Piotr Bański, Hanno Biber, Evelyn Breiteneder
Marc Kupietz, Harald Lüngen, Andreas Witt (editors)

Proceedings of the 3rd Workshop on *Challenges in the Management of Large Corpora* (CMLC-3)

Lancaster, 20 July 2015

Corpus Linguistics 2015

CMLC-3

Challenges in the
Management of Large Corpora

Challenges in the Management of Large Corpora (CMLC-3)

Workshop Programme 20 July 2015

Session A (9:30 -10:40)

Introduction

Michal Křen,
Recent Developments in the Czech National Corpus

Dan Tufiş, Verginica Barbu Mititelu, Elena Irimia, Stefan Dumitrescu, Tiberiu Boros, Horia Nicolai Teodorescu,
CoRoLa Starts Blooming – An update on the Reference Corpus of Contemporary Romanian Language

Poster presentations and coffee break (10:40–11:30)

Piotr Bański, Joachim Bingel, Nils Diewald, Elena Frick, Michael Hanl, Marc Kupietz, Eliza Margaretha, Andreas Witt,
KorAP – an open-source corpus-query platform for the analysis of very large multiply annotated corpora

Hanno Biber, Evelyn Breiteneder,
Large Corpora and Big Data. New Challenges for Corpus Linguistics

Sebastian Buschjäger, Lukas Pfahler, Katharina Morik,
Discovering Subtle Word Relations in Large German Corpora

Johannes Graën, Simon Clematide,
Challenges in the Alignment, Management and Exploitation of Large and Richly Annotated Multi-Parallel Corpora

Session B (11:30–13:00)

Stefan Evert, Andrew Hardie,
Zigurrat: A new data model and indexing format for large annotated text corpora

Roland Schäfer,
Processing and querying large web corpora with the COW14 architecture

Jochen Tiepmar,
Release of the MySQL-based implementation of the CTS protocol

Closing Remarks

Editors / Workshop Organizers

Piotr Bański	Institut für Deutsche Sprache, Mannheim
Hanno Biber	Institute for Corpus Linguistics and Text Technology, Vienna
Evelyn Breiteneder	Institute for Corpus Linguistics and Text Technology, Vienna
Marc Kupietz	Institut für Deutsche Sprache, Mannheim
Harald Lüngen	Institut für Deutsche Sprache, Mannheim
Andreas Witt	Institut für Deutsche Sprache, Mannheim and University of Heidelberg

Workshop Programme Committee

Workshop Homepage

http://corpora.ids-mannheim.de/cmlc.html

Table of contents

Author Index

Introduction

Creating extremely large corpora no longer appears to be a challenge. With the constantly growing amount of born-digital text – be it available on the web or only on the servers of publishing companies – and with the rising number of printed texts digitized by public institutions or technological giants such as Google, we may safely expect the upper limits of text collections to keep increasing for years to come. Although some of this was true already 20 years ago, we have a strong impression that the challenge has now shifted to the effective and efficient processing of the large amounts of primary data and much larger amounts of annotation data.

On the one hand, the new challenges require research into language-modelling methods and new corpus-linguistic methodologies that can make use of extremely large, structured datasets. These methodologies must re-address the tasks of investigating rare phenomena involving multiple lexical items, of finding and representing fine-grained sub-regularities, and of investigating variations within and across language domains. This should be accompanied by new methods to structure search results (in order to, among others, cope with false positives), or visualization techniques that facilitate the interpretation of results and formulation of new hypotheses.

On the other hand, some fundamental technical methods and strategies call for re-evaluation. These include, for example, efficient and sustainable curation of data, management of collections that span multiple volumes or that are distributed across several centres, innovative corpus architectures that maximize the usefulness of data, and techniques that allow the efficient search and analysis.

CMLC (Challenges in the Management of Large Corpora) gathers experts in corpus linguistics as well as in language resource creation and curation, in order to provide a platform for intensive exchange of expertise, results, and ideas. The first two meetings of CMLC were part of the LREC workshop structure, and were held in 2012 in Istanbul, and in 2014 in Reykjavík. This year's meeting, co-located with Corpus Linguistics 2015 in Lancaster, promises a good mix of technical and general topics.

The contributions by Evert and Hardie, by Schäfer, and by Tiepmar discuss, among others, systems and database architectures: Evert and Hardie propose a new data model for the well-established Corpus Workbench (CWB), Schäfer presents a tool chain for cleaning, annotation, and querying in the COW14 (“Corpora from the Web”) platform and moves on to issues of dataset management, while Tiepmar introduces a MySQL-based implementation of the Canonical Text Services (CTS) protocol in the context of the project “A Library of a Billion Words”.

Křen, Schäfer, and Tufiş *et al.* present news from ongoing projects: Křen's is a concise report on the current state of the Czech National Corpus, overviewing the data composition, internal and external tool architecture, as well as the user ecosystem. In a contribution mentioned above, Schäfer highlights some of the user-oriented decisions undertaken in COW14. Tufiş *et al.* provide an update on the state of development of the emerging national corpus of Romanian, CoRoLa.

The contributions by Graën and Clematide as well as by Buschjäger *et al.* focus primarily on emerging methodologies and new research questions posed by very large corpora. Graën and Clematide take on large multi-parallel corpora and discuss the data model and the storage architecture that facilitate efficient retrieval and mining. Buschjäger *et al.* apply the novel Word Embeddings approach to German data and demonstrate its usefulness for capturing subtle word relations and analogies.

Two CMLC-3 presentations contributed by some of the organizers and not included in the present volume, also provide project reports. Bański *et al.* present KorAP, a scalable corpus analysis platform designed to serve very large datasets, a deliverable of a project recently concluded at the Institut für Deutsche Sprache in Mannheim. Biber and Breiteneder discuss the way in which the Austrian Academy Corpus meets challenges presented by modern corpus linguistics.

We would like to thank the members of the Programme Committee for their effort in refereeing the contributions in the present volume. We would also like to express our sorrow at the passing of Adam Kilgarriff, who had participated in the reviewing process and whom we were hoping to meet again in Lancaster.

Recent Developments in the Czech National Corpus

Michal Křen
Charles University in Prague
Institute of the Czech National Corpus
michal.kren@ff.cuni.cz

1 Introduction

The Czech National Corpus (CNC) is a long-term project striving for extensive and continuous mapping of the Czech language. This effort results mostly in compilation, maintenance and providing free public access to a range of various corpora with the aim to offer a diverse, representative, and high-quality data for empirical research mainly in linguistics.

Since 2012, the CNC is officially recognized as a research infrastructure funded by the Czech Ministry of Education, Youth and Sports which has caused a recent shift towards user service-oriented operation of the project. All project-related resources are now integrated into the CNC research portal at http://www.korpus.cz/.

Currently, the CNC has an established and growing user community of more than 4,500 active users in the Czech Republic and abroad who put almost 1,900 queries per day using one of the user interfaces. The paper discusses the main CNC objectives for each particular domain, aiming at an overview of the current situation supplemented by an outline of future plans.

2 Corpus compilation

Most of the CNC corpora can be characterized as traditional (as opposed to the web-crawled corpora), with emphasis on cleared copyright issues, well-defined composition, reliable metadata and high-quality data processing.

Synchronic written corpora of the SYN series (Hnátková et al., 2014) with current overall size 2.2 billion word tokens (i.e. tokens not including punctuation). The series consists of three general-language representative corpora (containing a large variety of fiction, newspapers and professional texts) published every five years that cover consecutive time periods, and large newspaper corpora. The annotation of the SYN-series corpora includes detailed bibliographical information, lemmatization and morphological tagging.

Synchronic spoken corpora of the ORAL series with current overall size 4.8 million word tokens; the corpora include only unscripted informal dialogical speech. The newest corpus of the series, ORAL2013 (Válková et al., 2012), is designed as a representation of contemporary spontaneous spoken language used in informal situations on the area of the whole Czech Republic; it features manual one-layer transcription aligned with audio. A new ORTOFON series with two-layer transcription (orthographic and phonetic) has been recently established (Kopřivová et al., 2014).

Multilingual parallel corpus InterCorp (Čermák and Rosen, 2012; Rosen and Vavřín, 2012) with Czech texts aligned on sentence level with their translations to or from 30+ languages (some of them lemmatized and/or tagged). The core of the InterCorp consists of manually aligned and proofread fiction, and it is supplemented by collections of automatically processed texts from various domains. The total size of foreign-language texts is almost 1.4 billion word tokens, out of which 173 million make up the core (version 7 published in December 2014).

Diachronic corpus of historical Czech DIAKORP (Kučera and Stluka, 2014) with current size 2 million word tokens includes texts from the 14th century onwards. However, the current focus of DIAKORP development is on the 19th century.

Specialized corpora of various kinds and for specific research purposes that supplement the variety of hosted corpora. The specialized corpora include most prominently a dialectal corpus and a corpus of Czech texts written by the deaf (neither of them published yet).

3 Data processing and annotation

Apart from the corpus compilation, the CNC develops or adapts software technologies for data processing and annotation that supplement standard project-independent tools.

- Software environments for internal project **work flow management** of data collection (large networks of external collaborators for the spoken corpora and the InterCorp) and processing of various corpora. For the most part, the environments function as a web-based "wrapper" that combines both CNC and third-party tools.

  - **SynKorp** – database and data processing toolchain for the SYN-series corpora of written language (text conversion, clean-up, metadata annotation and text classification).

  - **Mluvka** – database and integrated project management system for coordination of spoken and dialectal data collection, manual two-layer annotation (orthographic and phonetic), expert revision and balancing.

  - Database of parallel texts and integrated project management system for coordination of the **InterCorp**, manual verification and revision of the alignment (implements a three-level project coordination hierarchy similar to Mluvka). The work flow includes **InterText** (Vondřička, 2014), a project-independent editor of aligned parallel texts.

- Tools for **linguistic annotation** of Czech language data on morphological and syntactic level. For this purpose, the CNC mostly adapts language-independent software tools and develops Czech-specific ones.

  - The **morphological level** includes Czech morphological analyser and lexicon (Hajič, 2004) (both provided by LINDAT/CLARIN; `http://lindat.mff.cuni.cz/`) that is being continuously administered in collaboration with the CNC. Subsequent morphological disambiguation involves a combination of language-independent stochastic tagger with rule-based components developed specifically for Czech (Hnátková et al., 2014; Jelínek, 2008; Petkevič, 2006; Spoustová et al., 2007). Works on extension of the current morphological annotation to spoken and diachronic data are already under way.

  - **Syntactic level** annotation is – similarly to the morphological one – carried out by Czech-specific adaptation of existing stochastic language-independent third-party tools for syntactic parsing and enhancement of their results by various methods, including rule-based corrections (Jelínek, 2014). The first syntactically parsed CNC corpus will be published by the end of this year.

4 Application development

Design and development of new intuitive analytical web-based applications as well as continuous enhancement of the existing ones are an integral part of the effort to promote empirical linguistic research. All the applications are open-source and all of them (except for KWords) currently use Manatee (Rychlý, 2007) as their backend.

KonText (`http://kontext.korpus.cz/`), a web-based general-purpose corpus concordancer (CNC fork of the NoSketch Engine; Rychlý, 2007) with built-in basic statistical functions, subcorpus manager, filtering, word-to-sound alignment support etc. It is the only application that requires user registration to switch from restricted functionality to regular access.

SyD (`http://syd.korpus.cz/`; Cvrček and Vondřička, 2011), a web application for corpus-based analysis of language variants. In the synchronic part, frequency distribution and collocations of variants can be compared across different domains of contemporary written and spoken texts, while the diachronic part shows their development over time.

Morfio (`http://morfio.korpus.cz/`; Cvrček and Vondřička, 2012), a web application for study of word formation and derivational morphology. It searches the corpus to identify and analyze selected derivational patterns, specified by prefixes, suffixes or word roots. It can be used to analyze morphological productivity of affixes and to estimate the accuracy of a selected derivational model in Czech.

KWords (`http://kwords.korpus.cz/`), a web application for corpus-based keyword and discourse analysis of Czech and English. It enables users to upload their own texts to be compared against one of the reference corpora available or against a selected text. It also supports the analysis and visualization of distance-based relations of keywords.

5 User services

User support and services are concentrated at the CNC research portal at `http://www.korpus.cz/`, a common platform for language research aimed at both the research community and the general public that integrates web applications mentioned above with active support. In addition to the research portal, the CNC offers also organization of workshops and lectures, involvement in academic training, expert consultations and tutoring etc.

- **User Forum**: a virtual platform accessible to all registered users. It features an advisory centre (with Q&A) that also handles all web requests for new application features and bug reports, which serve as a valuable source of user feedback.

- **CNC Wiki** (corpus linguistics knowledge base) with an on-line manual is freely available on the portal without registration. It contains an introduction into corpus linguistics, details about the CNC resources, and an on-line tutorial in seven lessons aimed at both beginners and advanced users (for the time being in Czech only).

- **Biblio**: a repository of CNC-based research outputs; users are encouraged not only to submit references about their research papers, books or theses based on CNC resources, but also to upload them directly to make them accessible to all visitors of the CNC portal.

- **Corpus hosting**: the CNC provides hosting service of – mostly small and/or specialized – corpora created at other institutions which do not have the possibility or know-how to ensure adequate final technical processing of their data (including quality checks with possible labour-intensive corrections). This is offered by the CNC, as well as maintenance of the resulting corpora, providing public access to them and related services; appropriate credit of the hosted corpus is always given, including a link to the relevant publication. Hosted corpora constitute a valuable enrichment of the CNC-compiled corpora and include learner corpora, web corpora and foreign-language corpora (including Upper and Lower Sorbian).

- **Data packages**: the CNC strives to be as open as possible also in terms of language data. On the other hand, restrictions arising from the laws in force have to be observed and this is one of the reasons why the CNC has introduced the service of providing data packages. This service enables users to obtain corpus-derived data with less restrictive licensing than the licensing of the original corpus texts. The data packages are either available through LINDAT/CLARIN repository, or they can be prepared in accordance with individual requirements of the particular user or institution. The licensing depends on the nature of the data and it ranges between the CC BY license (for word lists or n-grams for small n) to proprietary license that permits neither commercial use nor redistribution (for full texts shuffled at the sentence level).

6 Future plans

The applications and user services are planned to be maintained continuously, with new functionality added to the existing applications and new ones developed while responding to user requirements. To mention just a few planned enhancements:

- better visualization of query results, especially their diachronic development;
- multi-word unit identification and extraction component based on alternative approaches;
- interface enhancements leading the users to more appropriate interpretations and comparative statistical evaluation of corpus search results.

The spectrum of collected data will be broadened in the near future by adding semi-formal spoken language and by establishing a new corpus series that would contain selected specific semi-official language used on the internet, including blogs, discussion forums etc. (i.e. not yet another web corpus). In the long-term perspective, one of the main goals is to compile a monitor corpus of written Czech that would cover the period from 1850 to the present and enable a systematic and sophisticated study of language change. This corpus will help to eventually bridge the gap between the diachronic and synchronic data in the CNC, while taking full advantage of the CNC's twenty year tradition of data collection.

Acknowledgements

The data, tools and services described in this paper are a result of team work. Many thanks to all for their ideas, hard work and endurance that make the project possible.

This paper resulted from the implementation of the Czech National Corpus project (LM2011023) funded by the Ministry of Education, Youth and Sports of the Czech Republic within the framework of Large Research, Development and Innovation Infrastructures.

References

V. Cvrček and P. Vondřička. 2011. Výzkum variability v korpusech češtiny. In F. Čermák, editor, *Korpusová lingvistika Praha 2011. 2 Výzkum a výstavba korpusů*, pages 184–195. NLN, Praha.

V. Cvrček and P. Vondřička. 2012. Nástroj pro slovotvornou analýzu jazykového korpusu. In *Gramatika a korpus 2012*. Gaudeamus, Hradec Králové.

F. Čermák and A. Rosen. 2012. The case of InterCorp, a multilingual parallel corpus. *International Journal of Corpus Linguistics*, 13(3):411–427.

J. Hajič. 2004. *Disambiguation of Rich Inflection (Computational Morphology of Czech)*. Karolinum, Praha.

M. Hnátková, M. Křen, P. Procházka, and H. Skoumalová. 2014. The SYN-series corpora of written Czech. In *Proceedings of LREC2014*, pages 160–164, Reykjavík. ELRA.

T. Jelínek. 2008. Nové značkování v Českém národním korpusu. *Naše řeč*, 91(1):13–20.

T. Jelínek. 2014. Improvements to dependency parsing using automatic simplification of data. In *Proceedings of LREC2014*, pages 73–77, Reykjavík. ELRA.

M. Kopřivová, H. Goláňová, P. Klimešová, and D. Lukeš. 2014. Mapping diatopic and diachronic variation in spoken Czech: the Ortofon and Dialekt corpora. In *Proceedings of LREC2014*, pages 376–382, Reykjavík. ELRA.

K. Kučera and M. Stluka. 2014. Corpus of 19th-century Czech texts: Problems and solutions. In *Proceedings of LREC2014*, pages 165–168, Reykjavík. ELRA.

V. Petkevič. 2006. Reliable morphological disambiguation of Czech: Rule-based approach is necessary. In M. Šimková, editor, *Insight into the Slovak and Czech Corpus Linguistics*, pages 26–44. Veda, Bratislava.

A. Rosen and M. Vavřín. 2012. Building a multilingual parallel corpus for human users. In *Proceedings of LREC2012*, pages 2447–2452, İstanbul. ELRA.

P. Rychlý. 2007. Manatee/Bonito – a modular corpus manager. In *1st Workshop on Recent Advances in Slavonic Natural Language Processing*, pages 65–70, Brno.

J. Spoustová, J. Hajič, J. Votrubec, P. Krbec, and P. Květoň. 2007. The best of two worlds: Cooperation of statistical and rule-based taggers for Czech. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing, ACL 2007*, pages 67–74, Praha.

L. Válková, M. Waclawičová, and M. Křen. 2012. Balanced data repository of spontaneous spoken Czech. In *Proceedings of LREC2012*, pages 3345–3349, İstanbul. ELRA.

P. Vondřička. 2014. Aligning parallel texts with InterText. In *Proceedings of LREC2014*, pages 1875–1879, Reykjavík. ELRA.

CoRoLa Starts Blooming – An update on the Reference Corpus of Contemporary Romanian Language

Dan Tufiș, Verginica Barbu Mititelu, Elena Irimia, Ștefan Daniel Dumitrescu, Tiberiu Boroș
Research Institute for Artificial Intelligence "Mihai Drăgănescu"
13 Calea 13 Septembrie, 050711, Bucharest, Romania
{tufis, vergi, elena, sdumitrescu, tibi}@racai.ro

Horia Nicolai Teodorescu, Dan Cristea, Andrei Scutelnicu, Cecilia Bolea, Alex Moruz, Laura Pistol
Institute for Computer Science, Iași
2 T. Codrescu St, 700481, Iași, Romania
hteodor@etti.tuiasi.ro, dcristea@info.uaic.ro, andreiscutelnicu@gmail.com, ceciliа.bolea@iit.academiaromana-is.ro, mmoruz@info.uaic.ro
laura.pistol@iit.academiaromana-is.ro

Abstract

This article reports on the on-going CoRoLa project, aiming at creating a reference corpus of contemporary Romanian (from 1945 onwards), opened for online free exploitation by researchers in linguistics and language processing, teachers of Romanian, students. We invest serious efforts in persuading large publishing houses and other owners of IPR on relevant language data to join us and contribute the project with selections of their text and speech repositories. The CoRoLa project is coordinated by two Computer Science institutes of the Romanian Academy, but enjoys cooperation of and consulting from professional linguists from other institutes of the Romanian Academy. We foresee a written component of the corpus of more than 500 million word forms, and a speech component of about 300 hours of recordings. The entire collection of texts (covering all functional styles of the language) will be pre-processed and annotated at several levels, and also documented with standardized metadata. The pre-processing includes cleaning the data and harmonising the diacritics, sentence splitting and tokenization. Annotation will include morpho-lexical tagging and lemmatization in the first stage, followed by syntactic, semantic and discourse annotation in a later stage.

1 Introduction

In 2012 the Romanian Academy Research Institute for Artificial Intelligence "Mihai Drăgănescu" from Bucharest (RACAI) finalized the Romanian Balanced Corpus (ROMBAC[1]) (Ion et. al, 2012) containing 44,117,360 tokens covering four domains (News, Medical, Legal, Biographic and Fiction). The nucleus of ROMBAC was represented by the RoCo_News corpus (Tufiș and Irimia, 2006), a hand validated corpus of almost 7 million tokens from the weekly magazine Agenda (2003-2006).

Since 2014 the concern for creating a bigger corpus has been joined by the Institute for Computer Science in Iasi, in a larger priority project of the Romanian Academy: The Reference Corpus of Contemporary Romanian Language.

The time span covered by the project is 1945-present, with two subperiods (1945-1990, 1990-present), with clear differences, mainly at the lexical level. From this perspective, a big challenge for us is the collection of electronic texts to cover the whole period. For the last couple of decades there is an important amount of such texts available. However, in the case of the texts from previous decades considerable effort needs to be done for finding the owners of the texts IPR, for scanning, OCRizing and correcting the

[1] http://www.meta-net.eu/meta-share

texts. This could imply raising the awareness of main libraries about the cultural responsibility of digitizing even contemporary books, not only the old ones.

2 Objectives

When finished, CoRoLa will be a medium to large corpus (more than 500 million word forms), IPR cleared, in which all functional styles will be represented: scientific, official, publicistic and imaginative. Although the colloquial style is not a major concern for us, it will definitely be included, due to its use in imaginative writing. The provisional structure of the corpus is described in some details in Barbu Mititelu and Irimia (2014). Unlike its predecessor, CoRoLa will include a syntactically annotated sub-corpus (treebank) and an oral component. All textual data will be morpho-lexically processed (tokenized, POS-tagged and lemmatized). The treebank (we target 10,000 hand validated sentences) and the oral component (targeted: 300 hours of transcribed recorded speech) have additional annotations (dependency links, respectively speech segmentation at sentence level, pauses, non-lexical sounds and partial explicit marking of the accent).

Particular attention is paid to data documentation, i.e. associating it with standardized metadata. We adopted the CMDI (Component MetaData Infrastructure)[2] approach for the creation of our metadata.

3 Data Collection and Cleaning

The resource we are building will have two important attributes: it will be representative for the language stage, thus covering all language registers and styles; it will be IPR cleared, which is a challenging task, triggered by the need to observe the intellectual property law. The categories of content excepted by this law are: political, legislative, administrative and judicial. Therefore, without the written accept from IPR owners, from the other kinds of texts only tiny fragments of no more than 10,000 characters can be used. We must also consider only texts written with correct diacritics (otherwise, the linguistic annotation will be highly incorrect).

To ensure the volume and quality of the texts in the corpus, as well as copyright agreements on these texts, our endeavour was to establish collaborations with publishing houses and editorial offices. So far (March 2015), we have signed agreements with the following publishing houses: Humanitas, Polirom, Romanian Academy Publishing House, Bucharest University Press, "Editura Economică", ADENIUM Publishing House, DOXOLOGIA Publishing House, the European Institute Publishing House, GAMA Publishing House, PIM Publishing House. Some magazines and newspapers have also agreed to help our project by providing access to their articles: România literară, Muzica, Actualitatea muzicală, Destine literare, DCNEWS, PRESSONLINE.RO, the school magazine of Unirea National College from Focșani, SC INFOIASI SRL, Candela de Montreal. Until now four bloggers have also agreed to allow us to include some of their posts in the corpus: Simona Tache[3], Dragoș Bucurenci[4], Irina Șubredu[5] and Teodora Forăscu[6]. Also, we have signed agreements with the writers Corneliu Leu and Liviu Petcu.Oral texts (read news, live transmissions and live interviews) (one hour per working day) are provided by Rador (the press agency of Radio Romania) and Radio Iaşi – a local broadcasting agency. All data providers readiness to get involved was a very pleasant surprise for us and we express here, again, our gratitude.

Another challenge in corpus creation is to have texts in a clean format, easy to process and annotate. Once our collaborators dispatch a textual resource (usually in unprotected pdf files, rarely in doc files), the first step is to convert it into an adequate format for our pre-processing tools[7].

Given the large amount of texts, we automated a part of the process (Moruz and Scutelnicu, 2014): the text is automatically retrieved from the pdf files, paragraph limits are recuperated, column marking newlines are erased as well as hyphens at the end of the lines. However, a lot of manual work remains to be done: separating articles from periodicals in different files, removal of headers, footers, page numbers, figures, tables, dealing with foot- or end-notes, with text fragments in foreign languages, with excerpts from other authors, etc. When copied from their original sources, the content is converted into the UTF-8 encoding and saved as plain text documents.

[2] http://www.clarin.eu/content/component-metadata

[3] http://www.simonatache.ro

[4] http://bucurenci.ro

[5] http://irina.subredu.name

[6] https://travelearner.wordpress.com

[7] http://www.racai.ro/en/tools/

CoRoLa is developed and refined in successive steps and the automatic processing chain of the texts to be included has to conform to the format requested by the indexing and searching platform, IMS Open Corpus Workbench (CWB, http://cwb.sourceforge.net/), an open source medium that allows complex searching with multiple criteria and support for regular expressions. It allows to choose the (sub)corpus/(sub)corpora with which to work (choose from among the domains and subdomains, but also from the available authors), to find out words frequencies in a (specified) (sub)corpus, to search for a word or a word form, to search for more words (either consequent or permitting intervening words), to find words collocations and co-occurrences (within a window of a pre-established size), to find lexicalization of specified morphological or/and syntactic structures, n-gram models, etc. The platform has already been installed and tested on the ROMBAC corpus and coupled with our processing chain which produces the adequate annotated format for morphological and shallow syntactic searches. For the near future, we plan to switch to the more powerful corpus management platform KorAP (Bański et al., 2014).

The TTL (Ion, 2007) processing chain ensures, at the time of this writing, the following specific functionalities: sentence splitting, tokenisation, tiered-tagging (Tufiş, 1999), lemmatising and chunking. Future services regarding processing and query facilities for discourse (Cristea & Pistol, 2012) will be provided. CoRoLa will be automatically annotated, but a fragment of it (~2%) will be manually validated.

4 Current Statistics

4.1 Textual Data

At the moment, the corpus contains the data presented in Table 1, where one can notice the domain distribution of the texts, as well as quantitative data related to each domain: tokens (word forms and punctuation).

A finer classification of the documents, according to their sub-domains, outlines the following categories: literature, politics, gossip columns, film, music, economy, health, linguistics, theatre, painting/drawing, law, sport, education, history, religious studies and theology, medicine, technology, chemistry, entertainment, environment, architecture, engineering, pharmacology, art history, administration, oenology, pedagogy, philology, juridical sciences, biology, social, mathematics, social events, philosophy, other.

In parallel with the CoRoLa corpus, at ICIA and UAIC a Romanian treebank is under development (Irimia and Barbu Mititelu, 2015), (Perez, 2014), (Mărănduc and Perez, 2015). Currently each of the two sections of the treebank contains almost 5,000 sentences, which are in the process of being mapped into the UD project specifications[8]. The final version of the CoRoLa corpus will include the Romanian treebank as well.

DOMAIN		STYLE	
arts&culture	32,838,881	journalistic	44,248,356
society	33,582,123	science	26,990,172
others	9,990,383	imaginative	11,945,283
science	19,923,533	others	1,777,475
nature	106,196	memoirs	1,511,676
		administra-tive	865,660
		law	9,102,494
TOTAL[9]	96,441,116	TOTAL	96,441,116

Table 1. Domain and style distribution of textual data.

4.2 Speech data

Speech data collected so far is accompanied by transcriptions (observing the current orthography). Partially (about 10%), it was automatically pre-processed and the transcriptions were XML encoded with mark-up for lemma, part-of-speech and syllabification. Additionally to the XML annotations we provide 3 files which contain the original sentences (".txt" extension) the stripped version (which is obtained by removing all punctuation from the original sentences – useful in training systems such as Sphinx or HTK (Hidden Markov Model Toolkit) – ".lab" extension) and time aligned phonemes (tab separated values which contain each phoneme in the text with its associated start and stop frame – ".phs" extension).

[8] https://code.google.com/p/uni-dep-tb/

[9] Currently more textual data, not included into CoRoLa, has been collected, which may be used for improving models of our statistical processing tools. Among them are Wiki-Ro, the Romanian part of a big collection of sentences extracted from Wikipedia within the ACCURAT European project (http://www.accurat-project.eu/) and the Romanian part of the Acquis-Communautaire (Steinberger et al. 2006). They are already pre-processed and contain more than 50 million words. Similarly, we acquired some audio-books (not IPR clarified and thus, not included into CoRoLa) used only for evaluation of our tools.

- **RASC** (Romanian Anonymous Speech Corpus) is a crowd-sourcing initiative to record a sample of sentences randomly extracted from Ro-Wikipedia (Tufiş et al., 2014). The corpus is automatically aligned at phoneme/word level.
- **RSS-ToBI** (Romanian Speech Synthesis Corpus) is a collection of high quality recordings compiled by (Stan et al., 2011) and designed for speech synthesis. It was enhanced with a prosodic ToBI-like (Tone and Break Indices) annotation (reference to be added). It is automatically aligned at phoneme/word level.
- **RADOR (**Radio Romania) and **Radio Iaşi** is a collection of radio news and interviews, provided daily by the Romanian Society for Broadcasting and the main Iaşi radio channel. At the time of this writing, the transcriptions are under preprocessing. They are not yet aligned at phoneme/word level.

Corpus	Type	Source	Time length (h:m:s)
RASC	many speakers	RoWikipedia	04:22:02
RSS-ToBI	single speaker	news&fairy tales	03:44:00
RADOR	many speakers	news& interviews	106:52:33
Radio Iaşi	many speakers	interviews	07:00:00 under development
			>121:58:35

Table 2. Speech corpora.

Besides these speech corpora, we contracted professional recordings (about 10 hours) of sentences selected by us from Romanian Wikipedia. These recordings will enlarge the RASC corpus.

Further information on the already processed speech data are given in the table below.

Corpus	sentences	words	phonemes
RASC	2,866	39,489	270,591
RSS-ToBI	3,500	39,041	235,150
	6,266	78,530	505,741

Table 3. Currently pre-processed speech corpora

A special mention deserves the site "Sounds of the Romanian Language" (Feraru et al., 2010), which is a systematically built, explanatory small collection of annotated and documented recordings of phonemes, words, and sentences in Romanian, pronounced repeatedly by several speakers; the corpus also includes as annex materials numerous papers on the topic and several instruments for speech analysis. Sections of the corpus are devoted to emotional speech, to specific processes as the double subject, and to phonetic pathologies. The corpus is maintained by the Institute for Computer Science of the Romanian Academy[10].

5 Metadata Creation

The challenge in CoRoLa is to create a corpus from which more than only concordances to be extracted, i.e. giving the user the possibility to construct his/her own subcorpus to work with, depending on the domain/style/period/author/etc. The only way to obtain this is to document each file with metadata. For documents sent by publishing houses, etc., we created the metadata files manually. For text files crawled from the web (articles, blogs), we automatically created metadata, with a preliminary phase of mapping the existent classifications of texts on those sites onto our classification of texts.

6 Annotation of the data

As mentioned before, a processing chain[11] has been established, consistent with the tabular encoding specific to the CWB platform and comprising more program modules that execute particular functions. The web-service chain provides:

- sentence splitting: it uses regular expressions for the identification of a sentence end;
- tokenization: the words are separated from the adjacent punctuation marks, the compound words are recognized as a single lexical atom and the cliticized words are split as distinct lexical entities;
- POS tiered-tagging with the large MULTEXT-East tag set; its accuracy is above 98%;
- lemmatization: based on the tagged form of the word, it recovers its corresponding lemma from a large (over 1,200,000 entries) human-validated Romanian word-form lexicon; the precision of the algorithm measured on running texts is almost 99%; for the unknown words (which are not tagged as proper names), the lemma is provided by a five-gram letter Markov

[10] http://www.etc.tuiasi.ro/sibm/romanian_spoken_language/en/

[11] http://ws.racai.ro/ttlws.wsdl

Model-based guesser, trained on lexicon lemmas with the same POS tag as the token being lemmatized. The accuracy of the lemma guesser is about 83%. A better lemma-guessing (about 93%) is ensured by a new neural network based-tagger (Boroș et al., 2013), not yet integrated in the processing chain for CWB.

- chunking: for each lexical unit previously tagged and lemmatized, the algorithm assigns a syntactic phrase, guided by a set of regular expression rules, defined over the morpho-syntactic descriptions.

For the further stages in the corpus development, we envisage adding other types of annotations: syntactic parsing, semantic annotation and discourse analysis.

The annotation of the speech data includes, additionally, the syllabation and accent mark-up plus the grapheme to phoneme alignment.

7 Annotation correction

In our previous experiments (Tufiș and Irimia, 2006) with the task of collecting corpora and ensuring a satisfying quality of the resources, we implemented a coherent methodology for the automatic identification of annotation errors.

Most of the errors identified in this manner can also be automatically corrected. This validation procedure was used in the past to correct tagging and lemmatization errors for the journalistic corpus RoCo_News and for ROMBAC and reduced the estimated error rates to around 2%.

The TTL processing workflow explicitly marks the out-of-dictionary words (ODW), excepting proper nouns, abbreviations and named entities. The ODW can be extracted, sorted and counted, then divided into frequency classes. In the past, we concentrated our analysis on the words with at least two occurrences in the corpus (assuming that the others are typographic errors or foreign words) and structured them into error classes, thus being able to split them into errors that need human correction and errors that can be dealt with by implementing automatic correction strategies.

Besides using the mentioned methodology to improve the quality of the entire corpus, we intend to manually validate a limited part of it (2%, i.e. 10 million words). As the process of collecting and managing such an important resource is a life-time task, our attention on assuring its quality will continuously accompany this enterprise.

8 Conclusions

In the international context of growing interest for creating large language resources, we presented here the current phase in the creation of a reference corpus of contemporary Romanian. It is a joined effort of two academic institutes, greatly helped by publishing houses and editorial offices, which kindly accepted the inclusion of their texts at no costs. The corpus will be available for search for all those interested in the study or processing of the Romanian language.

We emphasize the idea that, although large amount of texts are out there on the web, creating an IPR clear reference corpus is quite a challenge, not only due to vast efforts invested in persuading IPR holders to contribute to a cultural action, but also to achieve agreements on what texts and how much of them to include in the corpus. In spite of the decided CoRoLa structure (text types and quantities) of the linguistic data the supplementary data we manage to collect (mainly from the web) is not discarded, but stored for training specialized statistical models to be used in different data-driven applications (CLIR, Q&A, SMT, ASR, TTS).

Acknowledgements

We express here our gratitude to all CoRoLa volunteers, undergraduate, graduate and Ph.D. students, as well as researchers and university staff in computer science and linguistics, who, noble-minded and aware of the tremendous importance that such a corpus will have for the Romanian culture, have generously agreed to help in the process of filling in metadata and cleaning the collection of texts.

References

P. Bański, N. Diewald, M. Hanl, M. Kupietz, A. Witt. 2014. Access Control by Query Rewriting. The Case of KorAP. *Proceedings of the Ninth Conference on International Language Resources and Evaluation (LREC'14)*: 3817-3822.

V. Barbu Mititelu, E. Irimia. 2014. The Provisional Structure of the Reference Corpus of the Contemporary Romanian Language (CoRoLa). In M.Colhon, A. Iftene, V. Barbu Mititelu, D. Tufiș (eds.) *Proceedings of the 10th Intl. Conference "Linguistic Resources and Tools for Processing Romanian Language"*: 57-66.

T. Boroș, R. Ion, D. Tufiș. 2013. Large tagset labeling using Feed Forward Neural Networks. Case study on Romanian Language. *Proceedings of ACL 2013*: 692-700.

T. Boroș, A. Stan, O. Watts, S.D. Dumitrescu. 2014. RSS-TOBI - A Prosodically Enhanced Romanian Speech Corpus. *Proceedings of 9th LREC 2014*: 316-320.

D. Cristea, I.C. Pistol. 2012. Multilingual Linguistic Workflows. In Cristina Vertan and Walther v. Hahn (Eds.) *Multilingual Processing in Eastern and Southern EU Languages. Low-resourced Technologies and Translation*, Cambridge Scholars Publishing, UK: 228-246.

S.D. Dumitrescu, T. Boroș, R. Ion. 2014. Crowd-Sourced, Automatic Speech-Corpora Collection–Building the Romanian Anonymous Speech Corpus. *CCURL 2014: Collaboration and Computing for Under-Resourced Languages in the Linked Open Data Era*: 90-94.

S.M. Feraru, H.N. Teodorescu, M.D. Zbancioc. 2010. SRoL - Web-based Resources for Languages and Language Technology e-Learning. *International Journal of Computers Communications & Control*, Vol. 5, Issue 3: 301-313.

R. Ion. 2007. *Word Sense Disambiguation Methods Applied to English and Romanian*, PhD thesis, Romanian Academy (in Romanian).

R. Ion, E. Irimia, D. Ştefănescu, D. Tufiş. 2012. ROMBAC: The Romanian Balanced Annotated Corpus. In Nicoletta Calzolari et al. (Eds.) *Proceedings of the 8th LREC*: 339-344.

E. Irimia, V. Barbu Mititelu. 2015. Building a Romanian Dependency Treebank, Proceedings of *Corpus Linguistics 2015*.

A. Hardie. 2012. CQPweb – combining power, flexibility and usability in a corpus analysis tool. *International Journal of Corpus Linguistics* 17 (3): 380–409.

C. Mărănduc, A.C. Perez. 2015. A Romanian dependency treebank. *Proceedings of CICLing 2015*.

A. Moruz, A. Scutelnicu. 2014. An Automatic System for Improving Boilerplate Removal for Romanian Texts. In M. Colhon, A. Iftene, V. Barbu Mititelu, D. Cristea, D. Tufiș, *Proceedings of the 10th International Conference "Linguistic resources and Tools for Processing the Romanian Language"*: 163-170.

A. Perez. 2014. Resurse lingvistice pentru prelucrarea limbajului natural (Lingvistic Resources For Natural Language Processing). Ph.D. thesis, „Alexandru Ioan Cuza" University of Iași.

J. Sinclair. 1996. *EAGLES – Preliminary recommendations on Corpus Typology* EAG--TCWC--CTYP/P

A. Stan, J. Yamagishi, S. King, M. Aylett. 2011. The Romanian Speech Synthesis (RSS) corpus: building a high quality HMM-based speech synthesis system using a high sampling rate. *Speech Communication, 53*(3): 442-450.

R. Steinberger, B. Pouliquen, A. Widiger, C.Ignat, T. Erjavec, D. Tufiș, D. Varga. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In Proceedings of the 5th LREC Conference, Genoa, Italy, 22-28 May, 2006, pp.2142-2147, ISBN 2-9517408-2-4.

D. Tufiș. 1999. Tiered Tagging and Combined Classifiers. In F. Jelinek, E. Nöth (eds) *Text, Speech and Dialogue*, Lecture Notes in Artificial Intelligence 1692, Springer: 28-33.

D. Tufiș, E. Irimia. 2006. RoCo_News - A Hand Validated Journalistic Corpus of Romanian. In *Proceedings of the 5th LREC*: 869-872.

D. Tufiș, R. Ion, A. Ceauşu, D. Ştefănescu. 2008. RACAI's Linguistic Web Services. In Nicoletta Calzolari et al. (Eds.) *Proceedings of the 6th LREC*: 327-333.

D. Tufiș, R. Ion, Ș. D. Dumitrescu, D.Ștefănescu. 2014. Large SMT data-sets extracted from Wikipedia. In *Language Resources and Evaluation Conference (LREC 14)*.Reykjavik, Iceland, May 2014

Discovering Subtle Word Relations in Large German Corpora

Sebastian Buschjäger
Lehrstuhl für Künstliche Intelligenz
TU Dortmund

Lukas Pfahler
Lehrstuhl für Künstliche Intelligenz
TU Dortmund

Katharina Morik
Lehrstuhl für Künstliche Intelligenz
TU Dortmund

{sebastian.buschjaeger,lukas.pfahler,katharina.morik}@udo.edu

Abstract

With an increasing amount of text data available it is possible to automatically extract a variety of information about language. One way to obtain knowledge about subtle relations and analogies between words is to observe words which are used in the same context. Recently, Mikolov et al. proposed a method to efficiently compute Euclidean word representations which seem to capture subtle relations and analogies between words in the English language. We demonstrate that this method also captures analogies in the German language. Furthermore, we show that we can transfer information extracted from large non-annotated corpora into small annotated corpora, which are then, in turn, used for training NLP systems.

1 Motivation

Large text corpora are a rich source of information for testing language properties. Once we formulate a linguistic hypothethis, we can formulate queries to collect evidence from the corpus (Klein and Geyken, 2010). However, very large corpora allow us to perform automatic exploration of the corpus to identify subtle relations between words or word groups.

Unfortunately, the analysis of large corpora is computationally challenging. As the size of a corpus grows, the size of the used vocabulary also grows, because a larger subset of language is covered. We found that the German Wikipedia contains more than 1.6 million unique words.

In order to find instances of all possible word-word relations or word classes, a very large sample of text data must be drawn. We usually refer to this problem as the "curse of dimensionality". However, for most Natural Language Problems, only little annotated training data is available.

Recently, Mikolov et al. (2013c) introduced a method for discovering linguistic regularities in large corpora based on neural networks. Their method learns a mapping from words to vectors in $\mathbb{R}^D$ called *word embeddings*. Embeddings allow simple vector operations that seem to capture syntactical and semantical regularities. This method has been successfully applied to English text corpora. For the first time, we thoroughly evaluate this method for the German language.

Our goal is to extract information on word relations from large unannotated corpora and enrich smaller annotated corpora like the TüBa-D/Z treebank (Telljohann et al., 2009) – a collection of German newspaper articles – with this information. More specifically, we want to discover word similarities and analogies in order to aggregate words into groups.

The rest of this paper is organized as follows. In section 2 we formally introduce Mikolov's word embeddings, in section 3 we present our experiments for German and English Wikipedia documents. Then in section 4 we show related work. Section 5 concludes our work.

2 Word Embeddings

Mikolov et al. proposed a neural language model that estimates word transition probabilities from a training corpus (2013c). By gradually reducing the complexity of their model, the authors enable the efficient use of large text corpora resulting in a simple neural network with input layer, linear projection layer and log-linear output layer (Mikolov et al., 2013a; Mikolov et al., 2013b). The projection layer of this model implicitly calculates a mapping $u : \mathcal{V} \mapsto \mathbb{R}^D$ from the vocabulary $\mathcal{V}$ to the space of word embeddings $\mathbb{R}^D$.

Surprisingly, these embeddings show striking syntactic and semantic properties that allow us to

perform simple vector operations, e.g.,

$$u(Paris) - u(France) + u(Italy) \approx u(Rome)$$

In order to train such an embedding, Mikolov et al. present two closely related network topologies (cf. figure 1). The first model, called CBOW, estimates probabilities for words $v_i \in \mathcal{V}$ given their surroundings $w_1, \ldots, w_N$ using a softmax function. Let U be a weight matrix shared across all contextual words $w_1, \ldots, w_N$ and let $W_{i\bullet}$ denote the $i-$th row of the output matrix W, then this model can be formulated as follows:

$$\begin{aligned} \tilde{u} &= \sum_{i=1}^{N} U w_i \\ p(v_j | w_1, \ldots, w_N) &= \frac{\exp(W'_{j\bullet} \tilde{u})}{\sum_{i=1}^{V} \exp(W'_{i\bullet} \tilde{u})} \end{aligned}$$

The second model, called Skip-Gram (SG), reverses the CBOW task. Given a single word $v_i \in \mathcal{V}$ it estimates the probabilities for the surrounding contextual words $w_1, \ldots, w_N$. The mathematical formulation for this model is naturally extracted from the CBOW model by adding multiple output matrices $W^{(1)}, \ldots, W^{(N)}$ to the model while reducing the input layer to one word.

The authors show, that the word embeddings u capture semantic relations between words by using simple vector operations. Additionally, they find that similar words have similar embeddings by the means of Cosine similarity. This enables efficient queries for word similarities in a vocabulary since the word embeddings can be efficiently computed as a look-up in table u and the Cosine similarity can be implemented as linear-time vector operation.

3 Experiments

3.1 Training German Word Embeddings

We train our word embeddings using the German Wikipedia (Wikimedia, 2015). This set contains roughly 591 million words with a vocabulary of 1.6 million words. As a comparison, word embeddings for the English Wikipedia with approximately 1.7 billion words and a vocabulary size of 1.7 million words are trained as well (Wikimedia, 2015). An available subset of word embeddings computed by Mikolov et al. on a large GoogleNews text corpus will serve as a reference value for our experiments (Mikolov, 2015).

3.2 Identifying Word Analogies

Mikolov et al. analyze the accuracy of word embeddings on semantic and syntactic relations based on a test set. This test set contains phrases of the form "a is to b what c is to d." for different categories of relations, e.g.

`king` is to `queen` what `man` is to `woman`

The task of this test set is to predict the word d where words a, b, c are given. To do so, a simple nearest neighbor prediction is used:

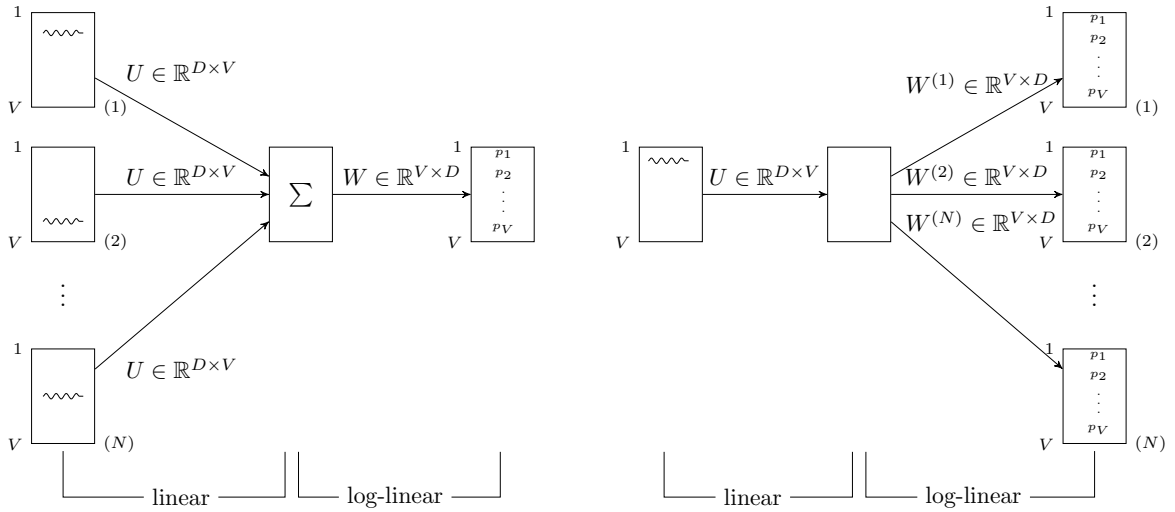
$$\widehat{d} = \underset{v \in \mathcal{V}}{\operatorname{argmin}} \{ \| u(a) - u(b) + u(c) - u(v) \|_2^2 \}$$

A question is correctly answered if $\widehat{d}$ equals d.

For the first time, we analyzed the accuracy of word embeddings in the German language. Therefore, we half-automatically translated this English test set into German using (Moraes, 2015). Additionally to this regularity test, we analyzed the performance of word embeddings on word analogies. To do so, we assembled a list of one thousand nouns for the German and English language. For every German noun, we queried twelve synonyms on average using OpenThesaurus (Naber, 2015). For the English language, OpenOffice (Foundation, 2015) provided a synonyms dictionary with thirteen synonyms per noun on average. We then computed the average Cosine similarity between word embeddings and their synonyms embeddings. As a reference we computed the average Cosine similarity between random nouns.

Results for the regularity test are presented in table 1. As you can see, the word embeddings capture regularities between nouns in the German language quite well (cf. category "capital-common" and "capital-world"), but show relatively poor performance on plural forms and past tense (cf. category "gram7" and "gram8"). Reasons for this may lie in the lexical character of the underlying training corpus, the relatively small size of the German Wikipedia compared to the English Wikipedia and Google News-Corpus as well as irregularities in word construction in the German language.

In table 2 the results of the synonym test can be found. The picture reverse here in contrast to the results in table 1. The average Cosine similarity for analogous words in the German language are roughly twice as high as for the English language. The average Cosine similarity between random nouns is, as expected, nearly zero.



(a) CBOW model

(b) Skip-Gram model

Figure 1: Network topology for CBOW and Skip-Gram model.

category	ref.	English	German	
		CBOW	SG	CBOW
capital-common	81.60	86.96 (+5.36)	**93.48** (+9.88)	91.70 (+8.1)
capital-world	83.30	**91.29** (+7.99)	82.55 (+1.25)	84.88 (+1.58)
gram7-past-tense	64.49	**65.26** (+0.8)	42.11 (-22.38)	42.17 (-22.32)
gram8-plural	86.64	**84.01** (+2.63)	45.16 (-41.48)	48.02 (-38.62)
gram9-pl-verb	67.93	62.07 (-5.86)	62.83 (-5.1)	**65.15** (-2.78)

Table 1: Accuracy for regularity test (excerpt).

	ref.	English	German	
		CBOW	SG	CBOW
synonyms	0.25	0.26	0.56	0.56
random nouns	0.08	0.04	0.06	0.05

Table 2: Average Cosine similarity.

3.3 Enriching Small Annotated Corpora with Word Embeddings

We want to demonstrate that natural language processing problems that rely on relatively small annotated corpora as training data can benefit from word embeddings learned on large, non-annotated corpora. We have seen that similar words have similar word embeddings. Clustering the embeddings with k-Means thus yields k partitions of similar words. Enriching a small annotated training corpus by tagging each word with the partition it belongs to has two possible advantages: First, we can handle unknown words the same way as words with similar embeddings. Second, we can pool related words and can estimate more reliable statistics for rare words (Andreas and Klein, 2014).

In our experiment, we consider the TüBa-D/Z treebank (Telljohann et al., 2009), a corpus of merely 3,444 newspaper articles whose sentences are annotated with dependence trees. This treebank is widely used for training natural language parsers for both constituency and dependency grammars. We evaluate a classification problem closely related to dependency parsing, where for an unlabeled arc in a given parsetree we want to predict the label of the arc. The TüBa-D/Z treebank in `.conll` dependency tree format has 34 classes of dependencies (Foth, 2006). We use Naive Bayes for classification using features for the word, the lemma and the POS-tag of both the head and tail of the arc. Additionally, we use the cluster of the word embedding for the corresponding word as a feature.

We select $k \approx \sqrt{1.6M}$, such that the space of pairs of words is about the size of the vocabulary. This makes estimating statistics about pairs of words feasible. Using a 10-fold, linearly split cross validation we show an accuracy

of $87.33 \pm 0.43\%$ using only traditional features. Using the additional features based on word embedding clusters, we get an accuracy of $88.33 \pm 0.43\%$, which is a significant increase of 1%.

4 Related Work

There have been many attempts to incorporate word embeddings into existing natural language processing solutions for the English language. Examples include Named-Entity Recognition (Turian et al., 2009), Machine Translation (Zou et al., 2013), Sentiment Analysis (Maas et al., 2011) or Automatic Summarization (Kageback et al., 2014). For Natural Language Parsing, there have been attempts to improve parser training by incorporating new features based on word embeddings. Andreas and Klein investigated their usefulness for constituency parsing (2014), Hisamoto et al. (2013) and Bansal et al. (2014) for dependency parsing. Their features are also based on clustered word embeddings and they also report small, but significant increases in accuracy for English dependency parsing.

5 Conclusion

We have shown that word embeddings can capture word similarities and word analogies for the German language. We demonstrated a significant improvement of parse tree labeling accuracy for German TüBa-D/Z treebank based on word embeddings.

References

Jacob Andreas and Dan Klein. 2014. How much do word embeddings encode about syntax. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Kilian Foth. 2006. Eine Umfassende Dependenzgrammatik des Deutschen.

Apache Foundation. 2015. Lingucomponent Sub-Project: Thesaurus Development. `http://www.openoffice.org/lingucomponent/thesaurus.html`. [Online; accessed on 02/18/2015].

Sorami Hisamoto, Kevin Duh, and Yuji Matsumoto. 2013. An Empirical Investigation of Word Representations. In *Proceedings of ANLP*, number C.

Mikael Kageback, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive Summarization using Continuous Vector Space Models. *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@EACL 2014*, pages 31–39.

Wolfgang Klein and Alexander Geyken. 2010. Das digitale Wörterbuch der deutschen Sprache (dwds). *Lexicographica*, 26:79–93.

Andrew L Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *2013 Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings*, pages 746–751.

Tomas Mikolov. 2015. word2vec – Tool for computing continuous distributed representations of words. `http://code.google.com/p/word2vec/`. [Online; accessed on 02/16/2015].

Manuela Moraes. 2015. Glosbe API. `https://glosbe.com/a-api`. [Online; accessed on 02/18/2015].

Daniel Naber. 2015. OpenThesaurus.de. `https://www.openthesaurus.de/about/download`. [Online; accessed on 02/18/2015].

Heike Telljohann, Erhard W Hinrichs, Sandra Kübler, Heike Zinsmeister, and Kathrin Beck. 2009. Stylebook for the Tübingen treebank of written German (TüBa-D/Z).

Joseph Turian, L Ratinov, Y Bengio, and D Roth. 2009. A preliminary evaluation of word representations for named-entity recognition. *NIPS Workshop on Grammar Induction*, pages 1–8.

Wikimedia. 2015. Wikipedia Dumps. `http://dumps.wikimedia.org/`. [Online; accessed on 02/24/2015].

Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. 2013. Bilingual Word Embeddings for Phrase-Based Machine Translation. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*.

Challenges in the Alignment, Management and Exploitation of Large and Richly Annotated Multi-Parallel Corpora

Johannes Graën
Institute of Computational Linguistics
University of Zurich
Zurich, Switzerland
graen@cl.uzh.ch

Simon Clematide
Institute of Computational Linguistics
University of Zurich
Zurich, Switzerland
siclemat@cl.uzh.ch

Abstract

The availability of large multi-parallel corpora offers an enormous wealth of material to contrastive corpus linguists, translators and language learners, if we can exploit the data properly. Necessary preparation steps include sentence and word alignment across multiple languages. Additionally, linguistic annotation such as part-of-speech tagging, lemmatisation, chunking, and dependency parsing facilitate precise querying of linguistic properties and can be used to extend word alignment to sub-sentential groups. Such highly interconnected data is stored in a relational database to allow for efficient retrieval and linguistic data mining, which may include the statistics-based selection of good example sentences. The varying information needs of contrastive linguists require a flexible linguistic query language for ad hoc searches. Such queries in the format of generalised treebank query languages will be automatically translated into *SQL* queries.

1 Introduction

The long-term goal of our project is the creation of a means for empirical linguistic research based on large amounts of multi-parallel texts, i.e. corresponding data for more than two languages.[1] Sample questions we seek to answer are: Which features trigger the use or absence of articles in English? How do other languages differ in their article use? What about languages which do not use the concept of articles?

Though we focus on linguists as end-users who use our system to find evidence to answer research questions, the option of relating several layers of linguistics metadata in the form of annotations and alignments may facilitate other use cases, such as dictionary look-ups for words in context in more than one corresponding target language[2], detecting triggers for translation variants of particular expressions and syntactical structures, and comparing corresponding patterns such as word order preferences across multiple languages.[3]

In this paper, we will discuss three prominent challenges to be addressed in our work. Section 2 deals with the characteristics of multi-parallel alignments and outlines techniques to attain them. Section 3 describes the data structures required for our research questions and how to map them to a database schema. Section 4 discusses the requirements for user-friendly reporting of query results and suggests an approach for an expressive linguistic query language.

2 Multi-parallel Corpus Data Preparation

At present, several large, multi-parallel corpora are freely available. *Europarl* (Koehn 2005) and *MultiUN* (Eisele and Chen 2010), for instance, comprise millions of tokens in 21 and 6 languages, respectively. Östling (2015, p. 6) illustrates some of the multi-parallel corpora available in terms of language count and average number of words per language. These corpora consist of parallel documents corresponding to each other.[4]

Pairwise sentence alignment for a number n of languages covered by the respective corpus re-

[1] The definition of 'large' in the context of corpora may well be a controversial one. We argue that counting entities, such as tokens, sentences, etc. does not suffice for measuring the largeness of a corpus, but that the richness of relations described by its data model is equally important.

[2] This particularly addresses language learners who are proficient in other languages.

[3] For a discussion of the needs of different user groups see Volk, Graën, and Callegaro (2014).

[4] More specific alignment is implicitly given by speaker turns in the case of *Europarl* (see Graën, Batinic, and Volk 2014, p. 224).

sults in $\binom{n}{2}$ pairs sets of pairwise alignments since the correspondences of sentences are expressed as bidirectional alignments.

2.1 Multi-parallel Alignments

To address questions that involve more than two languages, pairwise sentence alignments pose a problem since combining several sets of pairwise alignments (again $\binom{n}{2}$ pairs for n languages) yields rather big graphs of sentences, moreover, alignment errors tend to propagate. This is depicted in Fig. 1 for 3 languages and 3 sets of pairwise alignments.

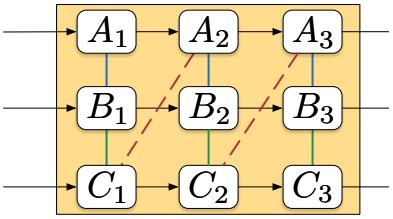


Figure 1: The alignment errors between language A and C (dashed lines) result in overly connected alignment graphs. The yellow box is the closure of all pairwise alignments.

Rather than closures of pairwise alignments, we require sets of corresponding sentences in all languages, denoting that all contained sentences mutually correspond to each other. We call such a set a **multi-parallel alignment** (MPA). MPAs may contain other MPAs, as depicted in Fig. 2, as long as these build a proper subset of the containing MPA.

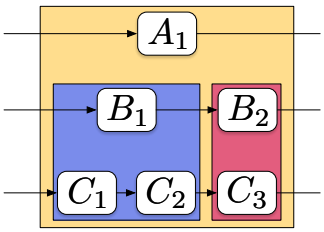


Figure 2: MPAs (coloured boxes) designate the elements of each language they extend over as corresponding.

The same problem applies to word alignment, once a multi-parallel sentence alignment has been found, where correspondence is usually calculated unidirectionally[5], which results in a set of $2 \times \binom{n}{2}$ unidirectional pairwise alignments. Several well-known algorithms exist to deduce a bidirectional word alignment from a pair of unidirectional ones[6], but they may result in a loss of valuable information for linguistic questions (Lehner, Graën, and Clematide 2015).

Analogous to the MPAs of sentences, different granularities of word correspondences can be expressed by nested MPAs as shown in Fig. 3.

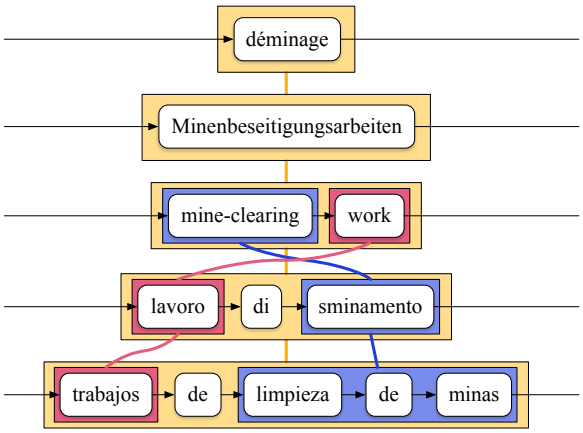


Figure 3: MPAs on a sub-sentential level, ranging from word to phrase alignment. Two MPAs with elements in three languages (red and blue) are contained by a broader MPA (yellow) which covers five languages.

2.2 Approaches for Attaining Multi-parallel Alignments

In order to obtain MPAs on a sentence level, we calculated the respective pairwise alignments for a total of five languages with *hunalign* (Varga, Halácsy, Kornai, Nagy, Németh, and Trón 2005) and combined the respective alignments in a graph as shown in Fig. 1. We then removed improbable links, i.e. those receiving less support from the other language pairs, by applying different heuristics which performed well for highly parallel texts. As soon as the translations became loose, our algorithms were unable to make good decisions.

In our opinion, this problem arises because, after the pairwise alignment step, alternative alignment scores get lost, and only the solution maximising the overall alignment score of the particular pair of texts is returned. A multi-parallel alignment performed on a joint alignment score is supposed to yield better, and a priori consistent, results. As the

[5] That means that a token t_a of language A being aligned with a token t_b of language B does not imply a reverse alignment between t_b and t_a.

[6] This process is called symmetrisation (Liang, Taskar, and Klein 2006; Tiedemann 2011, pp. 75–77).

costs of calculating scores for all possible alignment options grows quadratically, both time-wise and memory-wise, with the number of languages involved, an exhaustive search is not feasible. Instead, we are working on an approximate dynamic programming approach (Powell 2007).

To compute the MPAs on a word level, we plan to implement an algorithm similar to the one used for multilingual sentence alignment. We expect the complexity of this task to be considerably higher mainly since **(a)** sentences comprise more words than a textual unit contains sentences[7], **(b)** the constraint of sequentiality does not hold for words between multi-parallel aligned sentences, and **(c)** based on our previous investigations, we expect the word alignment ratio to vary strongly across languages.[8]

Bilingual alignment algorithms for phrase-structure parses have been reported by Zhechev and Way (2008). We plan to adapt their approach to our multi-lingual dependency parses.

3 Efficient Representation of Multi-parallel Corpora in an RDBMS

We expect our corpus compilation to be an aid in answering complex cross-linguistic questions by means of correlating different linguistically motivated data layers on a large scale of data. We identified the following eligible layers: sentence segmentation, tokenisation, lemmatisation, part-of-speech tagging, chunking, syntactical dependency parsing, coreference resolution (based on parse trees), sentence alignment, word alignment and sub-sentential alignment.

There are several NLP tools available for each layer. We allow for multiple annotation and alignment layers of the same kind, e.g. dependency parses by different parsers, with the exception of sentence and token segmentation where we commit to a single layer of primary data.[9] Apart from the primary data, each of these layers is based on at least one other layer such that the layer dependencies form a directed acyclic graph. In this vein, we know which dependent layers to recreate once a particular layer is rebuilt. In contrast to Bański, Fischer, Frick, Ketzan, Kupietz, Schnober, Schonefeld, and Witt (2012, p. 2906), we do not require query results to be reproducible after such layer rebuilds.

3.1 Data Types Required for the Representation of Linguistic Data Layers

In our considerations of the data structure required for building a conceptual data model incorporating those respective layers (and potential future ones), we identified three abstract data types which can be composed in such a way that all our requirements are met:

1. an interval on sequential elements,
2. a directed binary relation between two elements of the same type and
3. an undirected relation between several elements of the same type.

Each of these types, as well as a basic one without further definitions, may comprise any number of attributes such as labels, confidence scores, etc.

Tokens are basic elements and have attributes like their surface form, lemmas, and part-of-speech tags. Chunks are represented as intervals on tokens, dependency relations and unidirectional word alignments as relations between two tokens. Finally, the most complex type, n-ary relations between sets of elements, is needed for modelling MPAs[10], as well as for the modelling of coreference chains for instance.

3.2 Deriving a Database Schema from the Data Model

Corpus query systems are optimised for efficient retrieval rather than for processing new data, as the underlying linguistic data typically does not change. Richly annotated and aligned corpora allow for considerably more sophisticated corpus queries and thus require an efficient way to retrieve data in a less restricted fashion.

In times of freely available, advanced relational database management systems (RDBMS) which target flexible and efficient retrieval of large amounts of arbitrary structured data, building an own storage and retrieval system from scratch seems pointless (Davies 2005).

The limitation to the three described abstract data types allows us to define a translation pattern for the conversion of the data model into a

[7] In *Europarl*, a sentence contains three times more words on average than a textual unit contains sentences.

[8] As Fig. 3 illustrates, a ratio of 1:5 is not uncommon for aligned complex noun phrases, whereas ratios of 1:3 or more in sentence alignments are rare ($< 1\%$).

[9] Chiarcos, Ritz, and Stede (2009) discuss problems that arise with multiple tokenisation layers.

[10] In Fig. 3, these relations are expressed by connecting lines between sets of words in each language.

relational database schema, including normalisation, indices and access functions as stored procedures. Moreover, snippets for the retrieval of the particular data types can also be compiled uniformly based on the data model. As a further advantage, our RDBMS, PostgreSQL[11], includes an advanced query optimiser whose goal is defined to determine the most efficient query plan by rewriting a given query (see Momjian 2015).

4 User-friendly Reporting and Flexible Querying

Our third challenge involves two aspects:

1. How can we flexibly report user-friendly query results?
2. What is needed to enable contrastive corpus linguists, who are generally non-experts in *SQL*, to formulate their information needs more naturally in an expressive linguistic query language?[12]

4.1 User-friendly Reporting of Query Results

For the use case of cross-lingual frequency distributions of translations illustrated by example sentences, a simple form-based query menu is probably adequate. The user input, for instance, word or base forms including part-of-speech filters, can be easily interpolated into handcrafted *SQL* templates.

Applying such queries to large corpora is likely to yield large amounts of search hits. A practical challenge for the usability of such a system lies in the proper selection of sentences that are delivered to the end user as relevant and informative examples. This is an instance of the *Good Dictionary Example Extractor* problem (Kilgarriff, Husák, McAdam, Rundell, and Rychlỳ 2008), termed *GDEX* in the context of the *Sketch Engine* (Kilgarriff, Baisa, Bušta, Jakubíček, Kovář, Michelfeit, Rychlỳ, and Suchomel 2014).

Kosem, Husak, and McCarthy (2011) discuss many textual features (sentence lengths, punctuation, frequency thresholds on words, anaphoric expressions, etc.) that must be statistically evaluated for such a task. Our RDBMS includes the option to use *R* as an embedded statistical programming language, which we expect to be sufficient for our needs.

[11] `http://www.postgresql.org/`

[12] These linguists typically have varying research questions and a strong need for flexible and precise queries.

Furthermore, statistical evaluation of result sets, for instance, across different language pairs, could be provided given the ability to statistically analyse query results.

4.2 An Expressive Linguistic Query Language for Our Data Model

SQL allows the user to flexibly query every aspect of our data model, that is, every entity, attribute, relation and Boolean combination thereof. However, native *SQL* queries for our highly interconnected and normalised data structures are not an appropriate abstraction level for linguists; they cannot express their linguistic information needs in a natural way.

Therefore, there is a need for an expressive linguistic query language to flexibly describe the constructions contrastive linguists are interested in. Two important strains of linguistic query systems have been developed in the past:

1. Corpus linguistics tools for text corpora such as *CQP* (Christ 1994) and
2. treebank query tools such as *TIGERSearch* (König, Lezius, and Voormann 2003).

CQP supports annotated words, structural boundaries (sentences, constituents), and sentence-aligned parallel texts right from the beginning. For instance, a query for the word *car* in the English part of *Europarl* may be restricted to the co-occurrence of the German word *Auto* in the aligned sentence using the `within` operator:

```
[word="car"] within europarl7_de: [word="Auto"]
```

Although useful, this is not the level of expressiveness we have in mind.

In recent years, treebank query systems have been generalised in various ways. The *Stockholm Treealigner* (Lundborg, Marek, Mettler, and Volk 2007) introduced an operator for querying alignments between words or phrases of bilingual treebanks, freely combinable with precise monolingual *TIGERSearch*-like queries for syntactic structures. The *ANNIS* platform (Zeldes, Lüdeling, Ritz, and Chiarcos 2009) with its query language *AQL* for multi-level graph-based annotations offers operators for dependency relations, inclusion or overlap of token spans, and namespaces for annotations of the same type produced by different tools (for instance, the output of different dependency parsers, see also section 3). Our proposed

linguistic query language will include these operators and follow the logic-based style of this language family.[13] The next step in our work is therefore a translation of *Treealigner*/*AQL*-style queries into native *SQL* queries for our database. Rosenfeld (2010) describes the translation of *AQL* into *SQL*, which in turn is inspired by the implementation of the *DDDQuery* language (Faulstich, Leser, and Vitt 2006), an extended XPath query language for linguistic data.

Lai and Bird (2010) discuss the formal expressiveness of linguistic query languages and mention the known inherent limitation of *AQL*-style query languages to the fragment of existential first-order logic, which does not support queries for missing constituents. Recently, we proposed an approach where the result sets of several *AQL*-style queries are subtracted in order to identify configurations with missing constituents (Clematide 2015).

5 Conclusions

We identified three of the most prominent issues that we face building a system for querying large multi-parallel corpora with several inter-connected layers of linguistic information.

Typically, alignments have been calculated pairwise. Multi-parallel alignments, as we call the mutual correspondence relation between sets of elements of multiple languages, demand new, innovative approaches. Once the annotation and alignment data has been obtained, we need to store this complex accumulation in a fashion that supports efficient retrieval from multiple layers. Hence, we argue for the use of a relational database. We built a data model upon three abstract data types which incorporates the data structures of the aforementioned layers and allows for a direct translation into a database schema.

Having set up a database comprising multi-parallel corpus data with several layers of annotation and alignment, our intended end user requires a means to access said information in a convenient way. We sketched a flexible, yet user-friendly query language to deal with any kind of data layers defined within the data model whose queries can be mapped to *SQL* queries and thereupon processed by the database. On this basis, we discussed varying requirements regarding the presentation of query results (reporting), ranging from a selection of prototypical exemplars to an automatic statistical evaluation.

[13] We are aware of alternatives, for instance, *XPath*-style query languages such as LPath (Lai and Bird 2010) or PML-TQ (Štěpánek and Pajas 2010).

Acknowledgment

Thanks to Martin Volk for discussions and Rahel Oppliger for proof reading. This research was supported by the Swiss National Science Foundation under grant 105215_146781/1 through the project "SPARCLING – Large-scale Annotation and Alignment of Parallel Corpora for the Investigation of Linguistic Variation".

References

Bański, Piotr, Peter M Fischer, Elena Frick, Erik Ketzan, Marc Kupietz, Carsten Schnober, Oliver Schonefeld, and Andreas Witt (2012). "The new IDS corpus analysis platform: Challenges and prospects". In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*, pp. 2905–2911.

Chiarcos, Christian, Julia Ritz, and Manfred Stede (2009). "By all these lovely tokens... Merging Conflicting Tokenizations". In: *Proceedings of the Third Linguistic Annotation Workshop*, pp. 35–43.

Christ, Oliver (1994). "A modular and flexible architecture for an integrated corpus query system". In: *Proceedings of COMPLEX'94: 3rd Conference on Computational Lexicography and Text Research*. (Budapest), pp. 23–32.

Clematide, Simon (2015). "Reflections and a Proposal for a Query and Reporting Language for Richly Annotated Multiparallel Corpora". In: *Proceedings of the Workshop on Innovative Corpus Query and Visualization Tools*. (Vilnius). Nordic Conference of Computational Linguistics (NODALIDA), pp. 6–16.

Davies, Mark (2005). "The advantage of using relational databases for large corpora: Speed, advanced queries and unlimited annotation". In: *International Journal of Corpus Linguistics* 10.3, pp. 307–334.

Eisele, Andreas and Yu Chen (2010). "MultiUN: A Multilingual Corpus from United Nation Documents." In: *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*. (Valletta). European Language Resources Association (ELRA), pp. 2868–2872.

Faulstich, Lukas C., Ulf Leser, and Thorsten Vitt (2006). "Implementing a linguistic query language for historic texts". In: *Current Trends in Database Technology – EDBT 2006*. Springer, pp. 601–612.

Graën, Johannes, Dolores Batinic, and Martin Volk (2014). "Cleaning the Europarl Corpus for Linguistic Applications". In: *Proceedings of the 12th KONVENS*. (Hildesheim), pp. 222–227.

Kilgarriff, Adam, Vít Baisa, Jan Bušta, Miloš Jakubíček, Vojtěch Kovář, Jan Michelfeit, Pavel Rychlỳ, and Vít Suchomel (2014). "The Sketch Engine: ten years on". In: *Lexicography* 1.1, pp. 7–36.

Kilgarriff, Adam, Milos Husák, Katy McAdam, Michael Rundell, and Pavel Rychlỳ (2008). "GDEX: Automatically finding good dictionary examples in a corpus". In: *Proceedings of the 13th EURALEX International Congress*. (Barcelona).

Koehn, Philipp (2005). "Europarl: A parallel corpus for statistical machine translation". In: *Machine Translation Summit*. (Phuket). Asia-Pacific Association for Machine Translation (AAMT), pp. 79–86.

König, Esther, Wolfgang Lezius, and Holger Voormann (2003). *TIGERSearch 2.1 – User's Manual*. Institute for Natural Language Processing, University of Stuttgart.

Kosem, Iztok, Milos Husak, and Diana McCarthy (2011). "GDEX for Slovene". In: *Electronic lexicography in the 21st century: New Applications for New Users: Proceedings of eLex 2011*. (Bled), pp. 151–159.

Lai, Catherine and Steven Bird (2010). "Querying linguistic trees". In: *Journal of Logic, Language and Information* 19.1, pp. 53–73.

Lehner, Stéphanie, Johannes Graën, and Simon Clematide (2015). *Compound Alignment Gold Standard (COMPAL GS)*. URL: `http://pub.cl.uzh.ch/purl/compal_gs` (visited on May 29, 2015).

Liang, Percy, Ben Taskar, and Dan Klein (2006). "Alignment by agreement". In: *Proceedings of the Main Conference on Human Language Technology Conference (HLT-NAACL)*. (New York). Association for Computational Linguistics (ACL), pp. 104–111.

Lundborg, Joakim, Torsten Marek, Maël Mettler, and Martin Volk (2007). "Using the Stockholm TreeAligner". In: *Proceedings of the 6th Workshop on Treebanks and Linguistic Theories*. (Bergen), pp. 73–78.

Momjian, Bruce (2015). *Explaining the Postgres Query Optimizer*. URL: `http://momjian.us/main/writings/pgsql/optimizer.pdf` (visited on May 29, 2015).

Östling, Robert (2015). *Bayesian Models for Multilingual Word Alignment*. Department of Linguistics, Stockholm University.

Powell, Warren B (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality*. Vol. 703. John Wiley & Sons.

Rosenfeld, Viktor (2010). *An implementation of the Annis 2 query language*. Tech. rep.

Štěpánek, Jan and Petr Pajas (2010). "Querying Diverse Treebanks in a Uniform Way". In: *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*.

Tiedemann, Jörg (2011). "Bitext Alignment". In: *Synthesis Lectures on Human Language Technologies* 4.2, pp. 1–165.

Varga, Dániel, Péter Halácsy, András Kornai, Viktor Nagy, László Németh, and Viktor Trón (2005). "Parallel corpora for medium density languages". In: *Proceedings of the Recent Advances in Natural Language Processing (RANLP)*. (Borovets), pp. 590–596.

Volk, Martin, Johannes Graën, and Elena Callegaro (2014). "Innovations in Parallel Corpus Search Tools". In: *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*. (Reykjavik), pp. 3172–3178.

Zeldes, Amir, Anke Lüdeling, Julia Ritz, and Christian Chiarcos (2009). "ANNIS: A search tool for multi-layer annotated corpora". In: *Proceedings of Corpus Linguistics*. (Liverpool).

Zhechev, Ventsislav and Andy Way (2008). "Automatic generation of parallel treebanks". In: *Proceedings of the 22nd International Conference on Computational Linguistics (Coling)*. Vol. 1, pp. 1105–1112.

Ziggurat: A new data model and indexing format for large annotated text corpora

Stefan Evert
Friedrich-Alexander-Universität Erlangen-Nürnberg
stefan.evert@fau.de

Andrew Hardie
Lancaster University
a.hardie@lancaster.ac.uk

Abstract

The IMS Open Corpus Workbench (CWB) software currently uses a simple tabular data model with proven limitations. We outline and justify the need for a new data model to underlie the next major version of CWB. This data model, dubbed *Ziggurat*, defines a series of types of *data layer* to represent different structures and relations within an annotated corpus; each such layer may contain variables of different types. Ziggurat will allow us to gradually extend and enhance CWB's existing CQP-syntax for corpus queries, and also make possible more radical departures relative not only to the current version of CWB but also to other contemporary corpus-analysis software.

1 Introduction

With recent technological advances, it has become possible – and increasingly practical – to compile huge corpora (of 10 billion tokens and more) with complex linguistic annotation (token-level annotation such as part-of-speech tags, lemmatization, semantic tags; logical and typographical text markup encoded by XML tags; phrase structure trees; syntactic dependency graphs; coreference chains; …) and rich metadata (at text, paragraph or speaker level). At the same time, emerging international standards have begun to account for such richly annotated corpora – defining data models and serialization formats, as in the Linguistic Annotation Framework (LAF, ISO 24612: Ide & Suderman 2014); as well as different levels of query languages for complex linguistic annotations, as in the Corpus Query Lingua Franca (CQLF, ISO/CD 24623-1). Defined in a (currently draft) ISO standard, the CQLF metamodel distinguishes three levels of analysis, which correspond to linguistic annotations of different complexity:

- Level 1: plain-text search and token-level annotations
- Level 2: hierarchical structures and dependency graphs
- Level 3: multiple concurrent annotations

The current generation of software tools for querying large corpora – such as the IMS Open Corpus Workbench (CWB: Evert & Hardie 2011), Manatee/SketchEngine (Rychlý 2007) and Poliqarp (Janus & Przepiórkowski 2007) – are still based on a simple tabular data model that corresponds to CQLF Level 1 and was developed in the 1990s (Witten et al. 1999). This data model represents a text corpus as a sequence of tokens annotated with linguistic features coded as string values. It is equivalent to a data table where rows correspond to tokens and columns to the different annotations – similar to a relational database table, but with an inherent ordering of the rows.

This tabular data model was applied to linguistic corpus indexing by the first release of CWB (Christ 1994). CWB also extended the basic text-indexing structure outlined by authors such as Witten et al., by adding special provisions for simple structural annotation and sentence alignment. These were stored in the form of token ranges (pairs of integer corpus positions). The approach pioneered by the early versions of CWB was later embraced by many other software packages, including those cited above. The current release of CWB and its *Corpus Query Processor* (CQP), that is version 3, is widely used, especially through the user-friendly, browser-based CQPweb interface (Hardie 2012);

it still builds on the same data model and maintains full backwards compatibility. Though the data model has no "official" name, we will refer to it in this paper as the *CWB3 data model*. As lead maintainers and developers of CWB – now an open-source project – we have become increasingly acutely aware of a number of limitations in CWB's basic design. In addition to its simplistic data model, CWB3 is limited to corpora of at most 2.1 billion tokens, because it stores token positions as signed 32-bit. This design decision, while perhaps justifiable in the early 1990s, no longer makes real sense (as explained in Evert & Hardie 2011).

A number of indexing and query tools do in fact go beyond a data model parallel to CWB3, and can thus support more complex linguistic annotation. Examples include TIGERSearch, (Lezius 2002), ANNIS (Zeldes et al. 2009), and ICECUP (Quinn & Porter 1994). However, such software is usually designed for small, manually annotated data sets and fails to scale up to billion-word corpora harvested from the Web or other sources. This tendency is well-exemplified by ICECUP, which is distributed alongside the corpora it is intended to be used with, namely ICE-GB and the Diachronic Corpus of Present-Day spoken English (DCPSE), densely-annotated corpora on the order of one million tokens in extent.

There is an urgent need, therefore, for efficient corpus query tools that go beyond the limitations of the CWB3 data model, providing compact storage and efficient search over complex linguistic structures. The work of the CWB development team over the past two years has turned to the development of a new data model that can support complex annotation, and can do so at scale.

2 Introducing Ziggurat

We present a novel data model, and associated indexing format, which will underlie the next major version of CWB (version 4). Rather than refer to this as the "CWB4" model, we propose the name *Ziggurat* for the data model, the file format, and the database engine software that implements them. The name is inspired by the shape of the data model, which – as the remainder of this paper will illustrate – consists conceptually of a pile of rectangular layers on top of one another.

The design goals of Ziggurat are that it should (i) scale to corpora of arbitrary size; (ii) support rich linguistic annotation, in particular XML hierarchies, phrase-structure trees, dependency graphs and parallel-corpus alignment; and (iii) provide efficient indexed access to the data, enabling complex linguistic queries in reasonable time. In the long term, by defining the Ziggurat engine as a conceptually-separate entity to the CWB software and the query language that it provides (known as *CQP-syntax*), our aim is to be able to use Ziggurat as the underpinning for more than one (kind of) query language. Towards the end of this paper, we will speculate on the new types of query languages that the enriched data model supported by Ziggurat will enable. Let us first, however, survey some related work, justifying the need for a new database engine.

3 Related work and motivation

In recent years, researchers have explored several alternative approaches to efficient queries for large text corpora:

- A standard relational database with redundant representation of the corpus (e.g. n-gram tables), a large number of indexes and fine-tuning of the database server and SQL queries (as outlined by Davies 2005, although Davies' current architecture[1] is much-revised from this now somewhat outdated outline). It is unclear whether this approach can be generalized to more complex linguistic data structures and sophisticated query needs.
- A native XML or graph database used off-the-shelf, with built-in indexing and query facilities. Mayo et al. (2006) show that this approach is inefficient using XML databases; Proisl & Uhrig (2012) make the same observation for a popular graph database.
- An information retrieval or Web search engine such as Lucene, with custom modifications to support linguistic annotation and the kinds of query patterns supported by CQP-syntax. A recent example of this approach is the BlackLab[2] software. While it is difficult to assess the potential of the system due to a lack of scientific publications, a small number of blog posts about its internals suggest that it may be very

[1] Accessible at `http://corpus.byu.edu`
[2] `https://github.com/INL/BlackLab`

difficult to extend BlackLab to full tree structures and dependency graphs.

- *Corpuscle* (Meurer 2012) proposes new indexing structures based on suffix trees in order to optimise the performance of regular expressions and CQP-syntax queries. Having a focus on indexing and query algorithms, it does not attempt to go beyond the tabular CWB3 data model.

Despite introducing various innovations, none of these approaches has resorted to a ground-up rethink of the data model: all attempt to extend some existing data model. While such efforts have had notable short-term successes, we believe that ultimately they are self-limiting, for the reasons discussed above. We are convinced that it is necessary to go beyond the CWB3 data model; however, we are likewise convinced that working around other standard data models, whether those of XML databases or web-query engines, is not the best way to do it, especially for a community-driven effort with limited resources. This motivates our proposal of Ziggurat.

Ziggurat *does* represent a ground-up rethink of the CWB3 data model, keeping its basic idea – a tabular data model with implicitly-ordered rows representing sequence positions – but extending it considerably, and like CWB3 using custom index structures and file formats. We believe that this offers better support for the highly successful brute-force corpus search of CWB and similar query tools than a standard off-the-shelf backend such as a SQL RDBMS or Web search engine. Recognizing that it is better to have a simple but flexible tool that is available, well-maintained and actively developed by its user community than to design the "Perl 6" of corpus query engines – that is, a perfect redesign which remains unreleased and unavailable to most users for years on end – we resolved to keep the data model, index structures and file formats as simple and straightforward as possible. Thus, the entire Ziggurat data model builds on a small set of easily implemented data structures.

Further key requirements for the new data model are (i) full Unicode support, (ii) (nigh-)unlimited corpus size, (iii) logical backward compatibility with the CWB3 data model, (iv) full support for hierarchical XML annotation and other tree structures, (v) representation of dependency graphs, (vi) support for sentence (and preferably also word) alignment, and (vii) concurrent annotation layers forming independent or intersecting hierarchies. The Ziggurat data model thus encompasses all three levels of the CQLF metamodel.

4 The data model

In order to ensure a compact representation, efficient access and a simple implementation of the data model, a number of limitations are accepted:

- Corpora are "horizontally" static, i.e. no modification of the tokenization, annotation units or annotated values is allowed in an indexed corpus, and documents can neither be added nor deleted. However, corpora are "vertically" flexible, i.e. individual annotated features or entire annotation layers may be added and deleted.
- Individual physical corpora cannot be collected into a single "virtual" corpus, but queries can be restricted to subsets of a large physical corpus without loss of efficiency.
- The data format is token-based, without support for full-text representation and search.

In the proposed data model, a corpus is a collection of sequential *data layers*, which are connected into one or more annotation hierarchies over the primary text data. Each data layer consists of a sequence of annotation units annotated with one or more variables (i.e. linguistic features). Thus, a data layer in Ziggurat fundamentally has the same tabular format as the annotated token sequence in a CWB3 corpus, and the established representation and indexing approaches for such data structures (similar to Witten et al. 1999) can be used. A key difference between Ziggurat and CWB3 is that all Ziggurat data layers can be annotated with variables, not just the primary token sequence. Moreover, unlike CWB3, Ziggurat will support different *types* of variables:

- *Indexed strings* = string values where all distinct strings are collected in a lexicon and associated with numeric IDs (equivalent to CWB3 token-level annotations)
- *Raw strings* = string values stored without indexing, mainly used for free-form metadata (such as URLs) or unique IDs
- *Integers* = signed 64-bit integer values (which can also be interpreted by client software as fixed-point decimals), used for storing numeric information

- *Pointers* = references to a single parent annotation unit in the same layer, which can be used to structure the sequence of annotation units into a forest of unordered trees (e.g. a simple dependency parse without multiple parents); these will be stored as integers, and thus the maximum corpus size will be the positive limit of a 64-bit signed integer (somewhat over 9.2 quintillion)
- *Hashes* = indexed key-value stores with a lexicon similar to indexed strings, useful for storing variable metadata and the attributes of XML start tags.

Structural information is conveyed by the way in which different data layers are connected. In a Ziggurat index, a basic token sequence together with all token-level annotations forms the so-called *primary annotation layer*. All other types of data layers reference one or more *base layers*. These layers can in turn act as base layers of further data layers, forming a hierarchy of annotation layers. (This is the source of the name *Ziggurat*: the multiple rectangular data layers that are built on top of one another may be visualized in a shape reminiscent of a Mesopotamian ziggurat.)

Annotations are fully concurrent, allowing multiple independent or intersecting annotation hierarchies over the primary layer. In principle, a corpus may also contain multiple *primary* layers, e.g. representing different transcriptions of the same audio signal.

Ziggurat will have the following types of data layers (see appendix for an illustration):

- *Segmentation layer*: Each unit represents an uninterrupted range of base layer units (usually the tokens of a primary layer). Different ranges may neither overlap nor be nested within each other. This layer type extends the structural attributes used to represent multi-token structures in the CWB3 data model, but more flexibly; these layers are useful for storing a simple segmentation of the corpus (into sentences, texts, files, speaker turns, …) and the associated metadata.
- *Tree layer*: Each unit also represents an uninterrupted range of base layer units, but these ranges may be nested hierarchically, forming an ordered tree over the base layer sequence. An important application of tree layers is to represent XML annotation, with each annotation unit corresponding to one XML element. Empty ranges are expressly allowed by the data model for this purpose. Tree layers can also, however, represent the tree structures of constituency-parsing.
- *Graph layer*: Each unit represents a directed edge between two annotation units in the base layer, thus forming a directed graph over the base layer, where both edges (in the tree layer) and nodes (in the base layer) may be annotated with variables. Unlike other layers, graph layers may have two different base layers for the tails and heads of the edges. A graph between two different base layers represents an alignment of the base layers: a sentence alignment if they are sentence segmentation layers, or a word alignment if they are primary layers. This type of layer thus supports both dependency-parsing annotation (with a single base layer) and parallel-corpus alignment (with two base layers: the equivalent of a CWB3 alignment-attribute).

Ziggurat data structures are designed to be as simple and uniform as possible. The only value types are strings in UTF-8 encoding and signed 64-bit integers. Indexing is based on two simple generic structures: a sort index with integer sort keys, and a postings list similar to that used by Web search engines. The Ziggurat file formats are also simplified relative to CWB3, trading off compactness for simplicity and decompression speed. CWB3 uses bit-oriented Huffman and Golomb coding schemes, as proposed by Witten et al. (1999). However, through experiments using CQP we have found that these compression methods, though maximally economical of disk space, require an excessive amount of processor time when the system is running complex queries. Ziggurat instead utilizes variable-length byte encodings (without a codebook) and delta compression. A Ziggurat-encoded corpus will therefore take up more disk space, but will require less CPU time to decompress.

5 New corpus query approaches

The Ziggurat data model's greater expressiveness relative to CWB3 will allow, and therefore ultimately call for, more sophisticated query languages than CWB3 could support. While a concrete specification is not possible at this time, we

believe that the following three approaches are promising.

Approach 1 extends the CWB3-style "linear" queries based on regular expression notation, i.e. the kind of query language typified by CQP-syntax. It allows query paths to follow other axes than the token sequence (similar to XPath), in particular along the edges of a graph layer and to parents, children and siblings in a tree layer. Experience from Treebank.info (Proisl & Uhrig 2012) suggests that many linguistically plausible searches can be flattened into a single linear path; otherwise "branching" queries will be needed. This approach will be implemented in version 4 of CWB – the first application using Ziggurat. CWB version 4 will at first simply implement the existing CQP-syntax in terms of calls to the Ziggurat engine; but subsequently it will gradually extend the CQP-syntax query language over time to exploit more of the affordances of Ziggurat.

In Approach 2, a query specifies a finite set of anchor points (tokens or annotation units from a specified data layer), constraints on annotated variables, and relations between different anchors (such as co-occurrence, dominance or precedence). Similar to XQuery, this approach is used by many existing query engines for CQLF levels 2 and 3, including TIGERSearch, ANNIS (Krause & Zeldes in press) and the NXT Query Language (Evert & Voormann 2003).

Approach 3 derives from the following observation by Geoffrey Sampson:

> […] there are usually two possibilities when one wants to exploit corpus data. Often, one wants to put very obvious and simple questions to the corpus; in that case, it is usually possible to get answers via general-purpose Unix commands like grep and wc, avoiding the overhead of learning special-purpose software. Sometimes, the questions one wants to put are original and un-obvious; *in those cases, the developer of a corpus utility is unlikely to have anticipated that anyone might want to ask them, so one has to write one's own program to extract the information.* (Sampson 1998:365; our emphasis).

The most sophisticated corpus query requirements can only be satisfied by a Turing-complete query language. We therefore envisage corpus queries as programs for a virtual machine (VM) that interfaces closely with the corpus data model and index structures. High-level languages (such as JavaScript, Python or Lua) or parser generators can then be used to implement various simplified query languages with relative ease, compiling the queries written in these query languages into VM programs. This approach, then, ultimately will enable "power users" – those with an understanding of the data model and some coding ability – to write their own programs to carry out virtually every imaginable search.

By making the Ziggurat data model and database engine extremely flexible in the ways outlined above, we will establish a foundation on which any or all of these three approaches can be developed, within the same or different pieces of software.

References

Christ, Oliver (1994). A modular and flexible architecture for an integrated corpus query system. In *Papers in Computational Lexicography (COMPLEX '94)*, pages 22–32, Budapest, Hungary.

Davies, Mark (2005). The advantage of using relational databases for large corpora: Speed, advanced queries and unlimited annotation. *International Journal of Corpus Linguistics*, 10(3), 307–334.

Evert, Stefan and Hardie, Andrew (2011). Twenty-first century corpus workbench: Updating a query architecture for the new millennium. In *Proceedings of the Corpus Linguistics 2011 Conference*, Birmingham, UK.

Evert, Stefan and Voormann, Holger (2003). NQL – a query language for multi-modal language data. Technical report, IMS, University of Stuttgart. Version 2.1.

Hardie, Andrew (2012). CQPweb – combining power, flexibility and usability in a corpus analysis tool. *International Journal of Corpus Linguistics*, 17(3), 380–409.

Ide, Nancy and Suderman, Keith (2014). The linguistic annotation framework: a standard for annotation interchange and merging. *Language Resources and Evaluation*, 48(3), 395–418.

ISO 24612 (2012) Language resource management – linguistic annotation framework. Technical report, ISO.

ISO/CD 24623-1 (2014). Language resource management – corpus query lingua franca (CQLF) – part 1: Metamodel. Technical report, ISO.

Janus, Daniel and Przepiórkowski, Adam (2007). Poliqarp: An open source corpus indexer and search engine with syntactic extensions. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, Posters and Demonstrations Sessions*, pages 85–88, Prague, Czech Republic. Association for Computational Linguistics.

Krause, Thomas and Zeldes, Amir (in press). ANNIS3: A new architecture for generic corpus query and visualization. *Digital Scholarship in the Humanities*. Advance access.

Lezius, Wolfgang (2002). TIGERSearch – ein Suchwerkzeug für Baumbanken. In S. Busemann (ed.), Proceedings der 6. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS 2002), Saarbrücken, Germany.

Mayo, Neil; Kilgour, Jonathan; Carletta, Jean (2006). Towards an alternative implementation of NXT's query language via XQuery. In *Proceedings of the 5th Workshop on NLP and XML (NLPXML-2006)*, pages 27–34, Trento, Italy.

Meurer, Paul (2012). Corpuscle – a new corpus management platform for annotated corpora. In *Exloring Newspaper Language: Using the web to create and investigate a large corpus of modern Norwegian*, number 49 in Studies in Corpus Linguistics. John Benjamins.

Proisl, Thomas and Uhrig, Peter (2012). Efficient dependency graph matching with the IMS open corpus workbench. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC '12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Quinn, Akiva and Porter, Nick (1994). Investigating English usage with ICECUP. *English Today*, 10(3), 19–24.

Rychlý, Pavel (2007). Manatee/Bonito - a modular corpus manager. In *Proceedings of the 1st Workshop on Recent Advances in Slavonic Natural Language Processing*, pages 65–70, Brno. Masaryk University.

Sampson, Geoffrey (1998). Review of Sidney Greenbaum (ed.), *Comparing English worldwide: The international corpus of English*. Oxford: Clarendon Press, 1996. ISBN 0-19-823582-8, xvi+286 pages. *Natural Language Engineering*, 4, 363–382.

Witten, Ian H.; Moffat, Alistair; Bell, Timothy C. (1999). *Managing Gigabytes*. Morgan Kaufmann Publishing, San Francisco, 2nd edition.

Zeldes, Amir; Ritz, Julia; Lüdeling, Anke; Chiarcos, Christian (2009). ANNIS: A search tool for multi-layer annotated corpora. In M. Mahlberg, V. González-Díaz, and C. Smith (eds.), *Proceedings of the Corpus Linguistics 2009 Conference*, Liverpool, UK. Article #358.

Appendix: Illustrations of different Ziggurat layer types

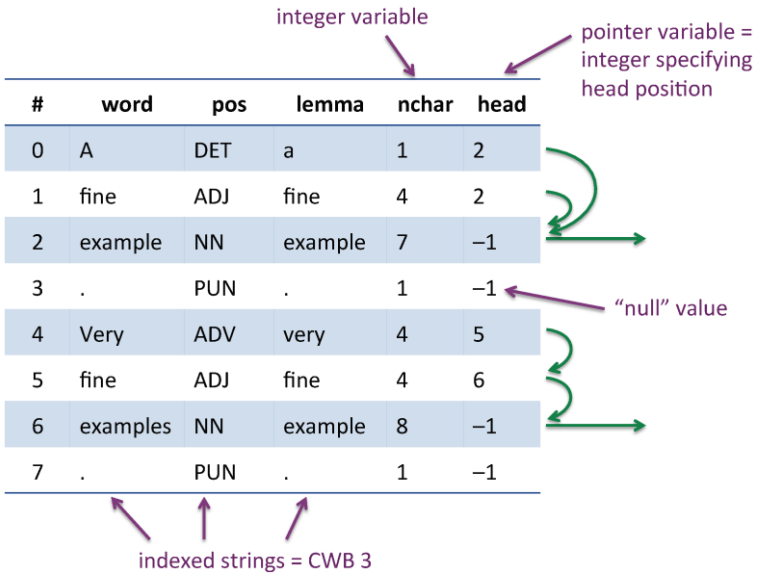


Fig 1. Illustration of different types of Ziggurat variables on a primary layer.
(Note that the simple tree structures defined by the pointer variable in the last column are less general than the graph layer in Fig. 2 and edges cannot be annotated with labels)

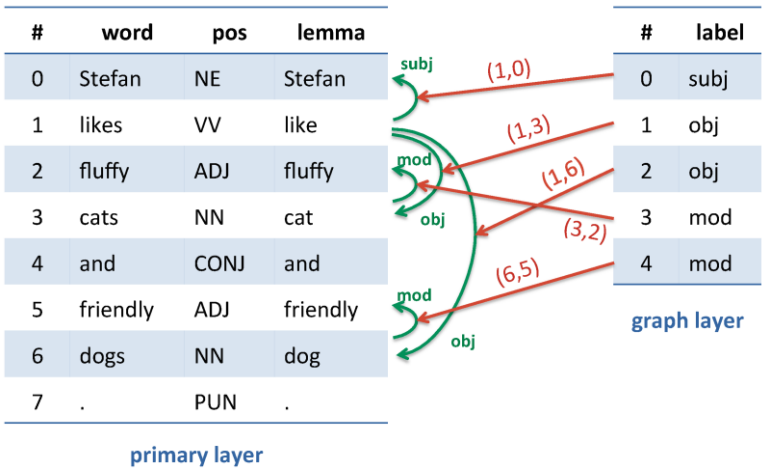


Fig 2. A graph layer (representing a dependency parse) and its base layer

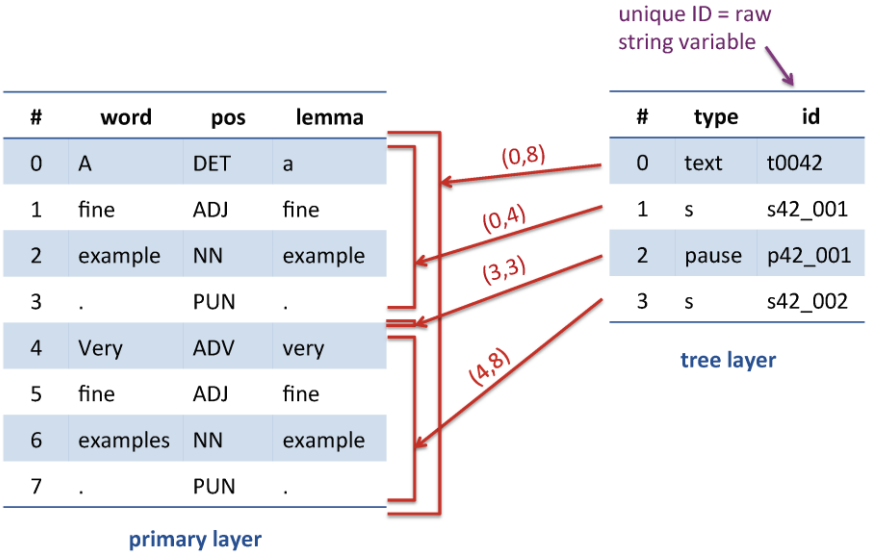


Fig 3. A tree layer (representing an XML hierarchy) and its base layer

Processing and querying large web corpora with the COW14 architecture

Roland Schäfer
Linguistic Web Characterization (DFG)
Freie Universität Berlin
roland.schaefer@fu-berlin.de

Abstract

In this paper, I present the COW14 tool chain, which comprises a web corpus creation tool called *texrex*, wrappers for existing linguistic annotation tools as well as an online query software called Colibri[2]. By detailed descriptions of the implementation and systematic evaluations of the performance of the software on different types of systems, I show that the COW14 architecture is capable of handling the creation of corpora of up to at least 100 billion tokens. I also introduce our running demo system which currently serves corpora of up to roughly 20 billion tokens in Dutch, English, French, German, Spanish, and Swedish.

1 Introduction

Large web corpora for empirical linguistic research have been available for over a decade (Kilgarriff and Grefenstette, 2003; Biemann et al., 2007; Baroni et al., 2009; Schäfer and Bildhauer, 2013; Biemann et al., 2013). Such corpora are an attractive complement to traditionally compiled corpora because they are very large, and they contain a lot of recent non-standard variation. Conceptual problems with web corpora may arise due to biases in the composition of crawled corpora (Schäfer and Bildhauer, 2013, Chapter 2), biases due to radical and undocumented cleaning procedures, and a lower quality of linguistic annotation (Giesbrecht and Evert, 2009). Major technical difficulties come from the fact that the creation of very large web corpora requires efficient preprocessing and annotation tools, necessarily using some type of parallelization. Also, for such corpora to be usable in an efficient way for linguists, intuitive and responsive interfaces have to be made available which abstract away from corpora which are partitioned or sharded across several machines. For most linguists, downloading gigabytes of data and running their own instances of corpus query tools on partitioned corpora is simply not an option.

In this paper, I introduce the COW14 ("Corpora from the Web") web corpus creation and query architecture (which is the second generation, following COW12) created as joint work with Felix Bildhauer at Freie Universität Berlin since 2011 (Schäfer and Bildhauer, 2012).[1,2] I focus on the performance of the tool chain and its parallelization on high-performance clusters as well as the features of our web-based query interface. The architecture is capable of handling data sets where the size of the input is several TB and the size of the final corpus is up to (conservatively estimated) 100 gigatokens (GT). The software is freely available, and we are running a test instance of the query interface serving gigatoken web corpora in several European languages without charge.

First of all, I describe our software package that performs standard web corpus cleaning procedures in Section 2. Secondly, I briefly talk about our chains of wrapped annotation tools (available for Dutch, English, French, German, Spanish, Swedish) in Section 3. Finally, I introduce our web interface based on the IMS Open Corpus Workbench or OCWB (Evert and Hardie, 2011), which allows linguists to query very large corpora efficiently and conveniently, in Section 4.

2 Preprocessing

2.1 Implementation

The preprocessing package *texrex* performs HTML stripping, crawler and HTML meta data extraction, boilerplate detection, in-document paragraph deduplication, combined language

[1] http://hpsg.fu-berlin.de/cow
[2] http://corporafromtheweb.org

detection and text quality assessment (Schäfer et al., 2013), near-duplicate document detection, conversion to UTF-8, some UTF-8 normalizations, and geolocation lookup based on server IP addresses.[3] The non-trivial steps in this chain are boilerplate detection and document deduplication. Boilerplate detection is implemented as language-specific multilayer perceptrons (MLP) trained on human decisions. The boilerplate status is decided for blocks of text which simply correspond to the contents of certain HTML containers (primarily `<p>` and `<div>`). The system achieves very good accuracy (0.952 for German) to near-perfect accuracy (0.990 for French) in systematic evaluations (Schäfer, 2015, in prep.), which is a significant improvement over the previous version (Schäfer and Bildhauer, 2012), cf. Table 1.

lang.	prec.	rec.	F_1	corr.	base.	err. red.
English	0.983	0.990	0.990	0.976	0.910	0.066
French	0.995	0.994	0.994	0.990	0.897	0.093
German	0.963	0.977	0.977	0.952	0.798	0.154
Swedish	0.977	0.983	0.983	0.983	0.866	0.117

Table 1: Evaluation (means over 10 folds in a cross validation) of the *texrex* boilerplate detector; including the baseline (correct decisions achieved by classifying everything as boilerplate) and the raw reduction of error achieved by the MLP compared to the baseline; from Schäfer (2015, in prep.)

Removal of near-duplicate documents uses a conservative (unmodified) *w*-shingling approach (Broder, 2000). While *w*-shingles are generated by the main *texrex* tool, a separate tool (*tender*) calculates the estimated document similarity based on the *w*-shingles, and a third tool (*tecl*) creates the final corpus without duplicates. The *tender* tool has a high memory footprint because sorting the shingle databases is done in memory. Therefore, it allows for a divide–sort–merge approach with multiple runs of the software in order to make it usable under low-memory conditions.

2.2 Performance

In this section, I assess the performance of the preprocessing tools on three different types of systems, including estimates of the performance on big data sets. First, I performed a detailed per-algorithm benchmark on a quadcore Intel Core i5 at 2.38 GHz. I measured the performance of each algorithm on 11,781 German HTML documents read from a single input file using four threads for processing. Table 2 summarizes the results, showing that most algorithms run very fast, and that it takes 39 ms to process a single document on average. Even on a low-end machine, this means that over 5,000 documents per CPU core and second are processed.

Shingling is costly because it involves word tokenization of the document, n-gram creation, followed by the computation of m different hashes of each n-gram (in our case, $m = 100$, $n = 5$), cf. Broder (2000) or Schäfer and Bildhauer (2013, 61–63) for details of the procedure. That said, 14.25 CPU milliseconds per document on a low-end machine is highly acceptable. The *4-thread efficiency* (CPU time ÷ wall clock time) measures whether a potential parallelization overhead (with four processing threads on four physical cores) eats into the increase in efficiency achieved by using multiple threads. The factor is roughly 4 for almost all algorithms, which means that the wall clock time is actually a fourth of the CPU time when four threads are used. Using more threads seems to linearly increase the efficiency of the system, at least when there are not more threads than physical cores.

Then, in a first production run, I processed 189,143,035 documents from two crawls performed in 2011 and 2014 in the top-level domains *at*, *ch*, and *de*. The DECOW14A corpus of 20 GT was created from this (and other) input.[4] To saturate the available physical cores, the software was configured to use 14 worker threads on a single 12-core Xeon X5650 at 2.67 GHz with 128 GB RAM. Processing the whole corpus took a total of 336,474 seconds or 3.89 days, which is quite long considering that this does not even include the document similarity calculations by *tender*.[5] Therefore, I switched to the high performance cluster (HPC) of our university.[6] It currently offers 112 nodes with 2 hexacore Xeon X5650 each and between 24 and 96 GB RAM per node.[7] The

[3] `http://texrex.sourceforge.net`

[4] `http://corporafromtheweb.org/decow14`

[5] Notice that this means that 562.13 documents per second were processed, i.e., 40.152 documents per and thread and second. This is consistent with the 25.64 documents per CPU and second on the low-end system, cf. Table 2.

[6] `https://www.zedat.fu-berlin.de/HPC/Home`

[7] A reviewer mentioned replicability and applicability issues of results obtained on HPC systems which not everybody has access to. I agree, but would like to point out that creating very large corpora will always take either a very long time (up

Algorithm	ms/doc	docs/CPU/s	docs/CPU/day	4-thread efficiency
perfect duplicate detector	0.2527	3957.61	341,937,504	3.81
basic processing	22.9938	43.49	3,757,536	3.94
UTF-8 validator	0.1874	5337.53	461,162,592	4.23
deboilerplater	3.1497	317.49	27,431,136	4.02
w-shingle creator	14.2489	70.18	6,063,552	3.98
text quality assesment	3.2807	304.81	26,335,584	3.90
normalizer	2.3648	422.87	36,535,968	4.00
paragraph deduplicator	0.1891	5287.70	456,857,280	2.20
full configuration	**39.0081**	**25.64**	**2,215,296**	**3.96**

Table 2: Benchmark breakdown by algorithm. All values are arithmetic means over CPU times measured over 5 runs with 2 minute cooling off between runs.

input data was split into 100 parts, and 100 separate jobs using 6 threads each were queued. Since the HPC uses the SLURM (fair share) scheduling system, run times vary depending on the current cluster load.[8] In three consecutive runs, however, processing the whole corpus was done in under 5 hours.

Since the *tender* document similarity calculation tool allows for a divide–sort–merge approach, this step was also split up (this time into 10 jobs), and it took roughly six hours.[9] Since SLURM allows users to queue jobs depending on other jobs to finish first, I finally configured the system to automatically run a sequence of *texrex* and *tender* jobs for the whole corpus without manual intervention in roughly 8 hours. Clearly, the creation of corpora up to 100 GT is feasible on such a system with our software in no more than 2 days. It should be noticed that compared to systems using MapReduce (such as Hadoop), operating a SLURM cluster is arguably much simpler.[10]

3 Linguistic annotation

For space reasons, I focus on the linguistic annotation of our current corpora of English (16.8 GT) and German (20 GT). The main criteria for choosing a tool as part of the COW14 tool chain were its efficiency and the availability of pre-trained models based on annotation schemes which are well known within the linguistic community. For sentence and word tokenization, I used Ucto, because it allowed me to implement language-specific improvements for the tokenization of text from forums, social media, etc. (e. g., emoticons, creative use of punctuation) in a very straightforward way.[11] For part-of-speech (POS) tagging and lemmatization I therefore used TreeTagger (Schmid, 1995) with the standard models (Penn Treebank and STTS tag sets). The German TreeTagger model was complemented with 3,866 lexicon additions in order to remedy the problem that the publicly available models (trained on newspaper texts) do not contain entries for more recent lexical items or those found in non-standard language (e. g., *Anime*, *bloggen*, *Email*) or names which are more frequent now than in the 1990s (such as *Obama* or *Özil*). German was additionally annotated for named entities using the Stanford NER tool (Finkel et al., 2005) and the available German models (Faruqui and Padó, 2010).[12] It was morphologically analyzed using the (quite slow) morphological analyzer from mate-tools (Björkelund et al., 2010).[13] English was parsed with MaltParser (Nivre et al., 2007), and we are working on German models for MaltParser, too.[14]

The tool chain simply consists of a series of Bash and Perl scripts for pre- and post-processing the data for each of the annotation tools and piping

to virtual infeasibility) or require very powerful machines. In the first production run, it was at least proven that gigatoken corpora can be created on more common machines with a few days of patience.

[8] `https://computing.llnl.gov/linux/slurm`

[9] The high memory demands of the tool incur a high penalty in the queuing system, hence most of these six hours was wasted waiting for high-memory nodes. More tests with smaller portions of data and consequently more modest memory needs are necessary to optimize the run time.

[10] `https://hadoop.apache.org`

[11] `http://ilk.uvt.nl/ucto`

[12] `http://nlp.stanford.edu/software`

[13] `https://code.google.com/p/mate-tools`

[14] `http://www.maltparser.org`

the data to the tools. SLURM is ideally controlled via Bash scripts, so this was the method of choice. The major problem was the fact that most annotation tools cannot deal with (or at least just skip) XML, and the *texrex* tool described in Section 2 creates XML output. Most of the extra pre- and post-processing was therefore related to working around this. The target format of our corpora produced by the annotation tool chain is XML with in-line linguistic annotations in VRT format, as accepted by the IMS OCWB.

Due to the influence of the SLURM queuing system on performance, it is difficult to give exact performance figures. What is more, the tool chain is not fully automated yet, such that time was lost due to periodic manual intervention. In practice, processing the whole German corpus (including the costly steps of named entity recognition and morphological analysis) of 20 GT took under six days with most time spent on named entity recognition and morphological analysis.

4 Access to the corpora

4.1 Distribution

We redistribute our corpora (download and query interface) as shuffle corpora (i. e., bags of sentences). Similarly, the Leipzig Corpora Collection (LCC) has for a long time been redistributing web corpora in shuffled form.[15] While the LCC offers downloads to everyone, we additionally require that users be registered. Only users who work in the academia and provide a short abstract of their research plan are granted access to COW. The percentage of registration attempts denied by us was 34.3% as of June 10, 2015, which illustrates that we strictly enforce the criteria set by our terms of use. The fact that the German Research Council (Deutsche Forschungsgemeinschaft, DFG) are currently funding work on COW based on a proposal which specifically mentions the redistribution of shuffle corpora is an encouraging backup for our strategy.

4.2 Target audience and interface

The intended users of the COW corpora and the Colibri[2] interface, to which I turn now, are linguists working on lexicography, morphology, syntax, and graphemics. Very often, these researchers need to have concordances locally available for further manual annotation. Hence, the typical corpus query workflow (assuming a web interface) is: (i) preview a query, and (ii) download concordance if results look good, or modify the query and go back to (i). The Colibri[2] interface implements exactly this workflow.[16] Users make queries, either in a simple syntax (cf. Section 4.3) or in native CQP syntax. Queries in simple syntax are transparently translated into CQP syntax, and manually entered CQP syntax is checked for well-formedness.

A preview of maximally 100 hits is then returned and displayed in a KWIC view, cf. Figure 1. Users can then decide whether they want to download a larger concordance for that query containing maximally 10,000 hits in tab-separated format, and including (if desired) any of the annotations contained in the corpus (Figure 2).[17] Filters on structural attributes can be defined semi-graphically (cf. Figure 3) in order to restrict queries to strata of the corpus for which some meta data annotation matches or does not match a regular expression.

4.3 Simplified query language

Users who do not want to enter CQP syntax themselves can use Colibri[2]'s simplified query language, which offers only a few basic operators for corpus searches. To keep it simple, the language will not be extended or modified. Translation to native CQP syntax is done exclusively and transparently in the interface.

First of all, case-sensitivity cannot be specified as part of a query but is rather switched on and off globally using a button. A query consists of a sequence of literal tokens and lemmas, wherein lemmas have to be prefixed with `^`. Within tokens and lemmas, `*` can be used as the wildcard for zero or more arbitrary characters. Token distances (other than the default of 0) can be specified as `\n` (fixed distance of n tokens) or `\n-m` (distance of n to m tokens). See Figure 1 for an example.

4.4 Context reconstruction

Because single sentences without a larger context are useless for some types of linguistic research, we have created a tool that reconstructs contexts

[15] `http://corpora.uni-leipzig.de`

[16] `https://webcorpora.org`

[17] The limitation to 10,000 is implemented in the interface and can be circumvented in API mode using HTTP GET requests.

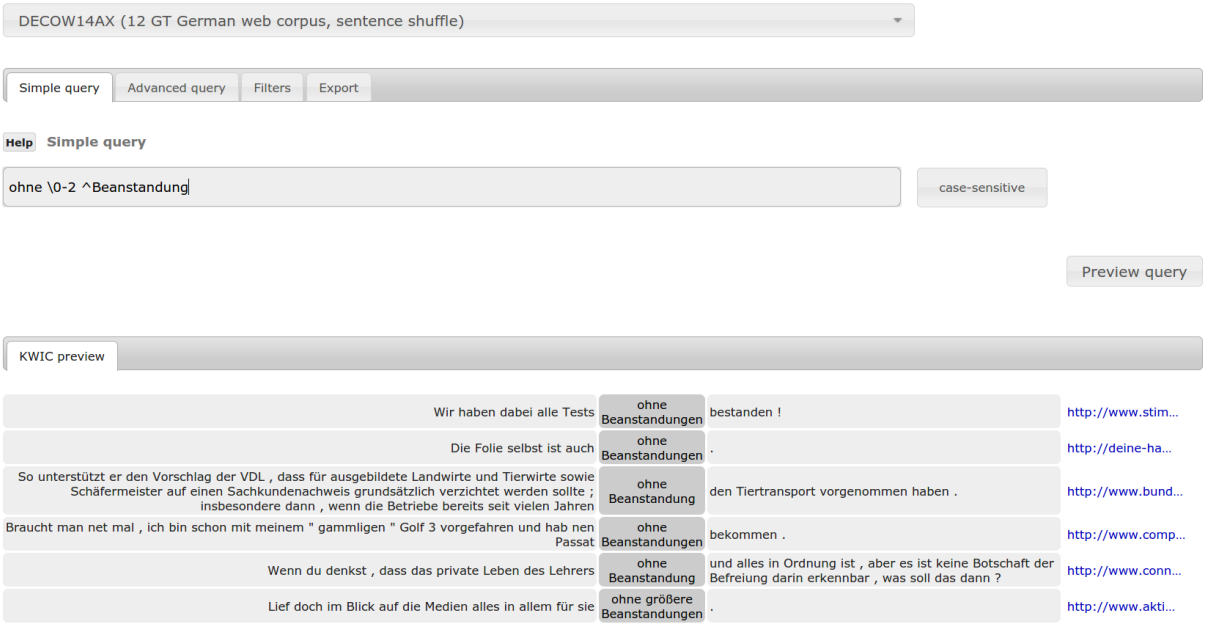


Figure 1: Colibri[2] simple search view and part of a KWIC preview; the simple query is translated to `[word="ohne"%c][]{0,2}[lemma="Beanstandung"%c]`

for at least some sentences in any concordance exported from Colibri[2]. The tool is called *Calf*, it is written in Python and available on all common platforms.[18] Using *Calf*, researchers can download the contexts of sentences in Colibri[2] concordances from the original resources available on the web.

Calf reads in concordances exported from Colibri[2] which include the URLs of the original web pages. If the web page is still available, it is downloaded, tokenized, and the sentence from the concordance is searched using a fuzzy matching strategy. In case this fails (i. e., the page is no longer available or its contents have changed), the sentence is queried using Google's search engine. *Calf* then tries to locate the sentence on the pages returned by Google. If the sentence was found either under the original URL or using Google, a context of a configurable number of characters is extracted and added to the concordance.

Detailed evaluations of the method will be published elsewhere, but as an example, I have exported a concordance returned by Colibri[2] for the word *Chuzpe* in DECOW14AX. It contained 201 sentences which *Calf* processed in 12 minutes and 54 seconds using an ordinary DSL line. Of the 201 sentences, 97 were found using the original URL, and an additional 36 sentences were found using Google, resulting in 133 (66%) successfully reconstructed contexts.

4.5 Architecture

The Colibri[2] system can deal with corpora of virtually arbitrary size, even though the underlying IMS OCWB has a hard limit of roughly 2 GT per corpus. To achieve this, the system accesses large corpora partitioned into several sub-corpora. Our German corpus, for example, comes in 21 partitions of roughly 1 GT each. These partitions can be installed on arbitrarily many back-end servers, where PHP code talks to the CQP executable, cf. Figure 4. The interface, implemented in the user's browser in JavaScript using jQuery and jQuery UI, sends queries to the front-end server. Query checking and management of user credentials are implemented exclusively in the front end server. If the user has the appropriate rights and the query passes all sanity checks, the front end server sends queries to the back end servers and aggregates the results, before serving the data to the user interface. The front end server talks to the back end servers either in serial or parallel mode, where in the parallel mode a configurable number of back end servers is called simultaneously. Especially the parallel mode allows the capacity of the system (in terms of numbers of users and corpus sizes) to grow, with the network traffic between front end server and back end servers being the main limit-

[18] `http://corporafromtheweb.org/calf`

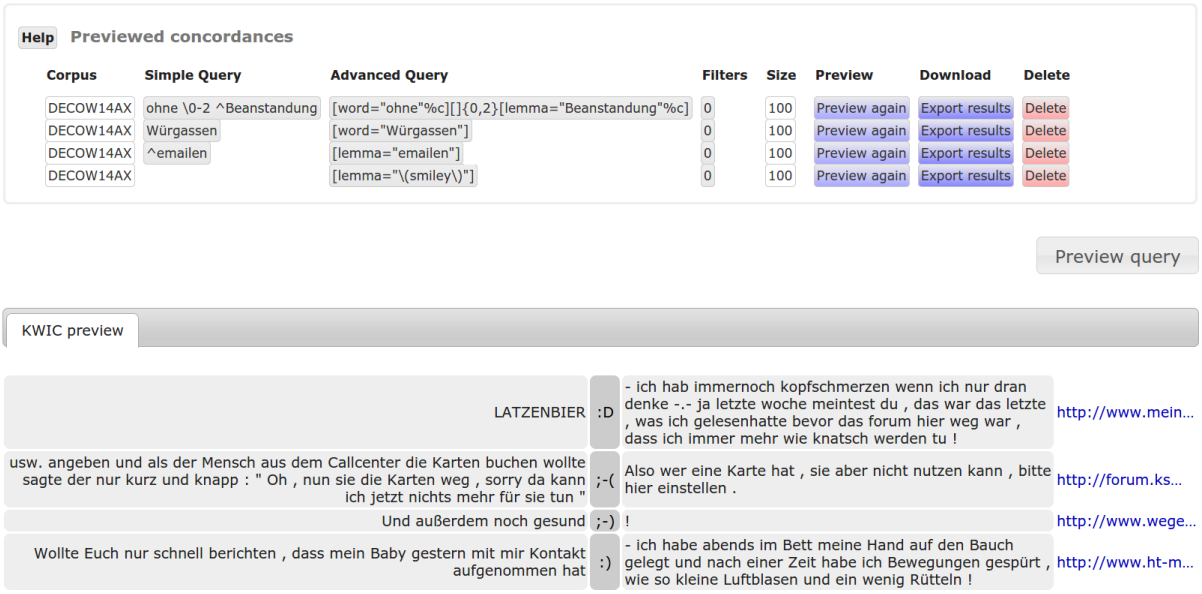


Figure 2: Colibri[2] results view and part of a KWIC preview

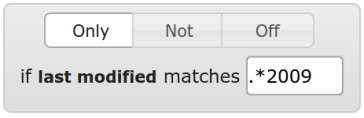


Figure 3: Sample filter on structural attributes; only sentences from web pages with *last-modified* header from 2009 will be returned

ing factor.

On our reference system, all communications are secured by SSL. The granularity of access rights is currently restricted to (i) public corpora and (ii) corpora requiring login. More fine-grained access rights management is planned. As of June 10, 2015, we serve 190 users on a single low-end virtual server with 14 virtual cores, 14 GB RAM, 400 GB SSD storage, and a 100 Mbit/s connection.[19] The server simultaneously acts as the front end server and the only back end server, so we do not even take advantage of the advanced load distribution features of the system. Nevertheless, there have so far been no performance issues.

5 Summary and outlook

The set of tools developed for COW14 as described in this paper allows us to efficiently build very large web corpora (conservatively estimated up to 100 GT). The use of a simple SLURM-based HPC approach to parallelization allows us to use any tool which we want for linguistic annotation by wrapping it in a Bash script, and we are therefore experimenting with more and advanced annotation tools for dependency parsing, text classification (register, genre, etc.), etc. Finally, we do not only create the corpora, but we also bring them to the working linguist free of charge. Based on user feedback, we have many plans for the interface. Above all, we are going to implement static links to absolute corpus positions, such that requests following the scheme `webcorpora.org/ref/<corpus>/<position>` will allow users to quote corpus examples with a unique identifier and also exchange such links.

[19]The SSD storage, although still highly expensive in servers, appears to be crucial for good performance.

Acknowledgments

I would like to thank Felix Bildhauer for ongoing joint work on the COW corpora since 2011. I would also like to thank the HPC service offered by the Zedat data center of Freie Universität Berlin for computing resources. Also, Stefan Müller of Freie Universität Berlin has provided an enormous amount of computing and storage resources for COW, for which I thank him. The work presented here was partially funded by the German Research Council (DFG) through grant SCHA1916/1-1.

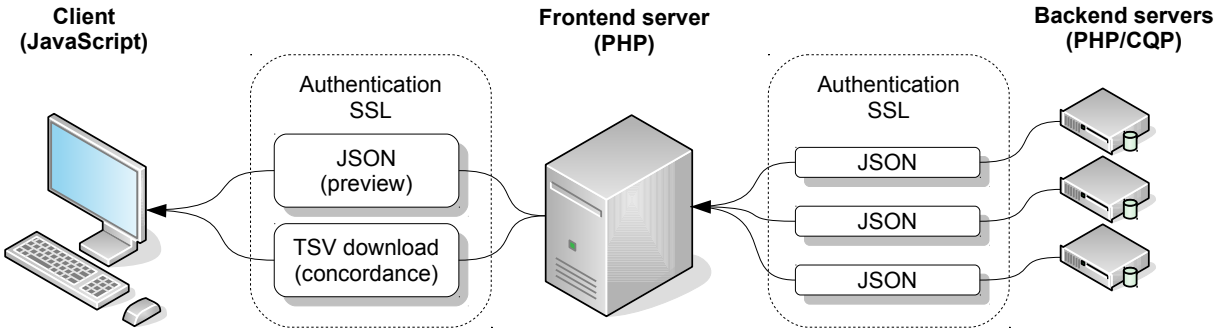


Figure 4: Colibri[2] architechture

References

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros ZanchettaBaroni. 2009. The WaCky Wide Web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Chris Biemann, Gerhard Heyer, Uwe Quasthoff, and Matthias Richter. 2007. The Leipzig Corpora Collection - Monolingual corpora of standard size. In *Proceedings of Corpus Linguistics 2007*, Birmingham, UK.

Chris Biemann, Felix Bildhauer, Stefan Evert, Dirk Goldhahn, Uwe Quasthoff, Roland Schäfer, Johannes Simon, Leonard Swiezinski, and Torsten Zesch. 2013. Scalable construction of high-quality web corpora. *Journal for Language Technology and Computational Linguistics*, 28(2):23–60.

Anders Björkelund, Bernd Bohnet, Love Hafdell, and Pierre Nugues. 2010. A high-performance syntactic and semantic dependency parser. In *Coling 2010: Demonstration Volume*, pages 33–36, Beijing.

Andrei Z. Broder. 2000. Identifying and filtering near-duplicate documents. In R. Giancarlo and D. Sanko, editors, *Proceedings of Combinatorial Pattern Matching*, pages 1–10, Berlin.

Stefan Evert and Andrew Hardie. 2011. Twenty-first century corpus workbench: Updating a query architecture for the new millennium. In *Proceedings of the Corpus Linguistics 2011 conference*, Birmingham. University of Birmingham.

Manaal Faruqui and Sebastian Padó. 2010. Training and evaluating a german named entity recognizer with semantic generalization. In *Proceedings of KONVENS 2010*, Saarbrücken, Germany.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 363–370.

Eugenie Giesbrecht and Stefan Evert. 2009. Part-of-speech (POS) tagging – a solved task? an evaluation of POS taggers for the German Web as Corpus. In Iñaki Alegria, Igor Leturia, and Serge Sharoff, editors, *Proceedings of the Fifth Web as Corpus Workshop (WAC5)*, pages 27–35, San Sebastián. Elhuyar Fundazioa.

Adam Kilgarriff and Gregory Grefenstette. 2003. Introduction to the special issue on the Web as corpus. *Computational Linguistics*, 29:333–347.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Helmut Schmid. 1995. Improvements in part-of-speech tagging with an application to German. In *Proceedings of the EACL SIGDAT-Workshop*, Dublin, Ireland.

Roland Schäfer and Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 486–493, Istanbul. ELRA.

Roland Schäfer and Felix Bildhauer. 2013. *Web Corpus Construction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool, San Francisco.

Roland Schäfer, Adrien Barbaresi, and Felix Bildhauer. 2013. The good, the bad, and the hazy: Design decisions in web corpus construction. In Stefan Evert, Egon Stemle, and Paul Rayson, editors, *Proceedings of the 8th Web as Corpus Workshop (WAC-8)*, pages 7–15, Lancaster. SIGWAC.

Roland Schäfer. 2015. Accurate and efficient general-purpose boilerplate detection. in prep.

Release of the MySQL based implementation of the CTS protocol

Jochen Tiepmar
ScaDS
Leipzig University
Ritterstrasse 9-13, 2.OG
04109 Leipzig
jtiepmar@informatik.uni-leipzig.de

Abstract

In a project called "A Library of a Billion Words" we needed an implementation of the CTS protocol that is capable of handling a text collection containing at least 1 billion words. Because the existing solutions did not work for this scale or were still in development I started an implementation of the CTS protocol using methods that MySQL provides. Last year we published a paper that introduced a prototype with the core functionalities without being compliant with the specifications of CTS (Tiepmar et al., 2013). The purpose of this paper is to describe and evaluate the MySQL based implementation now that it is fulfilling the specifications version 5.0 rc.1 and mark it as finished and ready to use. Further information, online instances of CTS for all described datasets and binaries can be accessed via the projects website[1].

1 Introduction

CTS is a protocol developed in the Homer Multitext Project[2] and, according to (Blackwell and Smith, 2014), "defines interaction between a client and server providing identification of texts and retrieval of canonically cited passages of texts" by using CTS URNs, that "are intended to serve as persistent, location-independent, resource identifiers".

These URNs are built in a way that resembles the hierarchy in- and outside the document. The URN *urn:cts:demo:goehte.faust.de:1.2-1.4* refers to the text passage spanning from act 1 scene 2 to act 1 scene 4 of the document Goethe's Faust. The first part *urn:cts:* marks it as an URN of the CTS protocol. The second part *demo:* refers to the namespace that the text belongs to. *goehte.faust.de:* refers to the edition (document) and *1.2-1.4* specifies the text passage inside the document. With the addition of the @-notation for subpassages, like in *1.2@hu-1.4@d*, you can specify any text passage in any translation or edition.

The citation depth and structure can differ between documents - while one document can be structured on 4 levels, like book, chapter, section and sentence, it is also valid to structure another document (or even another edition of the same document) in a different way. This means that – for example – while the passage 2.1 in a bible can refer to part 1 of book 2, in Shakespeare's Sonnets, 2.1 refers to verse 1 of sonnet 2. By reducing the type of each text unit to a label, the protocol makes it possible to use any possible text. The worst case scenario would be that no information about the structure of a document is available, in which case it is still possible to use lines as text units.

Even if it might not be intended to be used as such by the authors of the specifications, CTS can serve as a way to standardize texts and therefore work as a text catalogue or -repository. Furthermore, any tool that uses the methods that CTS provides, can work with any data that is or will be added, basically making CTS a framework and standard for public access to text.

Smith (2007) points out another advantage of the usage of CTS: "These Canonical Text Services URNs make it possible to reduce the complexity of a reference like "First occurrence of the string 'cano' in line 1 of book 1 of Vergil's ~~Aeneid~~" to a flat string that can then be used by any application that understands CTS URNs". This also means that you can reduce long texts to URNs and then request them as they are needed and this way reduce the memory needed for software that handles texts or text parts.

Using it as a text repository requires a very fast and efficient implementation of the protocol. The

[1] www.urncts.de
[2] http://www.homermultitext.org/

prototype already showed potential for this goal by building maximal passages with response times averaging at 78 MS with a text collection that contains 100‘000 documents with 1‘281‘272‘600 tokens (Tiepmar et al., 2013). As I will show in chapter 7, the implementation still performs fast as it is finished.

While working on this project, 3 major text collections were published as instances of CTS. They are described in chapter 6.

2 Using Canonical Text Services

This chapter is intended to give a rough overview about the specifications defined in (Blackwell and Smith, 2014) and explain the workflow with CTS. Data from CTS is collected via HTTP requests. Each request has to include a GET parameter *request* which specifies, what function of CTS is requested. Attributes are added as GET parameters to the HTTP request. The following functions are available in CTS 5.0 rc.1.

2.1 GetCapabilities

GetCapabilities returns the text inventory of the CTS with all the URNs of works or editions as well as meta information for each entry. The extend or content of the meta information is not specified in CTS.

2.2 GetValidReff(urn,level)

GetValidReff returns all the URNs that belong to the given *urn*. *level* is a required parameter specifying the depth of the citation hierarchy.

2.3 GetLabel(urn)

The request *GetLabel* returns an informal description of the *urn*.

2.4 GetFirstUrn(urn)

GetFirstUrn returns the first URN in document order belonging to the given *urn*.

2.5 GetPrevNextUrn(urn)

GetPrevNextUrn returns the previous and next URN in document order from the given *urn*.

2.6 GetPassage(urn,[context])

GetPassage returns the text passage that belongs to this *urn*. *context* is an optional parameter specifying, how many text units should be added to the passage as contextual information.

2.7 GetPassagePlus(urn,[context])

GetPassagePlus returns the combined information from 2.2 to 2.6

2.8 The Response

The response for each request is a XML-document describing the request and the response from the CTS. For example the response for a GetPassage request is structured according to the following XML-document:

```
<GetPassage>
<request>
<requestName>
GetPassage
</requestName>
<requestUrn>
urn:cts:latinLit:phi1014.phi001.lat1:1
</requestUrn>
</request>
<reply>
<urn>
urn:cts:latinLit:phi1014.phi001.lat1:1
 </urn>
<passage>
(...)
</passage>
</reply>
</GetPassage>
```

It may seem odd that the URN is listed two times. If you do not specify the exact edition it can happen that both URNs differ. Requesting the text passage with urn:cts:latinLit:phi1014.phi001:1 may result in the text passage for urn:cts:latinLit:phi1014.phi001.*lat1*:1[3].

There are contradictory information about whether or not the XML elements must reference CTS as a namespace, like <cts:urn> instead of <urn>[4]. All XML elements in the replies of this implementation are unique and there is no need to differentiate them with namespaces. That's why I chose to not include them. This can be changed as soon as the specifications make it clear, which format should be used.

[3] According to the specifications, an implementation of CTS is free to choose any suitable edition if the edition is not fully specified in the URN.

[4] Compare for example https://github.com/cite-architecture/ctsvalidator/blob/master/src/main/webapp/testsuites/4-09.xml and https://github.com/cite-architecture/cts_spec/blob/master/reply_schemas/prevnext.rng

3 Validation

The specifications refer to a validator that checks whether or not an instance of CTS is compliant with the specifications. Unfortunately, some of the results that the validator expects contradict the specifications making it impossible to validate this implementation[5].

4 Data Structure

This chapter will give an abstract overview about the data structure used in this implementation. A more technical description can be found in (Tiepmar et al., 2013).

To implement an efficient CTS it was crucial that the underlying data structure is as efficient as possible. The best case would be a data structure that resembles the hierarchical structure that is encoded in CTS URNs and this way minimizes the overhead that is needed to describe the structural information. By storing this information in a tree you get a structure that can be modelled similar to the tree in Figure 1.

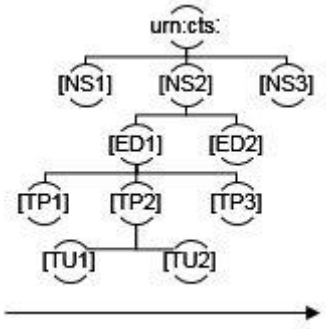


Figure 1, Visualization of the tree-like data structure
NS=Namespace (e.g. greekLit)
ED=Edition (e.g. Goethe's Faust)
TP=Text part (e.g. Chapter)
TU=Text unit (e.g. Sentence)

[TUx] contains the text content for each text unit. The nodes on [TU] level must be ordered as they appear in the document. This is done by using an incremental id indicated by the arrow.

To make sure that you cannot concatenate multiple editions, the CTS will always at least traverse down to edition level and return the first node on that level. Once the node for an URN is found, any related information can be returned. Parent child nodes can be calculated by deleting parts of the URN. The passage can be constructed by concatenating the text units that belong to the node. The child nodes resemble the URNs that belong to the given URN and the first and last child node correspond to the first and last child URN.

When searching for the URN urn:cts:[NS2]:[ED1]:[TP2] the implementation traverses through the tree to the node [TP2]. By this point it knows that this is a valid URN and can return any information associated with this node. If no suitable node is found, then the CTS knows that the URN is not valid. There may be a node [TP2] belonging to [ED2], but as soon as the CTS passed [ED1] this node is no longer in the potential result set.

Treelike data structures provide the benefit of logarithmic search times and (if implemented correctly) prefix- and suffix optimisation, which is beneficial for CTS because the URNs contain a lot of redundant prefixes.

MySQL uses B-Trees for string indices and therefore I considered it a perfect fit for CTS URNs. Another – maybe less technical and more intuitive – way of visualizing it, is that this implementation is using techniques that are generally used for automated completion of strings to build the hierarchy of CTS URNs.

5 Unique Features

There are four unique features to discuss: the possibility to post process the passage, the configuration parameter, the generated text inventory and possibility of multiple import methods. The following chapters will explain these features in detail, give examples of use cases and explain how they fit into the specifications.

5.1 Passage Post Processing

According to (Blackwell and Smith, 2014), the passage "may (…) be further structured or formatted in whatever manner was selected by the editor of the particular edition or translation". This means, that CTS does not restrict the content of the passage in any way as long as "The CTS implementation (…ensures…) that including the contents of the requested in the cts:passage element results in well-formed XML" (Blackwell and Smith, 2014)[6]. As long as it does not break the structure of the reply, the passage may be plain text or – for example – text that either contains XML tags as text or text with XML tags as meta information describing a part of the text.

The following examples help to illustrate the difference.

[5] See issue 26, 27, 28, 29 at https://github.com/cite-architecture/ctsvalidator

[6] The cts:passage element is the XML element in the CTS reply that contains the text passage specified the the URN

a) *The tag <speaker> refers to a speaker and must be closed by </speaker>*
b) *<speaker>Hamlet </speaker>To be, or not to be(...)*

While a) should clearly be seen as plain text describing the tag <speaker>, it is reasonable for an editor to prefer the structured output in example b).

Changing a) to

A) *The tag <speaker> refers to a speaker.*

it becomes obvious that this probably breaks the structure of the CTS reply.

One solution here would be to make sure that every document only contains valid XML. This means that you would either restrict your text to valid XML or have to make sure that anything that would potentially break the XML structure, must be escaped. This results in a lot of work for the editors since they cannot simply escape the whole text but have to differentiate structural tags used by the CTS (like <chapter>) from meta tags that are part of the text (like <speaker>).

The solution that I propose is to make it possible to adapt the content of the passage by the CTS to the needs of the individual text collection or even to the needs of the individual viewer or editor. As long as the post processing method, that is used to modify the passage, is not changed, the CTS still guarantees a persistent citation. One URN will always result in the same text passage, but the data is presented differently. The CTS does not change the textual content, but its representation (or the view on the data) changes.

On the side of the server, this is nothing different than the possibility to serve the text in "whatever manner was selected by the editor" (Blackwell and Smith, 2014). In general, this is the same as creating annotated editions of one document, which is already a common method in today's Digital Humanities as – for example – described in (Almas, 2013). Doing this on CTS level is just automating the process.

On the opposite side, the client can benefit from this by having options. Imagine someone who wants to develop a universal reader for documents in EpiDoc format. It would be very useful to be able to connect to a CTS and have the possibility to request any text in this format without the need to rebuild all the documents and add additional EpiDoc editions. Another reader wants to look up some text but the edition is heavily annotated, making it hard to read. A view without all the XML tags would probably be something nice.

To enable the client to control the format of the passage, it is required to give the possibility to specify a configuration that should be used. This can be achieved with the configuration parameter that I will discuss in the next chapter.

5.2 Configuration Parameter

The configuration parameter was added to this implementation to give any client the possibility to adapt the output of the CTS in different ways. Its use is not described in the specifications but a side note makes it clear, that it does also not violate them. One valid example URL is http://myhost/mycts?configuration=default&request=GetCapabilities[7]. Because this url is valid, it is allowed to add additional parameters to the requests. Therefore it does not contradict the specifications to use it to give the client the ability to configure the CTS as long as the results are still valid against the specifications. In especially the CTS must still make sure, that the reply results in valid XML and all of the required information is included.

It is possible to combine multiple parameters by combining them with "_". For example, the configuration *?configuration=div=true_stats=true* combines the parameters *div* and *stats*.

The following parameters are currently supported. The default values for each parameter can be defined for every CTS instance. The configuration that the client provides will overwrite this default configuration.

Div / Epidoc

The parameters *div* and *epidoc* are useful if you want to see the structure of the text passage – for example to render it nicely. *div* uses a notation with numbered elements and includes the type of the text units as a @type value.

```
<passage>
<div1 n="5" type="book">
<div2 n="1" type="line">
(TEXT)
</div2>
</div1>
</passage>
```

epidoc uses EpiDoc notation, a variation of TEI/XML.

```
<passage>
<tei:TEI>
```

[7] http://folio.furman.edu/projects/citedocs/cts/#client-server-communication

<tei:text>
<tei:body>
<tei:div n="1" type="song">
<tei:div n="1" type="stanza">
<l n="1">(TEXT)</l>
<l n="2">(TEXT)</l>
</tei:div></tei:div>
</tei:body>
</tei:text>
</tei:TEI>
</passage>

epidoc is ignored if *div* is set to true.

Stats

stats does not yet serve a useful purpose but illustrates this implementations flexibility nicely by adding some simple statistics as @-values in the numbered divs. This setting is ignored if *div* is set to false.

<div3 n="1" type="line" letters="24" tokens="4" avg_tokensize="6">
(TEXT)
</div3>

Escapepassage

escapepassage specifies whether or not the XML content of the passage should be escaped. This is always true if URNs with subpassage notation are requested to ensure the validity of the reply.

Seperatecontext

If *seperatecontext* is set to true, then the context that is specified for GetPassage or GetPassagePlus is returned in separate XML elements with the name context_prev and context_next. Else the context is added to the passage and returned inside the passage element.

Formatxml

formatxml configures whether or not the reply should be formatted. Formatted XML is easier to read but if you want to process it automatically, formatting may not be needed and influence the performance of the CTS negatively without having any benefit.

Smallinventory

smallinventory reduces the text inventory to a list of <edition> elements with their URNs. I noticed, that dealing with lots of documents can result in large text inventories that are hard to parse if all the meta information is included. This meta information may be unnecessary if you only need a list of the documents URNs.

Maxlevelexception

If you set *maxlevelexception* to true and then specify a level for GetValidReff that is higher than the levels that the document 'has left', it will return CTS error 4. Else it will return the URNs up to that level. For example if your document has two levels: chapter and sentence, and you request GetValidReff with level=100, then the CTS will return error 4 if this is set to true. It will return all the URNs that belong to the given URN if this is set to false.

The validator requires the CTS to return error 4 if you request a level higher than the document provides[8]. However since there is no way of knowing, how a document is structured and GetValidReff is the function that gives you this information, this would force a user to try out levels until they receive an error, which gets more complicated considering that the document structure is not fixed for the complete document. While in a document book 1 may have 3 levels – chapter, passage, sentence – book 2 of the same document may be structured in 2 levels – stanza, line. This means that you can never know, if you can request another level until you received an error. You can add this information as meta information in GetCapabilities but it is not required by CTS to do so and this solution would still make it problematic to work with documents containing different citation levels.

In my opinion it is more reasonable to ignore this error and make it optional for validation purposes.

This also fits with the specifications noting that "The GetValidReff request identifies all valid values for one on-line version of a requested work, up to a specified level of the citation hierarchy"(Blackwell and Smith, 2014)[9].

5.3 Dynamically Generated Text Inventory

GetCapabilities returns a text inventory containing all URNs that belong to works or editions. This text inventory is manually edited and serves as an overview about what texts are part of the CTS and as a guide for the CTS to know which XML tags of a document are part of the citation.

[8] See https://github.com/cite-architecture/ctsvalidator/blob/master/src/main/webapp/testsuites/3-19.xml

[9] http://folio.furman.edu/projects/citedocs/cts/#cts-request-parameters

Working with a big number of documents, it might be problematic to require someone to read all the documents, create citation mappings, collect the meta information for each document and store it in the inventory file.

While you still have to configure the citation mapping in this implementation, you do not need to do this for every document (you still can if you want). It can be configured in one line for all documents while setting up the CTS. This means that the text inventory is not required to import data, reducing its purpose to the output of GetCapablities. According to (Blackwell and Smith, 2014), the response of GetCapabilities is "a reply that defines a corpus of texts known to the server and, for texts that are available online, identifies their citation schemes". This information can be gathered in an automated process once the data is made available to the CTS.

This way a basic default text inventory is generated which contains all the referenceable editions without the need for manual editing. At the moment of writing, the label and author of an edition and the information, whether or not the edition can be parsed as valid XML, is added as meta information. This result is generated with every new request.

The following example shows the content that is currently included in the text inventory.

```
<TextInventory>
<textgroup urn="urn:cts:greekLit:tlg0003">
<groupname>tlg0003</groupname>
<edition urn="urn:cts:greekLit:tlg0003.
tlg001.eng1:">
<title>
History of the Peloponnesian War
</title>
<author>Thucydides</author>
<contentType>xml</contentType>
</edition>
</textgroup>
</TextInventory>
```

The citation mapping – as it is used to specify, which XML elements are used for citation in the CTS implementation based on a XML database – is not part of the generated inventory because from my understanding it is only useful for the data import. My argument is that once you reference texts with URNs, the citation mapping has only descriptive use and it is better located in the specific text passage or in the reply of the CTS request GetLabel. If you refer to a passage with a URN like urn:cts:demo:a:1.2, it is not relevant, whether the passage – 1.2 – refers to a sentence or verse or line. Adding it to the text inventory can however increase the complexity of the XML document making it harder to process the file. Especially consider that – in theory – every text unit that is referenced by an URN can have its own citation mapping. Mapping one unit to a sentence does not mean that every text unit is a sentence. In the worst case scenario, if citation mappings are included, the text inventory would have to contain one entry for any URN on level of the text units in the complete text collection.

By adding a file named inventory.xml, administrators can instead use one that is manually edited. It is a very reasonable workflow to save the generated inventory as inventory.xml and edit it further to manually add information.

5.4 Multiple Import Methods

The implementation is divided into two parts: one part imports the data into the database and the other part reads the data from the database. This separation makes it possible to plug in new import scripts. At the moment of writing, there exist 3 supported ways to import data.

Local import is the default way that this system uses.

CTS cloning makes it possible to clone one CTS. Since it relies on the div-configuration, it is currently only compatible with this implementation. In theory, this feature allows community driven decentralized data backups.

The third method relies on a MyCore installation that was used in the project "A Library of a Billion Words" and therefore might require a specific setup. However, together with this setup and using the possibility of timestamp related queries in OAI PMH, we created a self-updating CTS with support for versioning and this way created a persistent CTS with editable content[10].

6 Available Texts

While the implementation was still in progress, it was possible to collect 3 major text collections. For evaluation purposes another corpus containing 100‘000 editions with 1‘281‘272‘600 tokens was generated from random sentences.

[10] A cronjob collects the files, that were changed since the last update via OAI-PMH and timestamps as part of the URNs guarantees persistency.

6.1 DTA (Deutsches Text Archiv)

DTA includes 5136 editions from the German Text Archive of the BBAW in Berlin. All documents are published in 3 editions – .norm, .translit, .transcript – marking different states of normalization. The documents are structured with one citation level (sentence) and include 334‘820‘482 tokens.

6.2 PBC (Parallel Bible Corpus)

PBC is based on the project Parallel Bible Corpus and contains 831 translations of the bible (including 5 different german translations) with 247‘292‘629 tokens. The documents are structured in 3 citation levels (book, chapter, sentence).

6.3 Perseus

Perseus is the dataset from the Perseus project updated in November 2014. This is a well known text collection, containing mainly greek and latin documents that are manually annotated. The documents are structured heterogeneously and the citation depth varies for each document. This corpus adds another 27‘670‘121 tokens and is especially relevant since it is closely related to CTS (see Crane et al., 2014).

7 Evaluation

To evaluate this implementation I used a virtual machine (VM) that was part of our universities network. To make sure that the traffic outside of the VM does not interfere with the results, all requests were sent via localhost. I measured the time it needs to send the request and to get and read the response. Requesting the data from outside the VM would have been a more realistic scenario but would also have included the noise from the network. Since CTS cannot influence the latency of the network in any way, this would also not have been very constructive. Aside from whatever caching strategies are used by Apache Tomcat or MySQL, no caching is used by this implementation. Each response is generated as it is requested.

The test system has a Common KVM processor with one 2,4 GHz core and 1 GB memory. Only one dataset is loaded at any time during the tests and before any test is started, I rebooted the system.

All the URNs of editions were collected and for each one the passage spanning the 2 first URNs on citation level 1 was requested. If there was no second URN on level 1, then level 2 was used. If this was not possible, this edition is ignored.

Depending on the structure of the document, the passages can differ in text length. Passage 1-2 of Luther's “Die Bibel in Deutsch“ spans the books 1 to 2 while the same passage in Schillers “Kabale und Liebe“ as it is structured in this case includes the sentences 1 to 2. This means that the results are not comparable between the datasets. The average number of characters in the generated text passage is given for each diagram.

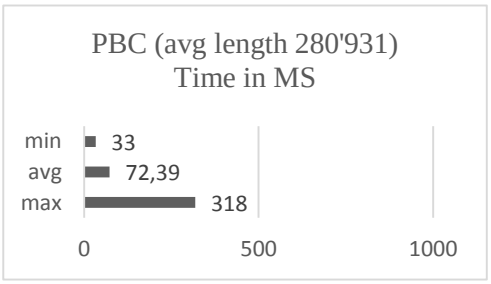


Figure 2, Minimum, average and maximum response times for the PBC dataset

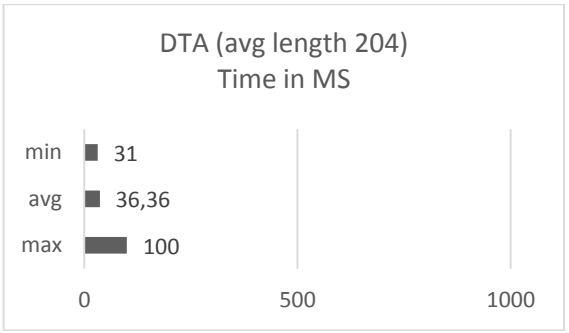


Figure 3, Minimum, average and maximum response times for the DTA dataset

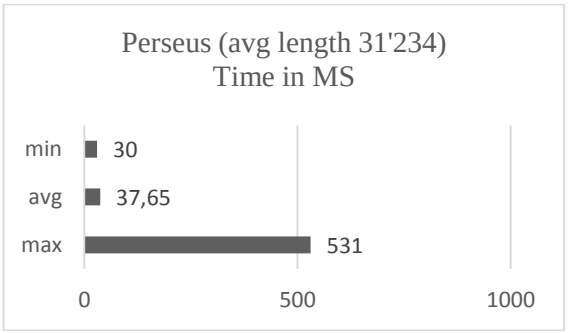


Figure 4, Minimum, average and maximum response times for the Perseus dataset

70/1176 editions of Perseus did not contain any text and were ignored.

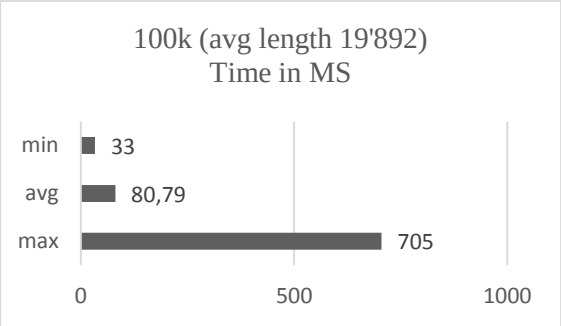


Figure 5, Minimum, average and maximum response times for the 100k dataset

4'800/100'000 documents consist of only 1 sentence and could therefore not deliver a passage 1-2.

In general the results show that the MySQL based implementation performs very well and stays under 1 second in any case. It seems like the response time depends more on the size of the passage that is requested than on the size of the text collection. If the passages length influences the response time, the average response time should reflect this if you limit the result set to 1/4 or 1/10 of the longest or shortest passages in one test run.

	Shortest 1/4 (MS)	Longest 1/4 (MS)
DTA	36,00	37,29
PBC	60,70	91,50
Perseus	33,76	47,86
100K	78,64	83,08

Table 1, Response times for 1/4 of the longest and shortest text passages

	Shortest 1/10 (MS)	Longest 1/10 (MS)
DTA	35.90	37,93
PBC	56,62	98,05
Perseus	33,59	60,24
100K	75,31	81,51

Table 2, Response times for 1/10 of the longest and shortest text passages

Unsurprisingly the length of the requested passage influences the response time (a little bit). However, the differences are small and background noise of the operating system might also have had an impact. It is hard to argue, that such small differences in milliseconds mean anything.

Comparing the results from DTA and PBC, it seems like other factors are also influencing the response time. The 3 longest passages in DTA are 1‘915, 1‘944 and 1‘974 characters long while the 3 shortest passages in PBC are 9‘099, 9‘718 and 9‘793 characters long. Any passage from PBC is longer and also deeper structured than any passage from DTA. Still the PBC CTS could often respond faster than the DTA CTS. This could indicate an influence of the documents structure.

Another interesting value is the response time needed to collect passages spanning complete documents. The following table shows the minimum, average and maximum values for a documents complete passage length and the response times for the corresponding GetPassage request.

	Passage length (in 1000 MS) min | avg | max	Response time (MS) min | avg | max
DTA	0.5 | 444 | 7‘406	32 | 182 | 3‘444
PBC	80 | 163 | 6‘655	57 | 548 | 4‘859
Perseus	35 | 170 | 8‘457	32 | 70 | 3‘088
100k	0.016 | 82 | 438	31 | 86 | 922

Table 3, Minimum, average and maximum response times compared to the minimum, average and maximum passage lengths

Perseus includes the longest document with 8‘457'677 characters and 1‘350'876 tokens. This request also took the maximum time in the dataset with 3‘088 MS. The longest document – and again the document with the highest value for the response time – in DTA is Abelinus Theatrum in its translit edition[11] containing 1’082’893 tokens or 7’406’366 characters.

Considering the hardware limitations and the very good and relatively stable response times, it seems reasonable to include a lot more data into future tests and especially test, at which point this implementation starts to struggle.

Factors that can also be investigated in future evaluations are the influence of the structure of the document and the length of individual text units.

8 Conclusion

This paper marks the release of the MySQL based implementation of the CTS protocol. It introduces features that are exclusive to this software and argues why they are useful additions to the protocol while not contradicting the specifications. Evaluation shows that the performance is very good and sets a baseline for future implementations. It has also shown that this implementation is easily capable of handling a text collection containing one billion words and can be used as a text repository.

[11] urn:cts:dta:abelinus.theatrum1635.de.translit:

Acknowledgements

Parts of the work presented in this paper is the result of the project “Die Bibliothek der Milliarden Wörter”. This project was funded by the European Social Fund. “Die Bibliothek der Milliarden Wörter” was a cooperation project between the Leipzig University Library, the Natural Language Processing Group at the Institute of Computer Science at Leipzig University, and the Image and Signal Processing Group at the Institute of Computer Science at Leipzig University. This project is part of the project Scalable Data Solutions (ScaDS) funded by BMBF. ScaDS is a cooperation project between the Leipzig University and TU Dresden. This projects number is 01/5140148.

Reference

Almas B, Beaulieu M. 2013. "Developing a New Integrated Editing Platform for Source Documents" in Classics in Oxford Journals Literary and Linguistic Computing, Volume 28, Issue 4.

Blackwell C, Smith N. 2014. Canonical Text Services protocol specification. Retrieved from http://folio.furman.edu/projects/citedocs/ctsurn/ and http://folio.furman.edu/projects/citedocs/cts/ 2015, February 19.

Crane G, Almas B, Babeu A, Cerrato L, Krohn A, Baumgart F, Berti M, Franzini G, Stoyanova S. 2014. Cataloging for a billion word library of Greek and Latin. In DATECH 2014: Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage.

Smith N. 2007. An architecture for a distributed library incorporating open-source critical editions. Retrieved from https://wiki.digitalclassicist.org/OSCE_Smith_Paper. 2015, February 19.

Smith N. 2014. Test suite to validate compliance of CTS instances with the CTS API. Retrieved from https://github.com/cite-architecture/ctsvalidator. 2015, February 19.

Tiepmar J, Teichmann C, Heyer G, Berti M and Crane G. 2013. A new Implementation for Canonical Text Services. in Proceedings of the 8th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH).

Project Website of this implementation http://www.urncts.de.

Deutsches Text Archiv http://www.deutschestextarchiv.de/

Parallel Bible Corpus http://paralleltext.info/data/all/

Perseus-CTS/XML https://github.com/PerseusDL/canonical/tree/master/CTS_XML_TEI/perseus

Appendix: Summaries of the Workshop Presentations

Michal Křen: **Recent Developments in the Czech National Corpus**

> The paper gives an overview of current status of the Czech National Corpus project. It covers all important aspects of its activities being carried out within the research infrastructure framework: compilation of a variety of different corpora (most prominently written, spoken, parallel and diachronic), morphological and syntactic annotation, development of tools for internal data processing and work flow management, development of user applications and providing user services. Finally, an outline of future plans is presented.

Dan Tufiş, Verginica Barbu Mititelu, Elena Irimia, Stefan Dumitrescu, Tiberiu Boros, Horia Nicolai Teodorescu: **CoRoLa Starts Blooming – An update on the Reference Corpus of Contemporary Romanian Language**

> This article reports on the ongoing CoRoLa project, aiming at creating a reference corpus of contemporary Romanian, opened for on-line free exploitation by researchers in linguistics and language processing, teachers of Romanian, students. We invest serious efforts in persuading owners of IPR on relevant language data to join us and contribute the project with selections of their text and speech repositories. The project is coordinated by two Computer Science institutes, but enjoys cooperation and consulting from professional linguists. We foresee a corpus of more than 500 million word forms, including also about 300 hours of oral texts. The corpus (covering all functional styles of the language) will be pre-processed and annotated at several levels, and also documented with standardized metadata.

Piotr Bański, Joachim Bingel, Nils Diewald, Elena Frick, Michael Hanl, Marc Kupietz, Eliza Margaretha, Andreas Witt: **KorAP – an open-source corpus-query platform for the analysis of very large multiply annotated corpora**

> We present KorAP, the new open-source analysis platform for large corpora, a deliverable of a project concluded in June 2015 at the Institut für Deutsche Sprache in Mannheim. We overview the background for the project, its goals, and the architecture of the system, including the way it is meant to handle richly annotated textual data and facilitate the use of virtual collections, as well as the way it implements ISO CQLF (Corpus Query Lingua Franca, a nascent standard of ISO TC37 SC4 that KorAP provides a reference implementation for).

Hanno Biber, Evelyn Breiteneder: **Large Corpora and Big Data. New Challenges for Corpus Linguistics**

The "AAC – Austrian Academy Corpus" is a German language digital text corpus of more than 500 million tokens. This historical text corpus is annotated in XML formats and constitutes a large text source for research into various linguistic areas. Several of the research questions relevant for corpus linguistics are also determined by latest developments in the fields of big data research so that new challenges for corpus linguistics have to be faced. The AAC has a primary research aim to develop language resources for computational philology and the careful study of texts by making use of corpus research methodologies. Large digital text corpora need to be structured in a systematic way for these purposes. Corpus based digital text studies and similar analytical procedures are among other parameters also determined by the descriptive and visual potential of information representation in various formats. The digital representation systems of linguistic data need to take the specific design issues into account for the processes of creating, generating and analyzing large corpora and related structures of information by transforming and interpreting the language data.

Sebastian Buschjäger, Lukas Pfahler, Katharina Morik: **Discovering Subtle Word Relations in Large German Corpora**

With an increasing amount of text data available it is possible to automatically extract a variety of information about language. One way to obtain knowledge about subtle relations and analogies between words is to observe words which are used in the same context. Recently, Mikolov et al. proposed a method to efficiently compute Euclidean word representations which seem to capture subtle relations and analogies between words in the English language. We demonstrate that this method also captures analogies in the German language. Furthermore, we show that we can transfer information extracted from large non-annotated corpora into small annotated corpora, which are then, in turn, used for training NLP systems.

Johannes Graën, Simon Clematide: **Challenges in the Alignment, Management and Exploitation of Large and Richly Annotated Multi-Parallel Corpora**

The availability of large multi-parallel corpora offers an enormous wealth of material to contrastive corpus linguists, translators and language learners, if we can exploit the data properly. Necessary preparation steps include sentence and word alignment across multiple languages. Additionally, linguistic annotation such as part-of-speech tagging, lemmatisation, chunking, and dependency parsing facilitate precise querying of linguistic properties and can be used to extend word alignment to sub-sentential groups. Such highly interconnected data is stored in a relational database to allow for efficient retrieval and linguistic data mining, which may include the statistics-based selection of good example sentences. The varying information needs of contrastive linguists require a flexible linguistic query language for ad

hoc searches. Such queries in the format of generalised treebank query languages will be automatically translated into SQL queries.

Stefan Evert, Andrew Hardie: **Ziggurat: A new data model and indexing format for large annotated text corpora**

The IMS Open Corpus Workbench (CWB) software currently uses a simple tabular data model with proven limitations. We outline and justify the need for a new data model to underlie the next major version of CWB. This data model, dubbed Ziggurat, defines a series of types of data layer to represent different structures and relations within an annotated corpus; each such layer may contain variables of different types. Ziggurat will allow us to gradually extend and enhance CWB's existing CQP-syntax for corpus queries, and also make possible more radical departures relative not only to the current version of CWB but also to other contemporary corpus-analysis software.

Roland Schäfer: **Processing and querying large web corpora with the COW14 architecture**

In this paper, I present the COW14 tool chain, which comprises a web corpus creation tool called texrex, wrappers for existing linguistic annotation tools as well as an online query software called Colibri2. By detailed descriptions of the implementation and systematic evaluations of the performance of the software on different types of systems, I show that the COW14 architecture is capable of handling the creation of corpora of up to at least 100 billion tokens. I also introduce our running demo system which currently serves corpora of up to roughly 20 billion tokens in Dutch, English, French, German, Spanish, and Swedish.

Jochen Tiepmar: **Release of the MySQL-based implementation of the CTS protocol**

In a project called "A Library of a Billion Words", we needed an implementation of the CTS protocol that is capable of handling a text collection containing at least 1 billion words. Because the existing solutions did not work for this scale or were still in development I started an implementation of the CTS protocol using methods that MySQL provides. Last year we published a paper that introduced a prototype with the core functionalities but without being compliant with the specifications of CTS. The purpose of this paper is to describe and evaluate the MySQL based implementation now that it is fulfilling the specifications version 5.0 rc.1 and mark it as finished and ready to use.