base for the same surface form, we can merge the result lists to obtain the most relevant entities of both approaches. We propose two possibilities for merging two lists R_l^{doc} and R_l^{ent} containing entities $e_i^{doc,D1}$ and $e_i^{ent,D1}$, with ent and doc describing the type of a knowledge base and $D1$ describing the subdomain of an entity. The parameter l denotes the length of a list R of the respective approach.

Our method chooses the first $l/2$ entities of both lists and creates a new list by inserting the entities according to their position in the original list. Generally, we prefer the entity of the entity-centric approach over the entity of the document-centric approach if two entities occur at the same position in their lists:

$$R_{doc \cup ent} = \{e_1^{ent,D1}, e_1^{doc,D1}, ..., e_{l/2}^{doc,D1}, e_{l/2}^{ent,D1}\} \quad (5)$$

Finally, we remove potential duplicates in the result list and take the next relevant entities according to the proposed ordering.

3.1.2 Disambiguating Tables

The task of disambiguating (web-)tables can be split up into two major parts:

1. Linking cell contents to an entity in a knowledge base

2. Selection of an appropriate column type

Both tasks require separated algorithms that must be executed in the order given above. In the following we provide a short description on both approaches.

Entity Linking Our entity disambiguation algorithm tries to link table cells to its appropriate entities. For this purpose, we use a modified version of our standard disambiguation service from section 3.1.1:

1. We retrieve a set $E_{c_i} = \{e_1, ..., e_k\}$ of possible entity candidates e_k for each cell c_i in a table column with our disambiguation service.

2. We investigate the types $U = \{Cat_{E_{c_1}} \cup ... \cup Cat_{E_{c_i}}\}$ of all entity candidates. $Cat_{E_{c_i}}$ denotes the set of all types that occur in the entity candidate set of cell c_i. We determine the correct entities for each cell by choosing the entities according to the most occurring type in the table column.

Although this algorithm seems to be quite simple, it attains convincing results if the knowledge base is richly annotated with different entity labels.

Type Inference We integrated two different algorithms to perform type inference on table columns. Majority Vote is a well-known algorithm to provide relevant column types. Our extended algorithm constitutes a Leaning To Rank approach to detect probably relevant types. Further, a simulated annealing algorithm selects the best types across all available columns.

Majority Vote Algorithm The workflow for the majority vote algorithm is as follows:

1. For every column value $c_i \in C$, get all potential URL U_i from the disambiguation service

2. For every candidate URI $u_{j,i} \in U_i$ look up all type relationships (i.e. rdf:type) in the Semantic Web, denoted as $T_{j,i}$

3. Apply a ranking function on all types that returns a ranked list of types, i.e. $rank(C) = rank(\bigcup_{c_i \in C, u_{j,i} \in U_i} T_{*,i}) = t_1, t_2, \ldots, t_o >$.

4. Assign the highest-ranked n-types $T^{selected} = argmax(rank(C))$ with equal score as type to column C and take the highest-ranked URI $u_{j,i}$ that has a type relationship to $T^{selected}$ as URI for c_i.

Advanced Type Inference Algorithm Our advanced inference algorithm constitutes a two-phase approach:
First, we identify relevant types that may represent a table column. Second, we determine the final types across all available table columns. Equation 6 describes the procedures mathematically. We denote a vector of all column type assignments in a table as t and a specific column header as s. Thus t_s denotes the type assignment for column header s respectively. S contains all column headers in the underlying table. $NP_s(t_s)$ constitutes the node potential for a specific type annotation for column header s. A node potential constitutes the linear combination of all weighted feature values. Finally,

Table 24: Overview of LTR features for extended type inference algorithm

Feature	Description
Number of Entities	Calculates the amount of cells having the same type. The amount is divided through the number of cells available within the column.
Column Header	TF-IDF (or BM-25) similarity between the column header and the labels of column types.
Increase of Entities	Determines the increase of cells belonging to this type.
Inverse Document Frequency	Inverse Document Frequency of a type
Layer Variance	Describes the relation of incoming edges in the type graph

$CP(t)$ represents the clique potential for all selected column types. The clique potential sums up all type agreements $r(t_s, t_{s'})$. An agreement between two types t_s and $t_{s'}$ is the amount of matching cells within the compared columns. Two cells match if there is a binary relation between each other. Additionally, the current selected types t_s and $t_{s'}$ must part of the relation schema. For this purpose we use the relation extraction tool patty [NWS12], which provides a huge set of relations between different entities.

$$\underset{t}{argmax}\, Pr(t) = \underset{t}{argmax}\, CP(t) \prod_{s} NP_s(t_s) \quad (6)$$

$$= \underset{t}{argmax} \sum r(t_s, t_{s'}) + \sum_{s \in S} w^\top f_s(t_s) \quad (7)$$

To compute the node potential we create a directed weighted graph representing the type hierarchy of our cells within a table column first. Thus, our algorithm is able to consider types that do not consitute leafs in the type graph. Further, we iterate over all graph nodes (types) and compute a score for each type. Scoring is done by using Learning To Rank with a set of features. Again, we use the ``pairwise'' Learning To Rank approach proposed by Joachims ([Joa02]). Table 24 shows our used feature set for type candidate selection.

The *Number of Entities* feature calculates the amount of cells having the same type (similar to Majority Vote algorithm). Hereby, the sum of the weights w of incoming edges to type t depicts the number if cells containing type t. An incoming edge with weight w from type $t-1$ to type t means that w cells in the column are annotated with type $t-1$, and thus with type t.
The feature *Increase of Entities* computes the increase of cells belonging to this type in contrast to the

preceding type.
Layer Variance investigates the weight variance of the incoming edges to a specific type. Lower variance results in higher feature values.
Column Header and *Inverse Document Frequency* are standard features and need not to be explained in detail.

The final computation of equation 6 is performed with a simulated annealing algorithm. The authors of [KSRC09] have shown that a local search algorithms performs well and leads to satisfying results.

3.2 Evaluation

3.2.1 Entity Disambiguation

Our disambiguation approaches are implemented in Java with all queries being executed with Apache Lucene 4.4[10]. In terms of learning algorithm we chose the Sofia-ml framework[11], a general machine learning framework that gives an efficient and suitable algorithm for massive data sets.

Our experiments comprise the disambiguation with an entity-centric and document-centric knowledge base before and after the integration of the LTR algorithm. The experiments of the entity-centric approach with LTR are divided in an evaluation with annotations and without. In our second experiment we evaluate our LTR approaches after increasing the amount of data and heterogeneity in our knowledge bases.

The CalbC data set provides an abundance of annotated entities that can be used to evaluate and contrast our approaches. We use the CalbCSmall data set for our evaluations, and the CalbCBig for the scalability experiments. When choosing the entity samples for evaluation purposes, we used k-fold cross validation. We selected $k = 5$ and averaged the respective evaluation values to obtain the presented results. We describe our results by a set of comprehensive measures, mean reciprocal rank (MRR), recall and mean average precision (MAP). The reciprocal rank constitutes the most important measurement in our evaluation since we are interested in a user-centric scenario where the user may select from different disambiguation suggestions. Thus, similar to search engines, relevant hits of a query should appear at the top of the result lists [JGP+05]. Unfortunately, measures like discounted cumulative gain (DCG) cannot be applied in our work. These measures are well suited for LTR algorithms, but relevance scores of relevant entities are required, which is not available in CalbC.

Parameters Both disambiguation approaches offer various parameter settings to tweak the overall results. We focus on evaluating generic parameters instead of specific algorithm settings or thresholds to show correlation between the result set and disambiguation parameter settings in general. Table 25 shows on overview of our chosen parameters with their corresponding values. The adjustment *Context Length* affects the number of words in both directions, before and after the corresponding surface form. Our parameter *Query* selects the consideration of the context of the respective surface form to show its significant influence. In this context we investigate the difference between term and fuzzy queries using the surface form (*SF query*) and context (*Context query*). Fuzzy queries match terms with a maximum edit distance of size 2 and regard different spellings and possible typing errors which might occur in all kind of documents. Additionally, the standard information retrieval measures TF-IDF and BM25 are compared (*Similarity*). It must be noted that Lucene's default TF-IDF score also takes internal parameters like term boosting and coordination factor into account, which may influence the result set slightly[12]. The option *Retrieved Results* influences the amount of retrieved results and *Top-T* denotes a major parameter during disambiguation with a document-centric knowledge base only.

With the presence of a detailed parameter notation we are interested in the influence of parameters on the disambiguation results with an entity-centric and document-centric knowledge base whereby we

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

[11] http://code.google.com/p/sofia-ml/

[12] http://lucene.apache.org/core/4_7_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html

Table 25: Settings of evaluation parameters

Parameter	Values
Context length	35, 70, 170, 350
Query	Surface Form, Surface Form & Description
SF query	Fuzzy Query, Term Query
Context query	Fuzzy Query, Term Query
Similarity	TF-IDF, BM25
Retrieved results	5, 10, 20, 50
Top-T (Kb_{doc} only)	20, 50, 100, 200

ignore improving techniques like Learning To Rank. Due to an enormous amount of analyzed parameter combination (256 with an entity-centric and 1024 with an document-centric knowledge base) we refrain from discussing every single result, but emphasize the most important and noteworthy statistics. Table 26 shows an overview of our disambiguation methods with different settings. The column *Settings* consists of the following three printed parameters: *SF Query*, *ContextQuery* and *Similarity*. Due to an unoptimized disambiguation service, poor results are generally noticeable.

When using an entity-centric knowledge base a comparison between TF-IDF and BM25 function indicates an increasing difference between them as soon as the overall results get better. Generally, TF-IDF shows better results in all experiments. The main difference is constituted by the distinction between term and fuzzy query. Combining a term query with the respective surface form results in very poor results considering all measures. Even the usage of stemming algorithms (i.e. Porter-Stemmer [Por80]) does not improve the results. After switching to fuzzy query the situation is different: Attaining better results in all measures means, that entity names mostly vary in their notation. However, changing query properties from term to fuzzy query when querying the context does not affect the result at all.
The usage of an document-centric knowledge base which does not make use of information extracted from the Semantic Web obtains much better results. The amount of used documents to rank the entities (*Top-T*) is constrained to 100 due to decreasing accuracy if we increase the document count. Experiments show an average recall value in the range of 65 and 75 percent and mean average precision values between 50 and 60 percent reveal that a utilization of a document-centric knowledge base is worthwhile. In contrast to the other approach, BM25 similarity does not drop behind TF-IDF similarity. Instead, results do not show any significant differences. The combination term and fuzzy query with a surface form features a difference of 6 to 8 percent in all measures, but this time a term query provides better results. Term queries top fuzzy queries due to similar appearances of terms in the document-centric knowledge base. Again, the difference of term and fuzzy queries when querying the context is not noteworthy and negligible.

After describing the most important settings we compare our approaches before and after integrat-

Table 26: Disambiguation results with an entity-centric and document-centric knowledge base

Settings	**MRR in %**		**Recall in %**		**MAP in %**	
(SF Query/Context Query/Sim.)	Kb_{ent}	Kb_{doc}	Kb_{ent}	Kb_{doc}	Kb_{ent}	Kb_{doc}
Term / Term / TF-IDF	18.4	77.3	12.0	73.9	8.6	58.1
Term / Term / BM-25	17.0	77.8	11.5	74.2	8.1	58.7
Term / Fuzzy / TF-IDF	18.2	77.4	12.1	73.1	8.6	57.8
Term / Fuzzy / BM-25	17.3	77.7	11.8	73.4	8.1	58.3
Fuzzy / Term / TF-IDF	35.4	71.7	31.1	68.9	20.9	53.9
Fuzzy / Term / BM-25	29.2	72.3	28.4	69.1	17.7	54.8
Fuzzy / Fuzzy / TF-IDF	35.4	68.8	31.8	66.4	21.0	50.9
Fuzzy / Fuzzy / BM-25	29.3	69.8	28.9	66.9	17.7	52.2

ing LTR. Table 27 shows an overview of the overall results attained by different algorithm combinations.

The first table section compares the entity-centric with the document-centric approach with and without LTR. During the evaluation of the entity-centric knowledge base we assume that no disambiguation process has been performed so far. Thus, we evaluate this approach without the Sense Prior feature. In the second table section, data from both knowledge bases is available for both approaches. Hence, the entity-centric approach has access to the document annotations in form of the Sense Prior feature. This can be compared by users exhaustively using the entity-centric approach and giving feedback to the LTR system in terms of entity correctness. The results after merging the results of both approaches are listed in this section as well. The column ``Settings'' summarizes algorithm settings. *LTR* and *NoLTR* specify the usage of LTR. The terms *SFTerm* (or *SFFuzzy*) and *CTerm* (or *CFuzzy*) declare that the surface form and surrounding context are combined with a term query or fuzzy query. *FeatureIntegration* refers to the setting where features extracted for LTR are used with weights set to 1 (no LTR applied). The parameter *SensePrior* constitutes the entity-centric approach with LTR additionally using the *Sense Prior* feature. The possibility of merging result lists are denoted as *SimpleMerge*.

Table 27: Accuracy improvements after integration of LTR and user feedback

Settings	**MRR**		**Recall**		**MAP**	
	Kb_{ent}	Kb_{doc}	Kb_{ent}	Kb_{doc}	Kb_{ent}	Kb_{doc}
NoLTR / SFTerm / CTerm / TF-IDF	18.4	77.3	12.0	73.9	8.6	58.1
NoLTR / SFFuzzy / CTerm / TF-IDF	35.4	71.7	31.1	68.9	20.9	53.9
NoLTR / FeatureIntegration	35.3	67.0	34.9	59.6	20.4	48.6
LTR	65.2	78.4	56.2	75.0	45.5	61.1
LTR / SensePrior	87.9	-	78.6	-	71.2	-
LTR / SimpleMerge / SensePrior	88.8		67.4		66.2	

Basically, the results of the entity-centric approach without machine learning algorithms are poor. The main difference when not using LTR is constituted by the distinction between term and fuzzy query. Term queries are responsible for very poor results under 20 percent considering all measures. The

integration of stemming algorithms, i.e. the Porter-Stemmer [Por80], does not influence the results in a positive way. Fuzzy queries fare better due to entity names mostly varying in their notation. The document-centric approach attains suitable results of over 70 percent MRR and nearly 70 percent recall without LTR. Basically, the document-centric approach is more robust against parameter variations.

The use of LTR without Sense Pior significantly boosts the results of the entity-centric approach, with the difference to the document-centric approach falling below 15 percent. The results nearly stay the same for the document-centric approach. This can be explained by the missing relation between the ranking task and the correct entities in the algorithm. If we possess annotated documents and exploit the surface forms of the annotated entities, the additional feature Sense Prior boosts all measures. A MRR of 87.9 percent shows that the first relevant entity almost always appears at the top of the result list. A closer look at the results attained after merging the result lists of both approaches shows a slight increase of the MRR. On the other hand, the recall decreases significantly by nearly 10 percent: The number of relevant non-retrieved entities after the merging process is higher than the gain of new relevant entities. Merging both approaches is only advisable if the application focus is best approximated by using the MRR measure.

In summary, it can be stated that LTR significantly boosts the entity-centric approach. Depending on the amount of annotated entities, the entity-centric approach may significantly outperform the document-centric approach. The usage of machine learning algorithms does not affect the results attained by the document-centric approach. A combination of both approaches provides excellent results in all measures. Thus, a combination would be worthwhile if a comprehensive document-centric knowledge base was constructed, i.e. by using the entity-centric approach in combination with a user-feedback mechanism.

3.2.2 Domain Heterogeneity and Knowledge Base Size

Previous work showed that the size of a knowledge base significantly influences the results when using an entity-centric knowledge base. It transpired that domain heterogeneity also plays a crucial role [ZSG13]. Now, we investigate the results after learning our feature set with LTR. In the following, we talk about an intra-specific domain extension if the knowledge base is extended with entities or documents from the same domain (e.g. combining Umls with Uniprot), as opposed to an inter-specific domain extension (e.g. combining UMLS with DbPedia). In this experiment we enrich our entity-centric knowledge base Kb_{ent}, basically comprising entities of the CalbSmall, with all entities from UMLS, Uniprot or Dbpedia. All documents of the CalbCBig are used for the document-centric knowledge base extension. We do not evaluate an inter-specific domain extension when using the document-centric approach due to non-availability of a suitable document corpus.

Table 28 shows an overview of the results before and after extending the knowledge bases. The column $Change$ expresses the changes of the combined result values. Therefore we average our measures MRR, Recall and MAP. Similar to the results without LTR, the entity-centric approach attains worse results after increasing the amount of entities. Increasing the domain heterogeneity by adding Dbpedia entities significantly worsens the results with a decrease of 30 percent (with Dbpedia only), respectively 42 percent (with Dbpedia, UMLS and Uniprot) in all measures. When comparing the outcomes with those of [ZSG13] our results do not vary considerably. As expected, the results of the document-centric approach stay unchanged. We assume that the document-centric approach is robust against any increase of the document amount.

Table 28: Results after increasing our knowledge base with different corpora

Settings	Integrated Knowledge Bases	MRR in %	Recall in %	MAP in %	#Entities / # Docs	Change in %
$Kb_{ent,\ intra}$	-	65.2	56.2	45.5	265.532	-
$Kb_{ent,\ intra}$	UMLS	56.7	50.4	37.9	2.098.824	-13.1
$Kb_{ent,\ intra}$	UMLS, Uniprot	55.8	49.0	35.7	32.407.960	-15.8
$Kb_{ent,\ inter}$	Dbpedia	46.0	37.9	32.5	4.643.509	-30.2
$Kb_{ent,\ inter}$	UMLS, Uniprot, Dbpedia	39.0	31.7	25.5	36.785.937	-42.2
$Kb_{doc,\ intra}$	-	78.4	75.0	61.1	174.999	-
$Kb_{doc,\ intra}$	CalbCBig	78.0	75.5	59.7	889.282	-0.7

As a main result, our evaluation shows that combining both approaches yields more robust results. LTR instead, is not able to prevent accuracy degeneration after increasing heterogeneity and size of an entity-centric knowledge base.

3.2.3 Table Disambiguation

Data Set To evaluate the performance of our table disambiguation algorithm we chose the data set provided by Limaye et al.[LSC10]. Therefore we use three collected table sets with ground truth annotations.

1. **Wiki-Manual:** 36 tables from Wikipedia article text, based on large content overlap with many web tables. They are annotated with many entities and types.
2. **Web-Manual:** The dataset contains 371 web tables similar to them [GS09]. These were then manually annotated. The main difference between Wiki Manual and Web Manual is that the cell, header, and context texts in the latter are more noisy.
3. **Wiki-Link:** To specifically test the cell entity annotation accuracy at large scale without laborious human judgment, we use tables from Wikipedia text that had more than 90% of their cells linked internally to entities in Wikipedia. This yielded 131 thousand cells with entity annotations spanning 6 thousand tables. No column type or relation annotations were made [LSC10].

All annotated entities are included in the YAGO dataset, the type annotations comprise Wikipedia and Wordnet types. Table 29 displays an overview of the data sets used for evaluation.

Evaluation To provide a competitive evaluation, we measure 0/1 loss, i.e., we lose a point if we get a cell wrong, including na when ground truth was not na [LSC10]. For column type, we report the F1 score (harmonic mean of recall and precision). If ground truth is missing for a entity or type, we drop it from the labeling task. For training model parameters w_1 through w_5, we chose the Wiki-Manual data for learning purpose. Again we used the Sofia-ml machine learning framework.

Table 29: Summary of data sets

Data set	#Tables	Average # rows	Total annotations Entity	Type
Wiki-Manual	36	37	1691	73
Web-Manual	371	35	9239	674
Wiki-Link	6085	20	131807	-

In this evaluation we compare the proposed algorithms, Majority Vote algorithm and Advanced Type Inference Algorithm, against the current state-of-the art algorithms for table disambiguation ([LSC10] and [MFJ13]). Limaye et al. use a graphical approach with a message passing algorithm to determine entities, types and relations in a global process. Mulwad et al. use a similar approach with a semantic message passing algorithm instead. However, it must be mentioned that Mulwad et al. used a different evaluation method to assess the type inference results. They marked all types with one of these labels: *vital*, *okay* or *incorrect*. In our case, there are only the options for *correct* and *incorrect*. Thus, the authors of Mulwad et al. attain much better results than our algorithm or the one by Limaye et al. Table 30 displays the cell disambiguation results and table 31 shows the type inference results, respectively.

Table 30: Overview of cell disambiguation results

Algorithm	Wiki-Manual	Web-Manual	Wiki-Link
Our Algorithm	89.01	86.5	93.7
Limaye et al. [LSC10]	83.92	81.37	84.28
Mulwad et al. [MFJ13]	63.07	67.42	75.89

Table 31: Overview of type inference results

Algorithm	Wiki-Manual	Web-Manual	Wiki-Link
Our Algorithm EXT	58.6	47.9	-
Limaye et al. [LSC10]	56.1	43.2	-
Mulwad et al. [MFJ13]	60.0	57.0	-

The results demonstrate that our entity disambiguation algorithm performs excellent in comparison to the state-of-the-art approaches. 0/1 loss values of 93 percent at the Wiki-Link dataset confirms that our simple but effective approach performs quite well. Our algorithm also clearly outperforms the other approaches on the Wiki-Manual and Web-Manual dataset. This can be explained by our independency of type and relation annotation from cell disambiguation. Both approaches by [LSC10] and [MFJ13] also take possible types and relations for entity disambiguation into account. Another main reason for those high evaluation values is our wealthy enriched knowledge base with entity synonyms and surface forms.

In terms of type inference, we notice a slight increase of accuracy in relation to the Limaye et al. approach. This might result out of better learning to rank features and applied relations. The displayed results belonging to the Mulwad et al. algorithm are not directly comparable with the other approaches due to the reason mentioned above.

4 Conclusions

The enrichment and integration components of the CODE project have been thoroughly been evaluated in order to assess the quality of the results. Thereby we focused on two different domains, the bio-medical domain and the field of computer science. Where applicable we favoured domain independent approaches, e.g. by following unsupervised machine learning approaches. We also conducted experiments to see how well our developed algorithms generalise to other domains.

The evaluation activities are linked with the scientific dissemination of the developed components. Here we managed to publish four papers in the course of the project, with three more being currently being under submission. One of our the papers has been awarded as the best paper of the TPDL conference in 2013.

The main two components are the Enrichment Service and the Disambiguation Service. The main task of the Enrichment Service is take a PDF document as input, parse this document and to analyse its content. Thereby the logical structure as well as the textual content are extraction. This task requires a series of processing steps, ranging from extracting the layout and formatting to the extraction of the table extraction. Here we managed to go beyond the current state-of-the-art in the respective fields and our results compare favourably to existing approaches. For the extraction of named entities, which has received a lot of attention in research, and their relations we are able match the performance of established solutions.

The task of the second main component, the Disambiguation Service is twofold. It first disambiguates the found named entities from the Enrichment Service, and then it disambiguates the table cells and columns of the extracted tables. For the disambiguation of named entities a number of approaches have been explored and their respective performance has been compared. In particular, we also implemented a learn to rank approach and integrated user feedback. We found that a combination of approaches might be the optimal solution to achieve robust results for many scenarios. For the table cell we applied out entity disambiguation algorithm and achieved excellent results when compared to the current state-of-the-art.

References

[CPV13] Alex Constantin, Steve Pettifer, and Andrei Voronkov. PDFX: Fully-automated PDF-to-XML Conversion of Scientific Literature. In *Proceedings of the 13th ACM symposium on Document Engineering*, 2013.

[GHOO13] Max Gobel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. ICDAR 2013 Table Competition. *2013 12th International Conference on Document Analysis and Recognition*, pages 1449--1453, August 2013.

[GS09] Rahul Gupta and Sunita Sarawagi. Answering table augmentation queries from unstructured lists on the web. *Proc. VLDB Endow.*, 2(1):289--300, August 2009.

[JGP+05] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 154--161, New York, NY, USA, 2005. ACM.

[Joa02] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133--142, New York, NY, USA, 2002. ACM.

[KJHG12] Roman Kern, Kris Jack, Maya Hristakeva, and Michael Granitzer. Teambeam meta-data extraction from scientific literature. *D-Lib Magazine*, 18(7):1, 2012.

[KK13a] Roman Kern and Stefan Kampfl. Extraction of references using layout and formatting information from scientific articles. *D-Lib Magazine*, 19(9):2, 2013.

[KK13b] Stefan Klampfl and Roman Kern. An unsupervised machine learning approach to body text and table of contents extraction from digital scientific articles. In *Research and Advanced Technology for Digital Libraries*, pages 144--155. Springer, 2013.

[KK14] Stefan Klampfl and Roman Kern. Unsupervised document structure analysis of digital scientific articles. In *Research and Advanced Technology for Digital Libraries*. Springer, 2014.

[KSRC09] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 457--466, New York, NY, USA, 2009. ACM.

[LG08] Robert Leaman and Graciela Gonzalez. BANNER: an executable survey of advances in biomedical named entity recognition. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 663:652--63, January 2008.

[Liu11] Tie-Yan Liu. *Learning to Rank for Information Retrieval*. Springer, 2011.

[LNK11] Minh-Thang Luong, Thuy Dung Nguyen, and Min-Yen Kan. Logical structure recovery in scholarly articles with rich document features. *International Journal of Digital Library Systems*, 1(4):1--23, January 2011.

[LSC10] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. Annotating and searching web tables using entities, types and relationships. *Proc. VLDB Endow.*, 3(1-2):1338--1347, September 2010.

[MFJ13] Varish Mulwad, Tim Finin, and Anupam Joshi. Semantic message passing for generating linked data from tables. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul T. Groth, Chris Biemann, Josiane Xavier Parreira, Lora Aroyo, Natasha F. Noy, Chris Welty, and Krzysztof Janowicz, editors, *International Semantic Web Conference (1)*, volume 8218 of *Lecture Notes in Computer Science*, pages 363--378. Springer, 2013.

[MJGSB11] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 1--8, New York, NY, USA, 2011. ACM.

[NWS12] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. Patty: A taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1135--1145, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[Por80] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130--137, July 1980.

[Scu10] D. Sculley. Combined regression and ranking. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 979--988, New York, NY, USA, 2010. ACM.

[TCR12] Dominika Tkaczyk, Artur Czeczko, and Krzysztof Rusek. GROTOAP: ground truth for open access publications. *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries*, pages 381--382, 2012.

[TTK+05] Yoshimasa Tsuruoka, Yuka Tateishi, Jin-Dong Kim, Tomoko Ohta, John McNaught, Sophia Ananiadou, and Jun'ichi Tsujii. Developing a robust part-of-speech tagger for biomedical text. In *Advances in informatics*, pages 382--392. Springer, 2005.

[ZS89] Kaizhong Zhang and Dennis Shasha. Simple Fast Algorithms for the Editing Distance between Trees and Related Problems. *SIAM Journal on Computing*, 18(6):1245--1262, December 1989.

[ZSG13] Stefan Zwicklbauer, Christin Seifert, and Michael Granitzer. Do we need entity-centric knowledge bases for entity disambiguation? In *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies*, i-Know '13, pages 4:1--4:8, New York, NY, USA, 2013. ACM.