LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

RĪKU KOPA LATVIEŠU VALODAS SEMANTIKAS ANALĪZEI

PROMOCIJAS DARBS

datorzinātņu doktora (*Dr. sc. comp.*) zinātniskā grāda iegūšanai

Nozare: datorzinātne

Apakšnozare: datoru un sistēmu programmatūra

Autors: **Pēteris Paikens**

Vadītājs: Dr. comp. sci. prof. Guntis Bārzdiņš

RĪGA, 2017

Promocijas darbs izstrādāts LU Matemātikas un Informātikas Institūtā no 2011. gada līdz 2017. gadam.

IEGULDĪJUMS TAVĀ NĀKOTNĒ

Darbs ir izstrādāts ar Eiropas Sociālā Fonda atbalstu projektā "Atbalsts doktora studijām Latvijas Universitātē".

Darba forma: publikāciju kopa Datorzinātnē, apakšnozarē Datoru un sistēmu programmatūra.

Vadītājs: Dr. comp. sci. prof. Guntis Bārzdiņš

Recenzenti:

Promocijas darba aizstāvēšana notiks Latvijas Universitātes Datorzinātnes nozares promocijas padomes atklātā sēdē _____, Latvijas Universitātes Matemātikas un Informātikas Institūtā (Rīga, Raiņa bulv. 29, telpa _____).

Ar promocijas darbu un tā kopsavilkumu var iepazīties Latvijas Universitātes bibliotēkā (Kalpaka bulv. 4, Rīga).

Promocijas padomes priekšsēdētājs: Jānis Bārzdiņš

Anotācija

Promocijas darba pētījuma priekšmets ir automātiskas teksta analīzes metodes, apskatot visus dabiskās valodas apstrādes līmeņus, kas nepieciešami teksta semantiskai analīzei, īpaši pievēršoties risinājumiem, kuri trūka latviešu valodas teksta analīzei. Darbs ir izstrādāts 5 gadu laikā LU MII 4 pētījumu projektu un 2 valsts pētījumu programmu ietvaros.

Darbā tiek aprakstītas autora realizētās metodes latviešu valodas nosaukto entitāšu atpazīšanai un piesaistei reālijām. Zināšanu formālās reprezentācijas vajadzībām ir izveidota FrameNet ontoloģija personu un organizāciju datu un attiecību modelēšanai.

Darbā ir piedāvāts un realizēts latviešu valodas morfoloģiskās struktūras formāls modelis ar plašu pārklājumu, kas ir piemērots patvaļīga teksta analīzei. Darbā ir apskatītas autora realizētās metodes latviešu valodas morfosintaktiskajai analīzei un realizēts neironu tīklu risinājums daudznozīmības novēršanai. Izstrādātais modelis ir aprobēts praksē vairākos projektos un dabiskās valodas rīku izstrādē.

Tāpat darbā ir piedāvāta un realizēta arhitektūra informācijas izguves rīku kopai. Pētīto metožu praktiskai aprobācijai darba gaitā ir izveidots informācijas izguves un zināšanu bāzes aizpildes sistēmas prototips faktu izguvei no latviešu valodas ziņu tekstiem. Šis prototips ir aprobēts ziņu aģentūrā LETA latviešu valodai, kā arī rezultāti ir novērtēti kontekstā ar labakajiem angļu valodas teksta analīzes rezultātiem.

Abstract

This work contains reasearch results on algorithms, resources and tools required for semantic text analysis, with a particular focus on filling in the gaps required for semantic analysis of Latvian language. This work has been developed during the last 5 years in University of Latvia Institute of Mathematics and Computer Science in 4 research projects and 2 state research programmes.

This work describes methods developed by the author for Latvian named entity recognition and linking with real world entities. A FrameNet ontology has been developed for formal knowledge representation and modeling person and organization attributes and relations.

A formal model of Latvian morphology is proposed and implemented in this work, adapted for wide coverage text analysis. This work covers methods for morphosyntactic tagging of Latvian developed by author, introducing a neural network solution for resolving ambiguity. The developed morphology model is approbated in multiple research projects and natural language tools.

As a part of this work, an architecture for an information extraction system and an entity-centric knowledge base is proposed and implemented, integrating the researched methods. This concept is validated on a prototype system for biographic data extraction from Latvian newswire data in news agency LETA and evaluated in context with best results in shared task competions for English knowledge base population.

Promocijas darbā iekļautās autora zinātniskās publikācijas recenzētos žurnālos un starptautisku konferenču materiālos

Darbā aprakstīto pētījumu rezultāti ir publicēti 14 recenzētos starptautisku konferenču rakstos, no kuriem septiņi ir iekļauti SCOPUS vai Thomson Reuters ISI Web of Science datubāzēs:

1. Paikens, P. (2007). **Lexicon-based morphological analysis of Latvian language.** In *Proceedings of 3rd Baltic Conference on Human Language Technologies (HLT 2007)*.

2. Paikens, P. and Grūzītis, N. (2012). **An implementation of a Latvian resource grammar in Grammatical Framework.** In *Eighth international conference on Language Resources and Evaluation (LREC 2012)* (Web of Science).

3. Gruzitis, N., Paikens, P., and Barzdins, G. (2012). **FrameNet resource grammar library for GF.** In *Controlled Natural Language*, lpp. 121–137. Springer Berlin Heidelberg (SCOPUS, Web of Science).

4. Paikens, P., Auziņa, I., Garkāje, G., and Paegle, M. (2012). **Towards named entity annotation of Latvian National Library corpus.** In *Baltic HLT*, lpp. 169–175 (SCOPUS, Web of Science).

5. Pretkalniņa, L., Paikens, P., Grūzītis, N., Rituma, L., and Spektors, A. (2012). **Making historical Latvian texts more intelligible to contemporary readers.** In *Proceedings of the workshop "Adaptation of Language Resources and Tools for Processing Cultural Heritage Objects" at the Eight International Conference on Language Resources and Evaluation (LREC'12).*

6. Paikens, P. (2013). **Automātiskas morfoloģiskas anotācijas izmantojums.** In *Vārds un tā pētīšanas aspekti 2013*.

7. Paikens, P., Rituma, L., and Pretkalniņa, L. (2013). **Morphological analysis with limited resources: Latvian example.** In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013) NEALT Proceedings Series 16*, lpp. 267–278, Oslo.

8. Znotiņš, A. and Paikens, P. (2014). **Coreference resolution for Latvian.** In *Proceedings of LREC 2014, Ninth International Conference on Language Resources and*

Evaluation (Web of Science).

9. Bārzdiņš, G., Goško, D., Rituma, L., and Paikens, P. (2014). **Using C5.0 and exhaustive search for boosting frame-semantic parsing accuracy.** In *Proceedings of LREC 2014, Ninth International Conference on Language Resources and Evaluation* (Web of Science).

10. Paikens, P. (2014). **Latvian newswire information extraction system and entity knowledge base.** In *Human Language Technologies – the Baltic Perspective* (SCOPUS, Web of Science).

11. Bārzdiņš, G., Paikens, P., Goško, D. (2015). **Riga: from FrameNet to Semantic Frames with C6.0 Rules.** In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval)*.

12. Spektors, A., Auzina, I., Dargis, R., Gruzitis, N., Paikens, P., Pretkalnina, L., Rituma, L., Saulite, B. (2016). **Tezaurs.lv: the Largest Open Lexical Database for Latvian** In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC)*.

13. Paikens, P. (2016). **Deep Neural Learning Approaches for Latvian Morphological Tagging.** In *Proceedings of Human Language Technologies – the Baltic Perspective* (SCOPUS, Web of Science).

14. Paikens, P., Barzdins, G., Mendes, A., Ferreira, D., Broscheit, S., Almeida, M. S. C., Miranda, S., Nogueira, D., Balage, P., and Martins, A. F. T. (2016). **SUMMA at TAC knowledge base population task 2016.** In *Proceedings of the Ninth Text Analysis Conference (TAC 2016)*.

Promocijas darba autora personiskais ieguldījums

Autori	Publikācija	Ieguldī-jums (%)	Ieguldījuma apraksts
P. Paikens	*Lexicon-based morphological analysis of Latvian language*	100	Idejas izstrāde un aprakstītās sistēmas praktiskā realizācija.
P. Paikens N. Grūzītis	*An implementation of a Latvian resource grammar in Grammatical Framework*	60	Piedalīšanās idejas izstrādē, aprakstītā risinājuma praktiskā realizācija.
N. Grūzītis P. Paikens G. Bārzdiņš	*FrameNet resource grammar library for GF*	40	Piedalīšanās idejas izstrādē, daļa no eksperimentālās realizācijas.
P. Paikens I. Auziņa G. Garkāje M. Paegle	*Towards named entity annotation of Latvian National Library corpus*	50	Piedalīšanās koncepcijas un anotēšanas standarta veidošanā, automātiskās tagošanas rīku pētījumi un realizācija, eksperimentālais novērtējums.
L. Pretkalniņa P. Paikens N. Grūzītis L. Rituma A. Spektors	*Making historical Latvian texts more intelligible to contemporary readers*	30	Piedalīšanās koncepcijas sagatavošanā, daļa no praktiskās sistēmas izstrādes.
P. Paikens	*Automātiskas morfoloģiskas anotācijas izmantojums*	100	Idejas izstrāde un apraksta sagatavošana.
P. Paikens L. Rituma L. Pretkalniņa	*Morphological analysis with limited resources: Latvian example*	70	Koncepcijas izstrāde, tagošanas rīku pētījumi un realizācija, eksperimenti un analīze.
A. Znotiņš P. Paikens	*Coreference resolution for Latvian*	30	Piedalīšanās ideju izstrādē, praktiskajā realizācijā un publikācijas sagatavošanā.
G. Bārzdiņš D. Goško L. Rituma P. Paikens	*Using C5.0 and exhaustive search for boosting frame-semantic parsing accuracy*	40	Piedalīšanās ideju izstrādē, daļa no praktiskās realizācijas, sistēmas eksperimentālais novērtējums un analīze.
P. Paikens	*Latvian newswire information extraction system and entity knowledge base*	100	Koncepcijas izstrāde, sistēmas praktiskā realizācija un apraksta sagatavošana.
G. Bārzdiņš P. Paikens D. Goško	*Riga: from FrameNet to Semantic Frames with C6.0 Rules*	40	Piedalīšanās ideju izstrādē, daļa no praktiskās realizācijas, dalība eksperimentos un analīzē.

A. Spektors I. Auziņa R. Darģis N. Grūzītis P. Paikens L. Pretkalniņa L. Rituma B. Saulīte	*Tezaurs.lv: the Largest Open Lexical Database for Latvian*	25	Piedalīšanās ideju izstrādē, dalība sistēmas praktiskajā realizācijā un publikācijas sagatavošanā.
P. Paikens	*Deep Neural Learning Approaches for Latvian Morphological Tagging*	100	Idejas izstrāde, aprakstītās sistēmas praktiskā realizācija un eksperimentālais novērtējums.
P. Paikens G. Bārzdiņš A. Mendes D. Ferreira S. Broscheit M. Almeida S. Miranda D. Nogueira P. Balage A. Martins	*SUMMA at TAC knowledge base population task 2016*	25	Piedalīšanās koncepcijas izstrādē, praktiskā realizācija un eksperimentālais novērtējums daļai no aprakstītās sistēmas, dalība publikācijas sagatavošanā.

Līdzautoru piekrišana publikāciju izmantošanai promocijas darbā

Ar šo tiek apliecināts, ka augstāk minēto publikāciju līdzautori piekrīt publikāciju iekļaušanai Pētera Paikena promocijas darbā.

_____ (N. Grūzītis)

_____ (G. Bārzdiņš)

_____ (I. Auziņa)

_____ (G. Garkāje)

_____ (M. Paegle)

_____ (L. Pretkalniņa)

_____ (L. Rituma)

_____ (A. Spektors)

_____ (A. Znotiņš)

_____ (D. Goško)

_____ (R. Darģis)

_____ (B. Saulīte)

PUBLIKĀCIJA I

Lexicon-based morphological analysis of Latvian language

Proceedings of 3rd Baltic Conference on Human Language Technologies (HLT 2007), 2007.

LEXICON-BASED MORPHOLOGICAL ANALYSIS OF LATVIAN LANGUAGE

Pēteris Paikens
University of Latvia, Institute of Mathematics and Computer Science (Riga, Latvia)

Abstract

This paper describes a practical solution for lexicon-based morphological analysis of Latvian language. As it is a flexive language, the core of this system is an implementation of word inflection based on a stem and its properties as listed in the lexicon. The main advantage of the described solution over similar implementations is augmenting the lexicon with methods for word derivation from related word stems, significantly increasing the recognition rate. The implemented system is able to provide full morphological detail for 96 % words of unrestricted Latvian language texts, even when using a rather limited lexicon of 25,000 word stems. For remaining unknown words, the system is extended with heuristics for recognising proper names, and determining verb and noun flexive forms based on ending, allowing a good quality guess for the linguistic properties of words that are not included in the lexicon. Such wide coverage allows the solution to be used in other linguistic tools as a transparent and robust layer for analysing word properties.

Keywords: morphology, part of speech, tagging, dictionary.

1. Introduction

For flexive languages, like Latvian language, morphological analysis and/or stemming often is the first required step in any text analysis process. However, there is a lack of publicly available morphological analysis tools for Latvian language, and most linguistic solutions and research – for example, current semantic projects in University of Latvia (Bārzdiņš et al 2007a) tend to use custom dictionaries to match exact word forms (instead of word stems) with the required information. Such approaches work acceptably within the domain covered by the dictionary, but for analysis of unrestricted corpora there will always be significant portion of words that are not included in the dictionary.

This paper describes a currently developed system that aims to provide a robust and extensible solution for morphological analysis for unrestricted corpora, in order to have an available solution to facilitate further analysis of Latvian language.

2. Solutions used currently

Most research purposes seem to use hard-coded dictionaries for the first steps of analysis, disregarding morphology entirely. Often the main reason for this is adaptation of tools originally designed for analysis of English language corpora, which don't

support flexive word forms, and treat all variations of a single stem as entirely separate words. This hampers linguistic analysis, but is often accepted due to technological difficulties. Availability of morphological tools for Latvian language might relieve this issue, and facilitate easy testing of other computational linguistics tasks on various corpora.

An interesting approach was seen (Krūze-Krauze 1998) that attempts to generate all possible words by defining formal grammar rules that govern the ways how different morphemes may combine together to make a single word. However, the system currently is in a dead-end, reaching an unmaintainable size of rules and special cases, and still fails to recognize a large portion of words, especially words adopted from other languages such as Greek or English.

There have also been a number of previous attempts of morphological analysis as well, with similar methods as described in the following section, but without much success, as the achieved coverage tends to be good enough for handmade research corpora, but is not acceptable for unrestricted text analysis. The current development attempts to include the experience of these previous attempts, especially in various heuristics for handling exceptional cases.

3. Lexicon-based analysis

In Latvian language, most word classes – nouns, verbs and adjectives – are flexive, consisting of an (almost) unchanging stem, and an ending that specifies various grammatical properties of the word. The exact endings vary depending on the basic stem, but almost all words in Latvian language can be split in a limited number (23 for the current implementation) of groups where every word form can be generated automatically. There are some irregular words, though (for example, *"būt" – "to be"*), but their number is rather limited, so they can be included manually in the lexicon.

Morphological analysis can be done by using this database of the endings in order to generate all possible variations where the ending is equal to the last letters of the analysable word, and looking up the remaining letters within the available dictionary of stems for a possible match. See Figure 1 for the workflow process. A major issue here is the stem changes that sometimes happen (Ceplīte et al 1991) in various word forms. This is analysed with a custom heuristics that lists all the cases occurring in literary Latvian language.

Ambiguity in this part of analysis is unavoidable, as there are many words in Latvian language where the part of speech details can be determined only in syntactic context. For example, word *"roku"* (given as an example in Figure 1) can be a word form of *"roks"* ("rock music"; masculine noun), various word forms in different cases of *"roka"* ("hand"; feminine nound), or a form of *"rakt"* ("to dig"; verb).

Thus the core data of the system is a dictionary of lexical units, containing word stems grouped in morphological types, and any information about these word stems that should be passed on in the results. As the goal of the system is to provide foundation for further syntactic and semantic analysis, a major focus is to provide extensibility for lexicon data, allowing the users to amplify the lexicon with any additional data relevant for the problem at hand. Current uses for additional information include verb transitivity information for the main lexicon, and semantic ontology groupings for a small research lexicon.

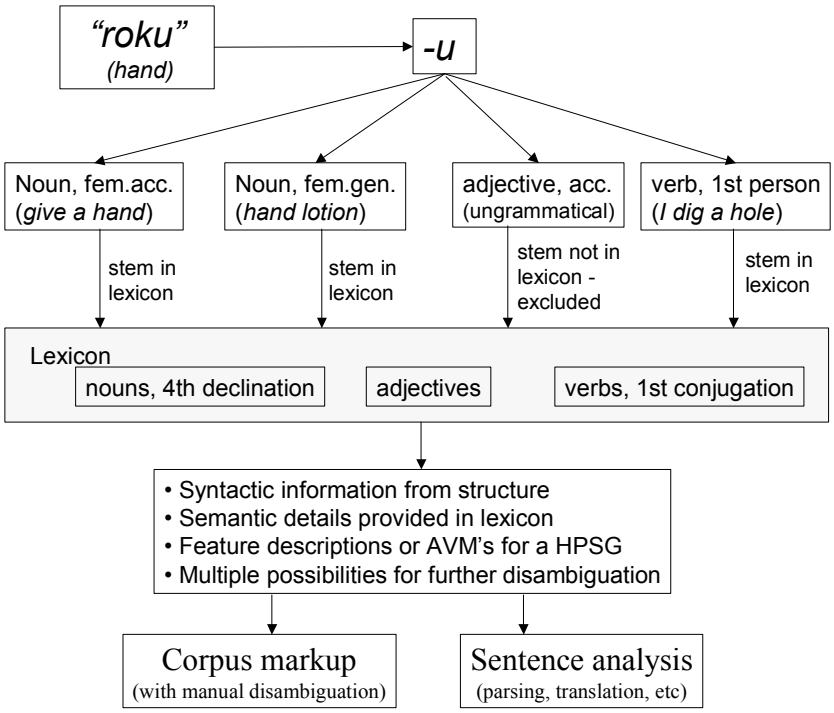


Figure 1 – Workflow of analysis process

4. Extending the lexicon

Of course, such analysis methods are limited by the extent of the dictionary used. This problem is further complicated by the fact that digitally available dictionaries in Latvian language are comparatively small. This research used a lexicon of approximately 27,000 word stems, based on an electronic version of an inverse dictionary (Soida, 1970). Initial analysis has showed that such a lexicon can cover 85–90% of unrestricted text[1]. This ratio is quite low and clearly not enough for practical purposes even with larger dictionaries. This has always been an issue for previous such developments, and was usually tackled by targeting a specific domain of text, and manually adapting the dictionary for that target domain.

However, analysis of missed words in these corpora (see Table 1) shows that most of them are in some way derived from other words that are included in our lexicon. This can be expected, as new words in Latvian language, as many others, often are formed by extending already existing words with similar meaning, or metaphorically using these words in a different meaning.

[1] Novels „Plāns ledus" and „Sofijas pasaule" („Sophie's world") were used as test corpora for this and other statistics.

Table 1 – Words not found in lexicon

Derivation via prefix	63%
Derivation via infix	2%
Compound nouns	2%
Proper nouns	5%
Reflexive verbs	11%
Not related to words in lexicon	15%
Erroneous words	2%

These derivation types can be all automatically analysed by attempting to match the word to stems in the existing lexicon. There is a rather small number of prefixes and infixes used in Latvian language, and it is computationally easy to try them all. Not all of these usages are valid, so this cannot be used for spell-checking type of solutions, but in the case when a new word is encountered in a corpus, and it matches some other in the lexicon with a derivation rule, we can be reasonably sure that we have encountered a correct (but new) word, and the appropriate part of speech information can be extracted.

On the other hand, if additional information is expected from the lexicon – for example, the inclusion of word senses in some ontology – then it is not safe to include this information. The derived words usually are related in meaning, but the nature of this relation may vary greatly, in some cases the relation is only metaphorical. Newly found words can then be used to enhance the lexicon, but this would require at least some review and approval for all non-morphological information.

5. Treatment of unrecognised words

The final part of this application, which gets used if the previous methods have failed to relate the word to some entry in our lexicon, is a heuristic for part-of-speech tagging based on the last letters of the analysable word.

A core part of the initial morphological analysis system is an exhaustive list of all endings for flexive word classes, linked with the morphological and part-of-speech information that each ending can represent. This list can also be used without looking at the lexicon – this will yield a lot of ambiguous possibilities, but still this process can usually exclude the majority of variations. For example, gender of the noun can usually be determined in this way, but several possibilities for number and case may remain. Many word forms in Latvian language can be uniquely identified in this way (Nau 1998), for example, verb participle forms. Comparison of the results with annotated corpus (Levāne 2001) shows that the generated variants always include the proper tagging – several mismatches were found in the comparison, but they were all found to be errors in the manual tagging.

For cases with more ambiguity, frequency distribution tables are used to determine the most likely word forms, or, alternatively, all possibilities are included, and ambiguity resolution can be left to the syntax part of analysis, since it is very likely that the specific form can be determined from grammatical requirements of case/number/gender agreement.

6. Technical implementation

The main consideration in the technical implementation has been compatibility with various other tools that may be used together with this morphological module. There are

two independent workflows implemented. One way of interaction is a batch-annotation mode, where a corpus is transformed into an XML file with the (ambiguous) morphological analysis results appended to each word. The other way is an online analysis service, which can be repeatedly queried for analysis of particular words, and can be accessed by other applications.

Current implementation is done in Java, with an interface that can be called from Prolog applications. Analysis performance is about 16,000 words per second on a standard desktop PC.

The lexicon and all other data are stored in XML files. For corpus tagging currently a custom XML format is used, but work is underway to use Tiger-XML formats everywhere to improve interoperability with different tools used in computational linguistics for other languages.

8. Evaluation and further developments

Current status of this development is stable enough to use it as a transparent layer of unrestricted text corpus analysis, extracting morphological and part-of-speech information for nearly all words within it. Increasing lexicon size will help decrease ambiguity of this analysis, but is not absolutely necessary. Using a modestly sized lexicon of 27,000 word stems and the abovementioned word derivation rules, 96 % of words in test corpora can be fully analysed, and for the remaining 4 % words all the part-of-speech possibilities are provided – which usually include two or three possibilities for grammatical case of the word.

Current usage of the system includes University of Latvia projects in semantic ontology (Bārzdiņš et al 2007b), and in corpora extraction from the Internet (Džeriņš et al 2007). The solution is considered to be publicly available for any research purposes upon contacting the author.

Details of all these tasks, naturally, can be developed further for improved results. In particular, further developments would include the following tasks:

- Adapting existing solutions for entity name recognition in order to treat proper nouns in a more precise way. This issue is common for many other languages, so solutions designed for English could be used.
- Changing the XML formats used to match Tiger-XML is underway.
- Support for morphological analysis of transliterated text[2], to improve coverage for corpora extracted from Internet. There are solutions that attempt to transform words from transliterated to proper form, but they are naturally ambiguous, and this ambiguity resolution can be improved if the morphological analysis is done directly on transliterated text.
- There are some ways of word derivation that occur less frequently and are not yet included – diminutive forms would be a good candidate for inclusion, as such forms occur in the language, but tend to be excluded from dictionaries.

[2] Spelling of text with exclusively latin characters, i.e. spelling *zaķīši* as *zakjiishi*, which is occasionally encountered on websites, blogs and Internet discussion forums.

12. References

Bārzdiņš, Guntis; Grūzītis, Normunds; Nešpore, Gunta; Saulīte, Baiba 2007a. Dependency-Based Hybrid Model of Syntactic Analysis for the Languages with a Rather Free Word Order. In: Nivre, J.; Kaalep, H.; Muischnek, K.; Koit, M. (eds.) *NODALIDA 2007 conference proceedings*. Tartu: University of Tartu.

Bārzdiņš, Guntis; Grūzītis, Normunds; Levāne-Petrova, Kristīne; Nešpore, Gunta; Saulīte, Baiba 2007b. A deep discourse representation structure for theme-rheme and anaphora resolution. In: *Human Language Technologies 2007 conference proceedings.*

Ceplīte, Brigita; Ceplītis, Laimdots 1991. Latviešu valodas praktiskā gramatika. Rīga: Zvaigzne.

Džeriņš, Jānis; Džonsons, Kristaps 2007. Harvesting national language text corpora from the Web. In: *Human Language Technologies 2007 conference proceedings.*

Krūze-Krauze, Baiba 1998. Datorizēta latviešu valodas morfēmiski morfoloģiskā analīze. Rīga: Latvijas Universitāte.

Levāne, Kristīne 2001. Paula Bankovska romāna ”Plāns ledus” pirmās nodaļas morfoloģiskā anotēšana un statistiskā analīze. Rīga: Latvijas Universitāte.

Nau, Nicole 1998. Latvian. Newcastle: LINCOM Europa.

Soida, Emīlija; Kļaviņa, Sarma 1970. Latviešu valodas inversā vārdnīca. Rīga: LVU.

PĒTERIS PAIKENS is a junior researcher of Institute of Mathematics and Computer Science, University of Latvia. He received his bachelor degree in computer science at University of Latvia. His research interests include corpus analysis, syntax analysis for Latvian language and formal grammar development. E-mail: PeterisP@gmail.com.

PUBLIKĀCIJA II

An implementation of a Latvian resource grammar in Grammatical Framework

Eighth international conference on Language Resources and Evaluation (LREC 2012), 2012.

An implementation of a Latvian resource grammar in Grammatical Framework

Pēteris Paikens, Normunds Grūzītis
Institute of Mathematics and Computer Science, University of Latvia
Raina blvd. 29, Riga, LV-1459, Latvia
E-mail: peterisp@ailab.lv, normundsg@ailab.lv

Abstract

This paper describes an open-source Latvian resource grammar implemented in Grammatical Framework (GF), a programming language for multilingual grammar applications. GF differentiates between concrete grammars and abstract grammars: translation among concrete languages is provided via abstract syntax trees. Thus the same concrete grammar is effectively used for both language analysis and language generation. Furthermore, GF differentiates between general-purpose resource grammars and domain-specific application grammars that are built on top of the resource grammars. The GF resource grammar library (RGL) currently supports more than 20 languages that implement a common API. Latvian is the 13th official European Union language that is made available in the RGL. We briefly describe the grammatical features of Latvian and illustrate how they are handled in the multilingual framework of GF. We also illustrate some application areas of the Latvian resource grammar, and briefly discuss the limitations of the RGL and potential long-term improvements using frame semantics.

Keywords: computational grammar, language generation, Grammatical Framework

1. Introduction

The long-term research behind this paper is aimed at semantic parsing of Latvian and natural language generation in Latvian. While our former focus has been on developing language resources and tools that can be primarily used for language analysis, in this paper, we describe a recent open-source implementation of a Latvian resource grammar that can be effectively used for both language *analysis* and language *generation*. We have implemented this resource grammar in Grammatical Framework (GF), a toolkit and formalism for rapid development of *multilingual* grammar applications (Ranta, 2011).

Latvian is an Indo-European language, a member of the Baltic language group, one of the official EU languages. In terms of speakers, it is a relatively small language (about 1.5 million native speakers and about 0.5 million non-native speakers). It uses a Latin-based alphabet that in almost all cases provides a one-to-one mapping between letters and phonemes. The general grammatical characteristic of Latvian is that it is a highly inflective language with a relatively free word order.

Large annotated language resources, such as treebanks and parallel corpora of various domains that would facilitate statistical parsing and generation, are scarce for Latvian – reusability of the developed computational grammars across general and domain-specific use-cases and across languages is very important.

A fairly successful attempt developing a robust, wide coverage partial parser of Latvian has been in lines with the dependency grammar approach (Bārzdiņš et al., 2007; Pretkalniņa et al., 2011). Other computational grammars of Latvian have been crafted for the needs of various machine translation systems (Skadiņa et al., 2007; Greitāne, 1997) and grammar checking tools (Deksne & Skadiņš, 2011). However, there has been no general-purpose wide-coverage computational grammar available for generating Latvian sentences.

Although dependency-based grammars allow for robust and effective parsing they lack the potential of language generation. This is the strength of phrase structure grammars, e.g. categorial grammars that link the surface structure with the underlying semantic representation. Among other features, GF essentially is an effective implementation of the categorial grammar approach.

2. Grammatical Framework

GF facilitates reusability by splitting the grammar development in two levels:

1. a general purpose *resource grammar* that covers a wide range of morphological features and syntactic structures,
2. and domain specific *application grammars* defining semantic structures and the subset of natural language that is used within a particular domain.

This allows developing and testing of the morphological and syntactic complexity once, which can be afterwards reused in multiple domains and in different usage scenarios without in-depth knowledge about the particular language and without the need to implement a large list of nuanced exceptional cases. The use-cases are ranging from controlled languages (e.g. dialogue systems and interfaces to formal languages) to domain-specific machine translation applications (e.g. speech-to-speech travel assistants).

GF differentiates not only between general-purpose resource grammars and domain-specific application grammars, but also between abstract syntax and concrete syntax. The abstract syntax captures the semantically relevant structure of language, defining grammatical categories and functions for building trees (Ranta, 2011). Concrete syntax defines the linearization of the abstract tree structures at the surface level. Translation among languages (concrete grammars) is provided via abstract syntax trees.

Note that in the GF grammar development there is no

concept of a language pair or a translation direction. Also there is no common semantic interlingua. Instead there are many application- and domain-specific interlinguas, and the concrete syntax can be built (but not necessarily) on top of the common resource grammar API.

The GF resource grammar library (Ranta, 2009), or RGL for short, currently supports more than 20 languages[1] that implement the common API. Latvian is among 13 (out of 23) official EU languages that are supported.

The common API specifies about 60 hierarchical grammatical categories and nearly 500 syntactic construction functions (including structural words and parameters used in the abstract trees)[2]. The large number of functions is still manageable from the application grammar developer perspective: due to extensive overloading, most of the functions are arranged in about 35 overload groups. Apart from the syntactic functions, there are also about 15 groups of lexical construction functions (the exact number of overloaded paradigms varies among languages; see Table 1 for a simplified example).

3. Morphology

Morphology plays an important part in grammatical analysis of Latvian, as there are many[3] inflected wordforms possible for each lemma: about 10 noun/pronoun forms, about 40 verb forms (excluding about 160 participle forms whose syntactic function is that of adjectives), and more than 100 adjective forms. Still, a lot of analytical wordforms are also used (e.g. analytical verb forms and prepositional phrases).

We have developed a GF morphology module for the full Latvian language by transforming and improving a previously developed morphological analyzer (Paikens, 2007) to the GF language, taking into account the language generation aspects. In particular, we have implemented a set of functions that detail the lemmatization and palatalization that occurs in Latvian, and an exhaustive list of word ending tables used in each paradigm. In the result, the Latvian GF morphology module and the analyzer by Paikens (2007) are quite different from the application point of a view. The latter one is designed as a highly robust analyzer for maximum coverage of an unrestricted text and is not appropriate for the generation needs as it suffers from overgeneration. However, the GF module is designed for high precision within a known lexical domain.

In Table 1, a simplified inflectional paradigm for Latvian nouns of the 5th declension is given along with the corresponding tiny fragment from the abstract grammar. A similar approach has been used for implementing morphology in GF for other inflective languages, e.g. Russian[4] (Khegai, 2006).

All the possible wordforms (linearizations) of a particular Latvian noun are given in Table 2 along with possible linearizations of the corresponding English noun.

Note that in the public API, the specific internal functions (operations) that deal with the lexical paradigms are hidden by overloaded functions (e.g. `mkN` in the case of nouns).

Common abstract grammar: categories

```
cat N ;
```

Latvian resource grammar: the morphology module

```
param
  Number = Sg | Pl ;
  Gender = Masc | Fem ;
  Case = Nom | Gen | Dat | Acc | Loc ;
  Declension = D1 | D2 | D3 | D4 | D5 | ... ;

oper
  Noun : Type = {
    s : Number => Case => Str ;
    g : Gender
  } ;

  mkNoun : Str -> Noun = \lemma ->
    let decl : Declension = case lemma of {
      ...
      s + "e" => D5 ; -- usually
      ...
    } in mkNoun_Decl lemma decl ;

  mkNoun_Decl : Str -> Declension -> Noun =
    \lemma,decl -> case decl of {
      ...
      D5 => mkNoun_D5 lemma ;
      ...
    } ;

  mkNoun_D5 : Str -> Noun = \lemma ->
    let stem : Str = cutStem lemma
    in {
      s = table {
        Sg => table {
          Nom => stem + "e" ;
          Gen => stem + "es" ;
          Dat => stem + "ei" ;
          Acc => stem + "i" ;
          Loc => stem + "ē"
        } ;
        Pl => table {
          Nom => stem + "es" ;
          Gen => palatalize stem + "u" ;
          Dat => stem + "ēm" ;
          Acc => stem + "es" ;
          Loc => stem + "ēs"
        }
      } ;
      g = Fem
    } ;
```

Latvian resource grammar: API

```
oper mkN = overload {
  mkN : (s : Str) -> N = \n -> lin N (mkNoun n) ;
  mkN : (s : Str) -> Declension -> N = \n,d ->
    lin N (mkNoun_Decl n d) ;
} ;
```

Table 1: A simplified fragment of RGL.

[1] http://www.grammaticalframework.org/lib/doc/status.html

[2] http://www.grammaticalframework.org/lib/doc/synopsis.html

[3] If compared to analytical languages like English or Scandinavian languages.

[4] In terms of grammar, the Slavic language group is the closest branch to the Baltic language group.

Domain-specific lexicon: abstract
`fun sun_N : N ;`
Domain-specific lexicon: Latvian
`lin sun_N = mkN "saule" ;`
Domain-specific lexicon: English
`lin sun_N = mkN "sun" ;`
Parsing into the abstract categories

`>> parse -lang=Lav "sauļu"`
`sun_N`
 	
`>> parse -lang=Eng "suns'"`
`sun_N`

Generating the full inflectional paradigms

`>> linearize -lang=Lav -table sun_N`
`s Sg Nom : saule`
`s Sg Gen : saules`
`s Sg Dat : saulei`
`s Sg Acc : sauli`
`s Sg Loc : saulē`
`s Pl Nom : saules`
`s Pl Gen : sauļu`
`s Pl Dat : saulēm`
`s Pl Acc : saules`
`s Pl Loc : saulēs`
 	
`>> linearize -lang=Eng -table sun_N`
`s Sg Nom : sun`
`s Sg Gen : sun's`
`s Sg Acc : sun`
`s Pl Nom : suns`
`s Pl Gen : suns'`
`s Pl Acc : sun`

Table 2: A sample domain lexicon (a part of an application grammar): its definition and usage.

4. Syntax

We have implemented grammar rules for all the common phrase structures in the conventional style of categorial grammars, basically: noun phrases with agreement rules for adjectives and other modifiers, adjective phrases, and verb phrases with the relevant complements. On one hand, the implemented rules cover only the most common (neutral) ways of expressing these phrases (in terms of word order), excluding several alternative word orderings that are occasionally used for special emphasis (e.g. to indicate the given vs. new information) or for poetic reasons. On the other hand, the grammar includes syntactic construction rules for a full range of dependent clauses and participle clauses used in Latvian language, thus ensuring a wide coverage for generating natural, complex sentences.

In essence, this approach models a subset of the full natural language which relies on rich lexical information about words in a specific domain and on a grammatically correct standard language, gaining high precision while accepting a lower recall rate if analyzing an unrestricted text. From the language generation point of a view, the design goal is that it should be possible to express every valid structure in the most common way, i.e., in the natural/neutral word order – but not necessarily in all the possible word orderings, as there is no well-defined model (for Latvian) for the exact semantic nuances transferred by alternative word order in a more or less unrestricted text[5].

4.1 Clauses

We treat clauses as elements that specify actions – a verb with its arguments – but leaves unspecified the way in which the actions are described. Traditional Latvian linguistics describes clauses in terms of moods and tenses. There are infinitive, indicative, relative[6], debitive[7] and imperative moods, as well as few subtypes of some of them and several types of participles. The relative and debitive moods are Latvian-specific and are used to express the reported speech and necessity or requirement accordingly.

In general, every action can be expressed in any of these moods by using different synthetic verb forms. In the case of a perfect tense, analytical verb forms are used. We have implemented the full set of mood, tense and polarity combinations used in Latvian language, some examples of which are illustrated in Table 3.

Parameters	Example	Translation
`Indicative Present`	zāle ir zaļa	grass is green
`Indicative Past`	zāle bija zaļa	grass was green
`Indicative Anterior Present`	zāle ir bijusi zaļa	grass has been green
`Relative Simultaneous Present`	zāle esot zaļa	[one says that] grass is green
`Debitive Simultaneous Present`	zālei jābūt zaļai	grass has to be green
`Conditional Simultaneous`	zāle būtu zaļa	grass would be green
`Relative Anterior Negated`	zāle neesot bijusi zaļa	[one says that] grass has not been green

Table 3: Examples of mood, tense and polarity variation in Latvian.

The basic abstract (language-independent) syntax used in GF RGL is based on a narrow view of tenses (present, past, future and conditional). This limits the easily (synthetically) available variety in generation of Latvian sentences. In a standard resource grammar, at the sentence level, the verb phrases are used in the indicative mood, however, keeping the other types of moods integrated allows us to reuse the same verb phrase

[5] Although, in the case of a highly controlled Latvian, there is a deterministic model defined by Grūzītis (2010).

[6] http://www.isocat.org/datcat/DC-3836

[7] http://www.isocat.org/datcat/DC-3835

constructing functions in application grammars that need the additional means of expression. This helps also when translating specific (structural) verbs such as 'must', 'might' or 'said' – in a proper translation to Latvian it is often necessary to modify the mood of the dependent clause governed by these verbs instead of including the literal translation of these verbs.
The API interface provided by the resource grammar is as follows:

1. Function `mkCl` (make clause), parameterised by the subject, core verb and any appropriate complements. For example, "`mkCl John_N give_V2 key_N Mary_N`" generates clauses that correspond to all combinations of tense and polarity for using in different kinds of sentences: "*John gives a key to Mary*", "*John has not given a key to Mary*", "*will John give a key to Mary*" etc.
2. Functions to apply such clauses – parameterised by tense, anteriority and polarity. For example, by applying "`mkS pastTense simultaneousAnt positivePol`" to the previously defined clause the specific declarative sentence "*John gave a key to Mary*" is generated.
3. Helper functions for building incomplete clauses that may be needed to form questions, imperative sentences or subclauses.

This structure enforces a clean separation between the actual predicate that is described, and the way in which it is described in a narrative. For example, an application may need to refer to the same action multiple times: first, (hypothetically) to request a confirmation from a user, and afterwards to refer to it as a completed action, requiring a completely different syntactic structure.
In the practical development of user interfaces in Latvian this is almost always done in an unsophisticated way, using simple declarative sentences where the correct wordform can be built easily by regular expressions or similar methods. This results in sentence structures that look clumsy to users, because humans would commonly use a more complicated structure with subclauses.
Such a resource grammar allows applications to express a particular clause once in a standardised way, and then use it in various forms or combine it in complex sentence structures without dealing with the rather complex rules of inflection, agreement and structural changes when using it as a subclause.

4.2. Verb phrases

The agreement rules for complements of multi-argument verbs are implemented by specifying the syntactic valences of each verb – the case or preposition that the relevant complement must or may have.
This presents a challenge for implementing a practical system for Latvian in relatively unrestricted language domains with large lexicons, as currently there is no publicly available syntactic valence dictionary for Latvian, and thus all such verbs would need to be defined manually instead of importing them from some database of verbs with appropriate morpho-syntactic information. However, if (application) grammar users define syntactic valences of verbs that are appropriate to the specific domain, it gives an opportunity to specify (at the same time) also semantic valences, so that the role of each complement can be obtained from the case (or preposition) used, allowing to integrate the grammar with frame semantics, e.g. with the data of FrameNet (Fillmore et al., 2003), or to map the verb valences to domain-specific predicate parameters.
In any case, this lexical information is necessary to ensure correct analysis or synthesis, as verb complement roles (both syntactic and semantic) are mainly defined by their case or preposition. In Table 4 we illustrate this valence mapping of semantic and syntactic roles for three related verbs.

(a) *saņemt* (to receive):

Sem. role	Latvian	English
`Recipient`	`Nominative`	`Subject`
`Theme`	`Accusative`	`Object-1`
`Donor`	`"no" ++ Genitive`	`"from" ++ Object-2`

Mērija saņem atslēgu no Jāņa – Mary receives a key from John

(b) *vajadzēt* (to need):

Sem. role	Latvian	English
`Recipient`	`Dative`	`Subject`
`Theme`	`Accusative`	`Object-1`
`Donor`	`"no" ++ Genitive`	`"from" ++ Object-2`

Mērijai vajag atslēgu no Jāņa – Mary needs a key from John

(c) *dot* (to give):

Sem. role	Latvian	English
`Donor`	`Nominative`	`Subject`
`Theme`	`Accusative`	`Object-1`
`Recipient`	`Dative`	`"to" ++ Object-2`

Jānis dod atslēgu Marijai – John gives a key to Mary

Table 4: Syntactic and semantic role mappings

Note that the examples given in Table 4 correspond to the neutral word order, but the other possible orderings that preserve the same morphological features are also valid in Latvian: "*Mērija no Jāņa saņem atslēgu*", "*no Jāņa atslēgu Mērija saņem*" etc. They convey virtually the same meaning, but the information structure (topic and focus) is different, affecting the further discourse analysis (Grūzītis, 2010).
The syntactic information specific to each of the (a), (b) and (c) verbs in Table 4 is necessary both to choose the proper complement wordform in language generation, and to determine the subject while parsing a sentence. This also means that in the case of verbs that are classified as three-place verbs some complements can be (and often are [8]) omitted while still keeping clear valences.

[8] Preliminary corpus analysis of Latvian verb valences indicates that in about 30% cases one or multiple frame elements are omitted.

This property is relevant to other languages as well[9], and the current GF approach of classifying verbs according to the number of arguments is not sufficient in the long term, especially in the multilingual environment where the syntactic realization of the same verb (concept) can be different across languages.

4.3 Noun and adjective phrases

Noun and adjective phrases are implemented in a straightforward manner as it is typical for inflective languages – the phrase constituent relations are determined from agreement of morphological features.

The treatment of determiners is somewhat interesting: definite and indefinite articles are not used in Latvian, and, in general, there is no difference between definite and indefinite noun phrases (at the surface level). A noun phrase might include an indefinite or demonstrative pronoun, or an adjective that have distinct definite and indefinite forms, however, the given and new information is often indicated implicitly – by rather systematic changes in the neutral word order (Grūzītis, 2010). These formal features can be exploited to ensure the proper translation in a multilingual application. In this regard, the definiteness property is tracked in noun phrases in order to determine the agreement between a noun and an adjective or a participle.

In Latvian, an attribute of a noun can be easily transformed into a (comma-delimited) attributive subclause or vice versa (in most cases). The resource grammar includes full support for deep nesting of such subclauses as they are typically used, for example, in legal texts.

5. Applications

GF has been used for a logic-based Latvian-English application grammar even before the Latvian resource grammar was available, creating a prototype for authoring and verbalizing OWL ontologies in controlled Latvian via Attempto Controlled English and its readily available infrastructure (Grūzītis & Bārzdiņš, 2011; Fuchs et al., 2008). Now it is possible to extend this research on the basis of the resource grammar library and on the basis of the work by Angelov & Ranta (2010).

However, the provided resource grammar is suitable also for significantly less controlled applications if the interpretation is left to the user, e.g. for tourist phrasebooks as demonstrated by Ranta et al. (2012).

Language generation facilities can be used to easily construct grammatically correct and natural sentences (or even a text) in various end-user interfaces: from simple use-cases like proper handling of named entities up to automatic verbalization of database query results or in hybrid machine translation systems (see the deliverables of the MOLTO project[10] for an example).

It should be emphasised that the limitations that are imposed by the RGL API are present only if we want to exploit the readily available multilingual parsing and generation facilities. For single-language applications it is possible to extend the resource grammar without preserving full compatibility with the shared API. For instance, the current system could be adapted for parsing texts in a weakly controlled language, e.g. legal documents. Furthermore, Angelov (2011) has demonstrated the potential of the current GF resource grammar library in statistical partial parsing of unrestricted texts.

Our future work is aimed at adaptation of the Latvian resource grammar and at creation of a reusable Latvian GF lexicon in order to enable semantic parsing of multi-domain texts. I.e., we are aiming at integration of the current approach with the frame semantics approach so that the semantic valences of a verb would be taken into account[11]. However, this would require significant modifications not only in the Latvian resource grammar, but also in the abstract syntax and to the current principles of building GF lexicons.

6. Conclusion

We have implemented a computational grammar for Latvian that works equally well for parsing and language generation. It is available as an open-source distribution in the GF release 3.3.3 and is available for download from the GF source code repository or as a part of binary packages[12]. Compiled GF application grammars are suitable for inclusion in third-party applications on various platforms.

For the developers of GF RGL modules for other languages, it may be interesting to note the discrepancies between the current resource grammar API and its implementation for Latvian. While the morphological layer is completely language-dependent, the sharing of common syntactic structures to some extent limits the resource grammar development and applicability in order to ensure the compatibility (transferability) among the languages. Our impression is that the current language-independent API is still rather biased towards peculiarities of English, and that it may be worthwhile to summarize the issues for all language implementations to identify the common limitations.

While we lack the knowledge to summarize the situation for all languages supported by the RGL, our experiments with Latvian-English-Russian parallel grammars suggest that development of accurate robust multilingual systems will eventually require including additional details in the abstract syntax layer of the RGL. Notably, we would recommend to replace the 'n-place' verb classification with more structured valence data, and to extend the common tense and mood system.

Acknowledgements

This work has been supported by the European Regional Development Fund under the project No. 2011/0009/

[9] For example, Khegai (2006) mentions similar issues.

[10] http://www.molto-project.eu/

[11] There is an ongoing work developing a valence dictionary for the most frequently used verbs in Latvian (Nešpore & Saulīte, 2012).

[12] http://www.grammaticalframework.org/

2DP/2.1.1.1.0/10/APIA/VIAA/112. The authors would like to thank Aarne Ranta for his helpful hints on the implementation details, and the anonymous reviewers for their suggestions on how to improve this paper.

References

Angelov, K. (2011). *The Mechanics of the Grammatical Framework*. PhD Thesis. Chalmers University of Technology and University of Gothenburg.

Angelov, K., Ranta, A. (2010). Implementing controlled languages in GF. In N.E. Fuchs (Ed.), *Controlled Natural Language (CNL 2009)*, Lecture Notes in Computer Science, Vol. 5972, Springer, pp. 82–101

Bārzdiņš, G., Grūzītis, N., Nešpore, G., Saulīte, B. (2007). Dependency-Based Hybrid Model of Syntactic Analysis for the Languages with a Rather Free Word Order. In *Proceedings of the 16th Nordic Conference on Computational Linguistics (NODALIDA 2007)*, Tartu, pp. 13–20

Deksne, D., Skadiņš, R. (2011). CFG Based Grammar Checker for Latvian. In *Proceedings of the 18th Nordic Conference on Computational Linguistics (NODALIDA 2011)*, Riga, pp. 275–278

Fillmore, C.J., Johnson, C.R., Petruck, M.R.L. (2003). Background to FrameNet. *International Journal of Lexicography*, 16, pp. 235–250

Fuchs N.E., Kaljurand K., Kuhn T. (2008). Attempto Controlled English for Knowledge Representation. In *Proceedings of the 4th International Reasoning Web Summer School*, Lecture Notes in Computer Science, Vol. 5224, Springer, pp. 104–124

Greitāne, I. (1997). Mašīntulkošanas sistēma LATRA (The Machine Translation System LATRA). *Proceedings of the Latvian Academy of Sciences*, Section A, 51 (3/4), pp. 1–6

Grūzītis, N., Bārzdiņš, G. (2011). Towards a More Natural Multilingual Controlled Language Interface to OWL. In *Proceedings of the 9th International Conference on Computational Semantics (IWCS 2011)*, Oxford, pp. 335–339

Grūzītis, N. (2010). Word Order Based Analysis of Given and New Information in Controlled Synthetic Languages. In *Proceedings of the Workshop on the Multilingual Semantic Web* (at WWW 2010), Raleigh, CEUR Workshop Proceedings, Vol. 571, pp. 29–34

Khegai, J. (2006). GF parallel resource grammars and Russian. In *Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING/ACL 2006)*, Sydney, pp. 475–482

Nešpore, G., Saulīte, B. (2012). Verbu valences apraksta iespējas latviešu valodā. In *Valoda: nozīme un forma. Teorija un metodoloģija latviešu valodniecībā*, Rīga: LU Akadēmiskais apgāds (to appear)

Paikens, P. (2007). Lexicon-Based Morphological Analysis of Latvian Language. In *Proceedings of the 3rd Baltic Conference on Human Language Technologies (Baltic HLT 2007)*, Kaunas, pp. 235–240

Pretkalniņa, L., Nešpore, G., Levāne-Petrova, K., Saulīte, B. (2011). A Prague Markup Language Profile for the SemTi-Kamols Grammar Model. In *Proceedings of the 18th Nordic Conference on Computational Linguistics (NODALIDA 2011)*, Riga, pp. 303–306

Ranta, A., Enache, R., Détrez, G. (2012). Controlled Language for Everyday Use: the MOLTO Phrasebook. In N.E. Fuchs, M. Rosner (Eds.), *Proceedings of the 2nd Workshop on Controlled Natural Language (CNL 2010)*, Lecture Notes in Computer Science, Vol. 7175, Springer (to appear)

Ranta, A. (2011). *Grammatical Framework: Programming with Multilingual Grammars*. Stanford: CSLI Publications

Ranta, A. (2009). The GF Resource Grammar Library. *Linguistic Issues in Language Technology*, 2 (2)

Skadiņa, I., Skadiņš, R., Deksne, D., Gornostaja, T. English/Russian-Latvian Machine Translation System. In *Proceedings of the 3rd Baltic Conference on Human Language Technologies (Baltic HLT 2007)*, Kaunas, pp. 287–295

PUBLIKĀCIJA III

FrameNet resource grammar library for GF

Controlled Natural Language, lpp. 121–137. Springer Berlin Heidelberg, 2012.

FrameNet Resource Grammar Library for GF

Normunds Gruzitis, Peteris Paikens, and Guntis Barzdins

Institute of Mathematics and Computer Science, University of Latvia
Raina blvd. 29, Riga, LV-1459, Latvia
{normunds.gruzitis,peteris.paikens,guntis.barzdins}@lumii.lv

Abstract. In this paper we present an ongoing research investigating the possibility and potential of integrating frame semantics, particularly FrameNet, in the Grammatical Framework (GF) application grammar development. An important component of GF is its Resource Grammar Library (RGL) that encapsulates the low-level linguistic knowledge about morphology and syntax of currently more than 20 languages facilitating rapid development of multilingual applications. In the ideal case, porting a GF application grammar to a new language would only require introducing the domain lexicon – translation equivalents that are interlinked via common abstract terms. While it is possible for a highly restricted CNL, developing and porting a less restricted CNL requires above average linguistic knowledge about the particular language, and above average GF experience. Specifying a lexicon is mostly straightforward in the case of nouns (incl. multi-word units), however, verbs are the most complex category (in terms of both inflectional paradigms and argument structure), and adding them to a GF application grammar is not a straightforward task. In this paper we are focusing on verbs, investigating the possibility of creating a multilingual FrameNet-based GF library. We propose an extension to the current RGL, allowing GF application developers to define clauses on the semantic level, thus leaving the language-specific syntactic mapping to this extension. We demonstrate our approach by reengineering the MOLTO Phrasebook application grammar.

Keywords: controlled natural language, frame semantics, FrameNet, multilinguality, Grammatical Framework.

1 Introduction

Controlled natural languages (CNL) can be divided into two general types according to the formalist or the naturalist approach [1]. The formalist approach supports a deterministic, bidirectional mapping of CNL to a formal language like first-order logic (FOL) or, more commonly, to description logic, namely OWL (Web Ontology Language) [2], allowing the integration with existing tools for reasoning, consistency checking and model building. Although logic-based CNL provides a seemingly informal high-level means for knowledge representation, essentially it is still a formal language that is just as expressive as the corresponding formalism, and whose interpretation is deterministic (predictable). In contrast, in the naturalist approach possible

T. Kuhn and N.E. Fuchs (Eds.): CNL 2012, LNCS 7427, pp. 121–137, 2012.

ambiguities are decreased but not excluded, thus allowing for a wider coverage of NL and more informal applications, such as semantically precise machine translation within a CNL.

In other words, there are CNLs that have an underlying logic-based formalism defining the semantics of a text (e.g. Attempto Controlled English [3]), and there are CNLs that do not have an underlying logic-based formalism (e.g. MOLTO Phrasebook [4] for multilingual translation of touristic phrases). In the first case, the semantics of CNL statements is interpreted by both a human and a formal-logic reasoning machine. In the second case, the interpreter is primarily a human and possibly a domain-specific application that uses CNL, for example, for information retrieval from a predefined domain-specific database.

Grammatical Framework (GF) [5] is a categorial grammar formalism and a toolkit for programming multilingual grammar applications. It is similar to definite clause grammars (DCG) in Prolog in that both support parsing and synthesis using the same (categorical) grammar definition. Besides the grammar formalism itself, an important part of GF is its Resource Grammar Library (RGL) [6] that encapsulates the low-level linguistic knowledge about morphology and syntax of currently more than 20 languages (the number is constantly growing). RGL facilitates rapid development and porting of application grammars in many parallel languages: all GF resource grammars implement the same syntactic interlingua (API) enabling automatic translation among languages via the abstract syntax trees. In particular, it has been shown that GF is a convenient framework for rapid and flexible implementation of multilingual CNLs – both those rooted in a formal language like the FOL-based Attempto Controlled English [7] and those rooted in a relatively informal language like standard touristic phrases [4].

In the ideal case, porting a GF application grammar to a new language or domain would only require introducing the domain lexicon – translation equivalents that are interlinked via common abstract terms. While it is possible for a highly restricted CNL, e.g. for authoring and verbalizing OWL ontologies (as implemented, for example, in [8]), developing and porting a less restricted CNL requires more linguistic knowledge about the particular language, and more experience in GF (particularly, in using RGL). Specifying a lexicon of nouns (incl. multi-word units) is mostly straightforward, however, specifying the lexicon of verbs is typically the most complex task in terms of both inflectional paradigms and argument structure, and may require specifying the whole clause (as in Phrasebook). Thus adding verbs to a GF application grammar's lexicon in a foreign language (or for a novice or less resourced GF developer even in his mother tongue) might not be a straightforward task and might present a stumbling block for potential GF multilingual application developers. Therefore in this paper we are focusing on verbs, investigating the possibility of creating a multilingual FrameNet-based GF resource grammar library.

The rest of the paper is organized as follows. In Section 2 we briefly re-capture the relevant architectural principles of FrameNet. In Section 3 we similarly re-capture some relevant GF application grammar development principles, demonstrating the current approach with a detailed example. Section 4 modifies the previous example, describing our solution for integrating FrameNet into GF application grammars. Finally we conclude with a brief discussion on the potential and benefits of the proposed FrameNet library for seamless multilingual CNL development.

2 FrameNet

FrameNet [9] is a lexicographic database that describes word meanings based on the principles of frame semantics. The central idea of frame semantics is that word meanings must be described in relation to semantic frames [10]. Therefore, the *frame* and the *lexical unit* are the key components of FrameNet. A lexical unit in FrameNet terms is the combination of a lemma with a specific meaning – each separate meaning of a word represents a new lexical unit. In FrameNet, each lexical unit is related to a semantic frame that it is said to *evoke* a frame (see Figure 1 and Figure 2).

The semantic frame describes a certain situation and the participants of that situation that are likely to be mentioned in the sentences where the evoking lexical unit (referred to as frame *target*) appears. The semantic roles played by these participating entities are called *frame elements* (FE). FrameNet makes a differentiation between *core* frame elements and *peripheral* frame elements. In general, frame elements that are necessarily realized are core elements. Peripheral elements represent more general information such as time, manner, place, and purpose and are less specific to the frame. Nevertheless all FrameNet frame elements are local to individual frames. This avoids the commitment to a small set of universal roles, whose specification has turned out to be controversial in the past [11]. In order to account for actual similarities between frame elements in different frames FrameNet includes also a rich set of frame to frame and FE to FE relations.

Residence		*This frame has to do with people (the Residents) residing in Locations, sometimes with a Co-resident.*
Core FEs	**Co_resident**	*A person or group of people that the Resident is staying with or among.*
	Location	*The place in which somebody resides.*
	Resident	*The individual(s) that reside at the Location.*
Lexical units		camp.v, dwell.v, inhabit.v, live.v, lodge.v, occupy.v, reside.v, room.v, squat.v, stay.v

Fig. 1. A sample FrameNet frame (only core frame elements shown)

The frame descriptions are coarse-grained and generalize over lexical variation. Therefore lexeme-specific information is contained within lexical unit entries that are more fine-grained and contain a definition of the lexical unit, the syntactic realizations of each frame element and the valence patterns. A sense of a lemma (word meaning) can evoke a frame, and thus form a lexical unit for this frame, if this sense is syntactically able to realize the core frame elements that instantiate a conceptually necessary component of a frame [12].

In Figure 2, a simplified (summarized) lexical entry of 'to live' (`Residence`) is given: information on non-core FEs is excluded (the rest is summed up); for each valence model only the most frequent realization pattern is given; valence models that contain multiple FEs of the same type are excluded; valence models that have ap-

peared in the corpus only once are excluded; prepositional phrase patterns (PP) are not distinguished by particular prepositions.

FE	Total	Pattern
Co_resident	14	PP.Dep (86%)
Location	131	PP.Dep (81%)
Resident	143	NP.Ext (90%)

(a)

Total	Patterns		
98		**Location**	**Resident**
71%		PP.Dep	NP.Ext
7	**Co_resident**		**Resident**
86%	PP.Dep		NP.Ext
7	**Co_resident**	**Location**	**Resident**
86%	PP.Dep	PP.Dep	NP.Ext

(b)

Fig. 2. A simplified lexical entry Residence.live. (a) Core FEs and their most frequent syntactic patterns in the FrameNet corpus. (b) Most frequent valence models of core FEs.

Our central point of interest in this paper is the multilingual dimension of FrameNet. A number of projects have investigated the use of English FrameNet frames for other languages, such as German (SALSA project [13]), Spanish [14], Japanese [15], and lately also for Thai, Chinese, Italian, French, Bulgarian, Hebrew [16]. A fundamental assumption of these projects is that English FrameNet frames can be largely re-used for the semantic analysis of other languages. This assumption rests on the nature of frames as coarse-grained semantic classes which refer to prototypical situations – to the extent that these situations agree across languages, frames should be applicable cross-linguistically. Also Boas [17] suggests the use of semantic frames as interlingual representation for multilingual lexicons.

While FrameNet multilinguality is clearly a very attractive assumption, its empirical validation comes primarily from the German SALSA project, which has found that the vast majority of English FrameNet frames can be directly applied to the analysis of German – a language that is typologically close to English. Meanwhile, some frames have turned out not to be fully interlingual and three main cross-lingual divergence types were found:

1. Ontological distinctions between similar frame elements.
2. Missing frame elements.
3. Differences in lexical realization patterns (e.g. German 'fahren' does not distinguish between Operate_vehicle and Ride_vehicle frames).

Nevertheless this empirical evidence shows that most of FrameNet frames indeed are language independent and therefore provide an opportunity for use as a multilingual coarse-grained lexicon in GF, as described in Section 4. Although FrameNet addresses all parts-of-speech, its strength and focus is on verbs for which the best coverage is provided. This is largely because while noun-phrase multi-word units are extensively "invented" to denote nominal concepts (especially in technical domains), phrasal verbs are more fixed, commonly reused (across domains) and are often captured in dictionaries of standard language. Since the advantage of valence structures is

more obvious for verbs, in this paper we consider only FrameNet frames with verbal frame evoking lexical entries.

3 GF Application Grammar Development: The Current Approach

GF facilitates reusability by splitting the grammar development in two levels:

1. A general-purpose *resource grammar* covers a wide range of morphological paradigms and syntactic structures and as such is highly ambiguous. GF provides a Resource Grammar Library (RGL) [6] implementing a common API for more than 20 languages.
2. Domain specific *application grammars* reuse the RGL, defining semantic structures and the subset of natural language (syntax and lexicon) that is used within a particular CNL. Application grammars reduce or even eliminate ambiguities.

Development of a resource grammar requires in-depth GF knowledge and in-depth linguistic knowledge about the particular language. Once a resource grammar is provided, application grammars are built on top of it significantly reducing the linguistic knowledge prerequisites (a non-linguist native or fluent speaker should be sufficient), as well as he or she can be less experienced with GF.

GF differentiates not only between general-purpose resource grammars and domain-specific application grammars, but also between *abstract syntax* and *concrete syntax*. The abstract syntax captures the semantically relevant structure of a CNL, defining grammatical categories and functions for building abstract syntax trees [5]. The concrete syntax defines the linearization of the CNL abstract syntax trees at the surface level for each language. Translation among languages (concrete syntaxes) is provided via abstract syntax[1].

We will describe the current approach to the RGL-based GF application grammar development using the MOLTO Phrasebook application [4] for multilingual translation of touristic phrases as an example. Phrasebook is a CNL implemented in 15 languages and is aimed to be usable by anyone without prior training. It has 42 categories and 290 functions. The number of phrases it can generate is infinite, but on a reasonable level of tree depth 3, Phrasebook has nearly 500,000 abstract syntax trees [4]. We will consider only a small subset of the Phrasebook grammar – categories (`cat`) and functions or constructors (`fun`) that are used to build the abstract syntax trees for the following sample sentences, and to generate these sentences from the corresponding abstract trees[2] as given in Figure 3.

In the next section we will modify the English implementation of the Phrasebook grammar by means of the proposed FrameNet-based resource grammar, acquiring a

[1] Note that in GF there is no concept of a language pair or a translation direction. Also there is no common semantic interlingua. Instead there are many application specific (i.e., CNL and domain specific) interlinguas.

[2] The provided abstract syntax trees are slightly simplified regarding the pronouns – their gender, number and politeness features – to avoid multiple variants.

simpler English Phrasebook (PhrasebookEng) implementation as the result, while preserving the same functionality. However, we will not make any changes neither in the Phrasebook functor (the common incomplete concrete syntax), nor in the abstract syntax, i.e., we will not impose any special requirements on application grammar design.

English sentences	Phrasebook abstract syntax
I like this pizza.	`PSentence (SProp (PropAction (ALike I (This Pizza))))`
I live in Belgium.	`PSentence (SProp (PropAction (ALive I Belgium)))`
I love you.	`PSentence (SProp (PropAction (ALove I You)))`
I want a good pizza.	`PSentence (SProp (PropAction (AWant I (OneObj (ObjIndef (SuchKind (PropQuality Good) Pizza))))))`
I want to go to a museum.	`PSentence (SProp (PropAction (AWantGo I (APlace Museum))))`

Fig. 3. Sample Phrasebook sentences along with their abstract syntax trees

The abstract syntax of Phrasebook that represents the syntactic and semantic model of the above phrases is given in Figure 4.

```
cat
  Action ;     -- proposition about a Person, e.g. "I love you"
  Phrase ;     -- complete phrase, e.g. "I love you."
  Country ;    -- e.g. "Belgium"
  Item ;       -- single entity, e.g. "this pizza"
  Kind ;       -- kind of an item, e.g. "pizza"
  Object ;     -- e.g. "a good pizza"
  Person ;     -- agent wanting or doing something, e.g. "I"
  Place ;      -- location, e.g. "a museum"
  PlaceKind ;  -- kind of location, e.g. "museum"
  Property ;   -- basic property of an item, e.g. "good"

fun
  Belgium : Country ;
  Good    : Property ;
  Museum  : PlaceKind ;
  Pizza   : Kind ;

  ALike   : Person -> Item    -> Action ;  -- Action(Person, Item)
  ALive   : Person -> Country -> Action ;
  ALove   : Person -> Person  -> Action ;
  AWant   : Person -> Object  -> Action ;
  AWantGo : Person -> Place   -> Action ;
```

Fig. 4. A fragment of Phrasebook abstract syntax (semantic model)

A fragment of the incomplete concrete syntax (aka functor) of Phrasebook is given in Figure 5. This is a technical intermediate layer between the abstract syntax and its implementation in concrete syntaxes. It defines language-independent syntactic categories and structures (e.g. `PSentence`, `SProp`, `PropAction`) that are common to all (or most) languages. Thus the concrete syntax of a particular language has to specify only language-dependent structures and the lexicon (e.g. `AWant`, `Good`, `Pizza`).

Note that the functor defines the mapping between the application-specific abstract syntax categories and the categories of the Resource Grammar Library. For instance, `Country`, `Item` and `Object` syntactically are realized as noun phrases (category `NP` in RGL). In Figure 5, there are also three Phrasebook categories that are not directly mapped to RGL categories (`Person`, `Place` and `PlaceKind`). Instead, they are defined as application-specific categories `NPPerson`, `NPPlace` and `CNPlace` that are specified as record types whose fields (e.g. `name`, `at`, `to`) are of RGL types.

```
lincat  -- category linearization types
  Phrase = Text ;
  Action = Cl ;
  Country, Item, Object = NP ;
  Person = NPPerson ;
  Place = NPPlace ;
  Kind = CN ;
  PlaceKind = CNPlace ;
  Property = A ;

oper  -- operations - functions in concrete syntax
  NPPerson : Type = {name : NP ; isPron : Bool ; poss : Quant} ;
  NPPlace  : Type = {name : NP ; pos : Adv ; dir : Adv} ;
  CNPlace  : Type = {name : CN ; pos : Prep ; dir : Prep} ;

  mkNPPerson : Pron -> NPPerson = \pron ->
    {name = mkNP pron ; isPron = True ; poss = mkQuant pron} ;

  mkCNPlace : CN -> Prep -> Prep -> CNPlace = \cn,prep1,prep2 ->
    {name = cn ; pos = prep1 ; dir = prep2} ;

  mkNPPlace : Det -> CNPlace -> NPPlace = \det,place ->
    let name : NP = mkNP det place.name in {
      name = name ;
      pos = mkAdv place.pos name ;  -- place - position
      dir = mkAdv place.dir name    -- place - direction
    } ;
```

Fig. 5. A fragment of Phrasebook incomplete concrete syntax (functor): common structures. To make the code more intelligible to readers unfamiliar with GF, it has been slightly modified.

The predication patterns (`Action`) with verbs at the centre are perhaps the most complex functions in Phrasebook (from the implementation point of view). Note that these actions are of type `Cl` (clause): this will be the gluing point for the integration of the FrameNet-based resource library (see Section 4). The linguistic (English) realization of the semantic model is specified by the concrete syntax (given in Figure 6),

which tells how abstract syntax trees are linearized (`lin`) into English strings. The same rules are also used for parsing.

```
lin  -- function linearization rules
  Belgium = mkNP (mkPN "Belgium") ;
  Good = LexiconEng.good_A ;
  Museum = mkPlaceKind "museum" "at" ;
  Pizza = mkCN (mkN "pizza") ;

  ALike pers item = mkCl pers.name (mkV2 (mkV "like")) item ;
  ALive pers country = mkCl pers.name (mkVP (mkVP (mkV "live"))
    (mkAdv SyntaxEng.in_Prep country)) ;
  ALove pers1 pers2 =
    mkCl pers1.name (mkV2 (mkV "love")) pers2.name ;
  AWant pers obj = mkCl pers.name (mkV2 (mkV "want")) obj ;
  AWantGo pers place = mkCl pers.name SyntaxEng.want_VV
    (mkVP (mkVP IrregEng.go_V) place.dir) ;

oper
  mkPlaceKind : Str -> Str -> CNPlace = \name,prep_pos ->
    mkCNPlace (mkCN (mkN name)) (mkPrep prep_pos) SyntaxEng.to_Prep ;
```

Fig. 6. A fragment of Phrasebook concrete syntax for English

Verbs, in general, are at the centre of a sentence, both syntactically and semantically. They have the most complex inflectional paradigms (at least in inflective languages). The syntactic and semantic valence of a verb is defined via its argument and modifier structure. This inevitably requires solid linguistic knowledge.

RGL differentiates among V (intransitive), V2 (transitive) and V3 (ditransitive) verbs, as well as some more specific types of verbs with syntactically fixed argument structure. The syntactic valence patterns for the predefined verb types are fixed when defining a verb in the application lexicon; these patterns do not depend on the argument (i.e. the case or preposition of the argument does not depend on a particular NP). Other valences are specified while constructing a verb phrase – as adverbial modifiers (`Adv`); their syntactic patterns are specified by the application developer for each target language, depending on the semantic role of the argument and syntactic properties of the language.

If compared to nouns, there are much less verbs and they are more ambiguous (see WordNet statistics[3], for example), thus verbs are also more reusable linguistic units than nouns. This suggests that a reusable lexicon of verbs would be helpful for GF application developers. However, this requires not only a dictionary, but also additional information about the basic syntactic valences for the direct and indirect objects. Even more helpful would be a multilingual resource grammar of verb valences.

There is a small, limited multilingual lexicon provided by the RGL, but it does not provide systematic means for scaling and expanding beyond the V-, V2- and V3-like

[3] `http://wordnet.princeton.edu/wordnet/man/wnstats.7WN.html`

valences. The basic lexicon also does not support polysemous verbs – valences often are different for various meanings of the same verb, and vice versa.

An impression of the syntactic coverage of the GF Resource Grammar Library can be obtained from its API documentation[4]. Figure 7 illustrates some of the constructors for clauses, verb phrases, noun phrases, common nouns, and adverbial modifiers that are referred in Figure 5 and Figure 6. For instance, Figure 7 illustrates that a clause can be built from a subject noun phrase with a verb and appropriate arguments. In general, a clause can be built from a subject noun phrase and a verb phrase.

Function	Type	Example
mkCl	NP -> VP -> Cl	*she always sleeps*
mkCl	NP -> V2 -> NP -> Cl	*she loves him*
mkCl	NP -> VV -> VP -> Cl	*she wants to sleep*
mkVP	VP -> Adv -> VP	*to sleep here*
mkNP	Det -> CN -> NP	*the old man*
mkNP	PN -> NP	*Paris*
mkNP	Pron -> NP	*we*
mkCN	N -> CN	*house*
mkAdv	Prep -> NP -> Adv	*in the house*

Fig. 7. A fragment of the Resource Grammar API documentation

In order to use the proposed FrameNet library (in addition to the Resource Grammar API), the application grammar developer will have to consult the FrameNet API as presented in the next section.

4 Our FrameNet-Based Approach

The current split of functionality between the application and the common libraries expects applications to define the domain specific knowledge in all languages by using specific verbs and defining their syntactic valences in each target language.

Our proposal is to raise the abstraction level for the common GF clause construction API from the current syntactic definition to a more semantic one. As we discussed in Section 2, the research on frame semantics suggests that an exhaustive cross-domain linguistic model of semantic frames and roles is possible, and it has been implemented for multiple languages. We believe that it is possible and reasonable to facilitate the development of multilingual application grammars in GF by referencing a common API of semantic frames that provide language-specific linearization for whole clauses or verb phrases (VP), and can optionally provide a default choice of a lexical unit that evokes the frame and default syntactic valence patterns.

In particular, we envision a resource grammar library that is built on top of the current RGL offering each of the FrameNet's semantic frames as a function that builds a

[4] http://www.grammaticalframework.org/lib/doc/synopsis.html

clause from given parameters: fillers of the core elements of that frame, and an optional list of elements filling the peripheral roles. The FrameNet API would be implemented semi-automatically by generating GF code from FrameNet data providing a set of overloaded functions for each frame – mapping the frame (its elements) to the default or specific syntactic realization (linearization). Our observation is that, in the current approach to GF application development, a miniature ad-hoc 'framenet' is actually implemented for each application. Moreover, it is often 'reused' in a copy-paste-edit manner from previous applications or from concrete syntaxes of other languages that implement the same application. We would like to promote systematic means for reusing this language-specific knowledge via common language-independent frames.

The following simplified assumptions underlie the default behaviour of our approach (the default behaviour can be overridden for specific syntactic patterns and lexical units – see systematic "exceptions" illustrated below):

1. For each frame element there is a typical syntactic pattern that is used in most cases – independently of the verb that evokes the frame. I.e., both semantic and syntactic valences can be defined at the frame level.
    1.1. There is a common syntactic realization of a frame (a clause or a verb phrase) that is reused by most verbs that evoke the frame.
2. It is possible to specify a default lexical unit (the most general and/or the most frequently used verb) that evokes the frame, so that it can be used in the linearization (translation) of the frame, if no specific verb is provided.
3. In the CNL settings, it is often sufficient that only core semantic valences (core frame elements according to FrameNet) are available.

These assumptions, of course, do not hold in general, but they help us to keep the presentation of our approach simpler. Even then we cannot fully isolate the application developer from providing some language-specific features. For example, the Phrasebook application in English (and similarly in Russian) needs to distinguish between locations that are "at place" or "in place" – the preposition does not depend on the frame and not even on the specific verb, but on the particular noun (the filler of a frame element). In contrast to the highly analytical English, in many languages it might be necessary to customize the realization of the whole clause, depending on the verb. For example, in Latvian (and similarly in Russian, Italian and German) there are verbs (systematic "exceptions") that instead of the subject in the nominative case and the object in the accusative case require the subject in the dative case and the object in the nominative case[5] (see Figure 8). In the actual implementation of the FrameNet RGL, such agreement variations have to be handled by alternative verb-specific clauses implemented in the frame functions.

[5] Here we use the term 'case' in a broad sense: in Italian, for example, there are no cases for nouns; cases are expressed by prepositions, pronouns and implicitly by word order.

	LOVE	LIKE
English	I$_{[NOM]}$ love pizza$_{[ACC]}$	I$_{[NOM]}$ like pizza$_{[ACC]}$
German	Ich$_{[NOM]}$ liebe Pizza$_{[ACC]}$	Ich$_{[NOM]}$ mag Pizza$_{[ACC]}$ Mir$_{[DAT]}$ gefällt Pizza$_{[NOM]}$
Italian	Io$_{[NOM]}$ amo la pizza$_{[ACC]}$	A me$_{[DAT]}$ piace la pizza$_{[NOM]}$
Latvian	Es$_{[NOM]}$ mīlu picu$_{[ACC]}$	Man$_{[DAT]}$ patīk pica$_{[NOM]}$
Russian	Я$_{[NOM]}$ люблю пиццу$_{[ACC]}$	Мне$_{[DAT]}$ нравится пицца$_{[NOM]}$

Fig. 8. Verb-specific realization of the frame elements `Experiencer` and `Content` in different languages. All these verbs belong to the `Experiencer_focus` frame.

As in the case of the syntactic RGL, the proposed semantic resource grammars of the FrameNet library will also be ambiguous as such: the same verb can evoke different frames, and the same frame might be evoked by contradicting verbs (e.g. both 'to love' and 'to hate' evoke the same `Experiencer_focus` frame). However, the intuition is that the developer of a domain-specific CNL will reduce or eliminate the semantic ambiguity by avoiding ambiguous mappings between lexical units and frames, by specifying concrete verb lexemes instead of relying on the default ones etc. – analogically as it is currently done at the syntactic level.

To illustrate the use of the FrameNet API, we provide a sample re-implementation of some clause-building functions from the MOLTO Phrasebook application (see Figure 9) in contrast to the current `PhrasebookEng` implementation of the same functions as shown earlier in Figure 6.

Before:

```
ALike pers item = mkCl pers.name (mkV2 (mkV "like")) item ;
ALive pers country = mkCl pers.name (mkVP (mkVP (mkV "live"))
  (mkAdv SyntaxEng.in_Prep country)) ;
ALove pers1 pers2 =
  mkCl pers1.name (mkV2 (mkV "love")) pers2.name ;
AWant pers obj = mkCl pers.name (mkV2 (mkV "want")) obj ;
AWantGo pers place = mkCl pers.name SyntaxEng.want_VV
  (mkVP (mkVP IrregEng.go_V) place.dir) ;
```

After:

```
ALike pers item =
  Experiencer_focus (mkV "like") pers.name item NIL NIL ;
ALive pers country = Residence pers.name NIL country ;
ALove pers1 pers2 =
  Experiencer_focus (mkV "love") pers1.name pers2.name NIL NIL ;
AWant pers obj = Possession (mkV "want") pers.name obj ;
AWantGo pers place =
  Desiring pers.name (Motion_VP IrregEng.go_V NIL place.name) ;
```

Fig. 9. Changes to the `PhrasebookEng` syntax using the proposed FrameNet API

As seen in Figure 9, the application grammar developer still has to provide the domain-specific knowledge that the application requires, and some simple constructors of the GF RGL are still used, but the code is more intelligible and 'flat' – it is not specified how the parameters (frame elements) are glued together to build up verb phrases and clauses[6]. The proposed API refers to the semantic roles only: if the user specifies, for example, the resident of the `Residence` frame (`ALive` action in Phrasebook), the FrameNet library maps it to the relevant syntactic role (subject in this case). Thus the verb and clause building part of application grammars such as Phrasebook is in essence reduced to mapping domain-specific concepts to the appropriate general FrameNet frames, and to specifying the omitted core frame elements, if any (`NIL`[7]).

In multilingual applications, there is a general issue of selecting lexical units – translation equivalents. For example, for the `Residence` frame, there are many possible verbs that describe the same situation with various semantic nuances (e.g. 'to camp', 'to dwell', 'to live', 'to stay'; see Figure 1). If these differences are relevant to the application domain, then a particular lexical unit can be explicitly specified. If the differences are not considered important for a particular use-case or concept, the preferred lexical unit for the chosen frame can be omitted, resulting in a robust system that would use a default verb (e.g. 'to live') when generating a text, and that would allow all frame-relevant verbs in parsing.

An advantage of this approach is the ability to build robust multilingual CNL applications without expertise in all covered languages. The benefit of using GF is that it would be possible to port such applications to other languages without going into details of their grammars – as they are already implemented in the common RGL. Furthermore, it is possible to omit the details about how the semantic roles are mapped to syntactic elements, as the same semantic element may be expressed by different syntactic means when translating the same clause to another language.

The API of the proposed FrameNet RGL is illustrated in Figure 10 (similarly as the API of GF RGL in Figure 7). The function names match the FrameNet frame names, thus the API can be automatically documented by FrameNet data providing definitions and examples for each frame (function) and each frame element (argument of the function).

Although currently we have handcrafted the code of the sample FrameNet library, we have done it systematically using the actual FrameNet data that is well structured and includes statistics from a FrameNet-annotated corpus. This has given confidence that FrameNet data can be used to automatically generate both the abstract syntax of the FrameNet API and its implementation for English and other languages using the current GF RGL syntactic categories and constructors, and properly addressing verb-specific valence patterns.

---

[6] Note that the implementation of the `AWantGo` function is not 'flat' – there are nested frames. I.e., it might be necessary to specify the semantic tree structure, but not the syntactic structure.

[7] We have not specified the implementation of `NIL` arguments yet, but this is only a technical mater.

We have performed some initial experiments on automatic GF code generation from FrameNet data, but the development of a more elaborated convertor is pending. Nevertheless, there are only about 1000 frames in FrameNet, therefore the generated code can also be manually debugged and improved afterwards.

Function/Frame	Type	Mapping to FEs
`Residence`	`V -> NP -> PP -> Adv -> Cl`	Resident → Co_resident → Location
	`V -> NP -> NIL -> Adv -> Cl`	
	`NP -> NIL -> NP -> Cl`	
`Possession`	`V -> NP -> NP -> Cl`	Owner → Possession
	`NP -> NP -> Cl`	
`Desiring`	`VV -> NP -> VP -> Cl`	Experiencer → Event
	`NP -> VP -> Cl`	
`Motion`	`V -> NP -> NP -> NP -> Cl`	Theme → Source → Goal
	`NP -> NP -> NP -> Cl`	
`Motion_VP`	`V -> NP -> NP -> VP`	Source → Goal
	`V -> NIL -> NP -> VP`	
	`NP -> NP -> VP`	
	`Adv -> Adv -> VP`	
`Experiencer_focus`	`V -> NP -> NP -> VP -> NP -> Cl`	Experiencer → Content → Event → Topic
	`V -> NP -> NP -> NIL -> NIL -> Cl`	
	`NP -> NP -> NIL -> NIL -> Cl`	

Fig. 10. A simplified fragment of the proposed FrameNet API. The `Desiring` frame has actually four core elements, and `Motion` – seven. Also all the possible combinations of `NP`, `PP`, `Adv` and `NIL` argument types are not included. Note that the `Motion_VP` is a special case of `Motion` – generated for use as a nested frame (as the `VP` object of a `VV` verb).

The manually generated code for several FrameNet frames, as shown in Figure 11, implements the features in a very similar manner as the Phrasebook application shown earlier in Figure 6 – which is to be expected, as it needs to realize similar syntactic structures with the same GF resources. However, a major difference is that this code would be reusable for multiple applications, and it could cover larger domains in a scalable way.

There are still some technical issues that need to be addressed, such as a more convenient way for specifying omitted core frame elements, but we believe that these are minor challenges. A particular concern is common peripheral semantic roles such as Time, Place and Manner that are encountered in nearly all frames (if they are not among the core roles for that frame). Again, the current Resource Grammar API deals with them on a syntactic level – providing means to attach various adverbial modifiers. We propose adding them as a (possibly empty) list of peripheral parameters, allowing the language-specific API implementation to handle the word order changes as needed.

```
-- Residence : NP -> NIL -> NP -> Cl
Residence resident NIL location = Residence (mkV "live") resident NIL
  (mkAdv SyntaxEng.in_Prep location) ;

-- Residence : V -> NP -> NIL -> Adv -> Cl
Residence verb resident NIL location =
  mkCl resident (mkVP (mkV "live") location) ;

-- Residence : V -> NP -> PP -> Adv -> Cl
Residence verb resident co_resident location = mkCl resident
  (mkVP (mkVP (mkV2 verb co_resident.prep) co_resident.np) location) ;

-- Possession : V -> NP -> NP -> Cl
Possession verb owner possession =
  mkCl owner (mkVP (mkV2 verb) possession) ;

-- Desiring : NP -> VP -> Cl
Desiring experiencer event =
  Desiring SyntaxEng.want_VV experiencer event ;

-- Desiring : VV -> NP -> VP -> Cl
Desiring verb experiencer event = mkCl experiencer verb event ;

-- Motion : V -> NP -> NP -> NP -> Cl
Motion verb theme source goal = mkCl theme (Motion_VP verb source goal) ;

-- Motion_VP : NP -> NP -> VP
Motion_VP source goal = Motion_VP (mkV "move") source goal ;

-- Motion_VP : V -> NP -> NP -> VP
Motion_VP verb source goal = mkVP (
  (mkVP (mkVP verb) (mkAdv SyntaxEng.from_Prep source))
  (mkAdv SyntaxEng.to_Prep goal)) ;

-- Motion_VP : V -> NIL -> NP -> VP
Motion_VP verb NIL goal =
  mkVP (mkVP verb) (mkAdv SyntaxEng.to_Prep goal) ;

-- Experiencer_focus : V -> NP -> NP -> NIL -> NIL -> Cl
Experiencer_focus verb experiencer content NIL NIL =
  mkCl experiencer (mkV2 verb) content ;
```

Fig. 11. English implementation of the proposed FrameNet API (a simplified fragment). `PP` extends the RGL set of categories; its linearization type is `{prep : Prep ; np : NP}`.

5 Discussion and Future Work

Currently the GF toolset provides a reusable syntactic framework for the development of multilingual domain-specific CNLs. When one acquires a solid understanding of

the RGL structure and design principles, and gets used to the RGL-based application grammar design patterns, it is a rather rapid development to provide a concrete syntax for a language he or she knows well[8]. However, the process still might not be straightforward, especially when porting a third-party application, as it might not be enough to look at the code of e.g. English implementation to (immediately) understand the intended meaning of a specific abstract word or clause to provide an appropriate translation. In addition, different application grammars that cover related domains will more or less overlap, so that the same structures are re-implemented for each application.

In the current approach, GF application grammar developers essentially provide a miniature domain-specific framenet for each application. We make a case for basing application development on a common, reusable semantic framework, and argue that it is reasonably possible to develop such a framework by leveraging the existing FrameNet data. Working on the semantic level requires specific knowledge and training as well[9], but the resulting systems are more generic and easier to reuse across languages and across applications and domains.

The proposed approach is aimed to lower the entrance barrier of the GF application grammar development by moving it from the language-specific syntactic level towards the language-independent semantic level. The long-term goal is to facilitate the development of multilingual applications by providing robust means for automatic alignment of translation equivalents (particularly verbs), and by reducing syntactic and lexical ambiguities that appear in the parsing and generation of less restricted CNLs. GF has been chosen as an advanced and well-resourced framework for this purpose, but the proposed general principle could be applied also to other grammar formalisms.

The main limitations of the proposed approach to some extent are related to the limitations of FrameNet, particularly its coverage (in terms of lexical units). Furthermore, the coverage might differ among languages. The list of the lexical units for each frame could be extended via WordNet, as it has been shown by Johansson and Nugues [18], however then we would have to fall back to the frame-specific (vs. verb-specific) valence patterns. Another limitation is that even the most frequently used verb-specific valence patterns might not be appropriate in specific cases.

Although we have tested our proposal only on the English FrameNet data and English Phrasebook grammar, considering other languages only theoretically, we believe that in overall this would ease the multilingual GF application development, and that the limitations can be overcome by using the RGL syntactic structures directly where necessary. By relying on the default syntactic realization and the default lexical units, one can quickly obtain the first working version of a multilingual aplication for further testing and tuning. In fact, the default behaviour can be specified already in the

[8] The experience of the MOLTO team shows that adding a new language to Phrasebook takes 1.5 days on average.

[9] GF developers would have to explore and consult the documentation of FrameNet data (`https://framenet.icsi.berkeley.edu/fndrupal/framenet_data`) while designing or porting an application grammar.

functor that defines the common language-independent structures of an application grammar.

Apart from the development of the GF code generation facility from FrameNet data and apart from a wider evaluation taking into account both more languages and more applications, future work is also to investigate the possibilities for semi-automatic multilinguality by choosing the most appropriate lexical units automatically, and by aligning lexical units (translation equivalents) among different languages.

An additional direction for future work is the application of this semantic layer to relatively unrestricted natural language – in line with the naturalist approach [1]. Angelov [19] has demonstrated the potential of the current GF Resource Grammar Library in statistical parsing of unrestricted texts (using weights extracted from a treebank). FrameNet data would provide additional means in disambiguation and would provide mapping of parse results to semantic categories. Also Barzdins [20] has addressed bridging the gap between the CNL and full natural language through use of FrameNet on the discourse level. Integration with frame semantics thus provides additional means towards semantic parsing of less controlled text.

Acknowledgments. This work has been supported by the European Regional Development Fund under the project No. 2011/0009/2DP/2.1.1.1.0/10/APIA/VIAA/112. The authors would like to thank the reviewers for the detailed comments and constructive criticism.

References

1. Clark, P., Murray, W.R., Harrison, P., Thompson, J.: Naturalness vs. Predictability: A Key Debate in Controlled Languages. In: Fuchs, N.E. (ed.) CNL 2009. LNCS, vol. 5972, pp. 65–81. Springer, Heidelberg (2010)
2. Schwitter, R., Kaljurand, K., Cregan, A., Dolbear, C., Hart, G.: A Comparison of three Controlled Natural Languages for OWL 1.1. In: Proceedings of the 4th International Workshop on OWL Experiences and Directions (OWLED), CEUR, vol. 496 (2008)
3. Fuchs, N.E., Kaljurand, K., Kuhn, T.: Attempto Controlled English for Knowledge Representation. In: Baroglio, C., Bonatti, P.A., Małuszyński, J., Marchiori, M., Polleres, A., Schaffert, S. (eds.) Reasoning Web 2008. LNCS, vol. 5224, pp. 104–124. Springer, Heidelberg (2008)
4. Ranta, A., Enache, R., Détrez, G.: Controlled Language for Everyday Use: The MOLTO Phrasebook. In: Rosner, M., Fuchs, N.E. (eds.) CNL 2010. LNCS (LNAI), vol. 7175, pp. 115–136. Springer, Heidelberg (2012)
5. Ranta, A.: Grammatical Framework: Programming with Multilingual Grammars. CSLI Publications, Stanford (2011)
6. Ranta, A.: The GF Resource Grammar Library. Linguistic Issues in Language Technology 2(2) (2009)
7. Angelov, K., Ranta, A.: Implementing Controlled Languages in GF. In: Fuchs, N.E. (ed.) CNL 2009. LNCS, vol. 5972, pp. 82–101. Springer, Heidelberg (2010)

8. Gruzitis, N., Barzdins, G.: Towards a More Natural Multilingual Controlled Language Interface to OWL. In: Proceedings of the 9th International Conference on Computational Semantics (IWCS), Oxford, pp. 335–339 (2011)
9. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The Berkeley FrameNet project. In: Proceedings of the COLING-ACL, Montreal, pp. 86–90 (1998)
10. Fillmore, C.J., Johnson, C.R., Petruck, M.R.L.: Background to FrameNet. International Journal of Lexicography 16(3), 235–250 (2003)
11. Burchardt, A., Erk, K., Frank, A., Kowalski, A., Pado, S., Pinkal, M.: Using FrameNet for the semantic analysis of German: Annotation, representation, and automation. In: Boas, H.C. (ed.) Multilingual FrameNets in Computational Lexicography: Methods and Applications, pp. 209–244. Mouton de Gruyter, Berlin (2009)
12. Ruppenhofer, J., Ellsworth, M., Petruck, M.R.L., Johnson, C.R., Scheffczyk, J.: FrameNet II: Extended Theory and Practice (2010)
13. Burchardt, A., Erk, K., Frank, A., Kowalski, A., Pado, S., Pinkal, M.: The SALSA corpus: a German corpus resource for lexical semantics. In: Proceedings of the 5th International Conference on Language Resources and Evaluation (2006)
14. Subirats, C.: Spanish FrameNet: A frame-semantic analysis of the Spanish lexicon. In: Boas, H.C. (ed.) Multilingual FrameNets in Computational Lexicography: Methods and Applications. Mouton de Gruyter, Berlin (2009)
15. Kyoko, O., Fujii, S., Ishizaki, S., Ohori, T., Saito, H., Suzuki, R.: The Japanese FrameNet Project: an Introduction. In: Proceedings of the Workshop on Building Lexical Resources from Semantically Annotated Corpora (at LREC), Lisbon, pp. 9–11 (2004)
16. Leenoi, D., Jumpathong, S., Porkaew, P., Supnithi, T.: Thai FrameNet Construction and Tools. International Journal on Asian Language Processing 21(2), 71–82 (2011)
17. Boas, H.C.: Semantic frames as interlingual representations for multilingual lexical databases. International Journal of Lexicography 18(4), 445–478 (2005)
18. Johansson, R., Nugues, P.: Using WordNet to Extend FrameNet Coverage. In: Proceedings of the Workshop on Building Frame Semantics Resources for Scandinavian and Baltic Languages (at NODALIDA), Tartu, pp. 27–30 (2007)
19. Angelov, K.: The Mechanics of the Grammatical Framework. PhD Thesis, Chalmers University of Technology and University of Gothenburg (2011)
20. Barzdins, G.: When FrameNet meets a Controlled Natural Language. In: Proceedings of the 18th Nordic Conference on Computational Linguistics (NODALIDA), Riga, pp. 2–5 (2011)

PUBLIKĀCIJA IV

Towards named entity annotation of Latvian National Library corpus

Baltic HLT, lpp. 169–175, 2012.

Towards named entity annotation of Latvian National Library corpus

Peteris PAIKENS[1], Ilze AUZINA, Ginta GARKAJE and Madara PAEGLE
University of Latvia, Institute of Mathematics and Computer Science

Abstract. The paper describes a work in progress of building a catalogue of named entities – people, places and organizations – based on a recently digitized large (4.5 billion tokens) Latvian corpus. The authors propose an annotation standard for markup of named entities within Latvian corpus, according to which a representative set of documents (150 000 words) are manually annotated. This corpus is used for training and evaluation of an automated named entity recognition system based on Stanford CRF classifier, achieving an F-score of up to 81%. The named entities indexed within the Latvian National Library corpus and the annnotated documents are publicly available for linguistic and historical research online.

Keywords. Named entity recognition, NER, Latvian, corpus indexing

Introduction

Recent digitizing of National Library of Latvia archives[1] has provided a valuable potential resource for researchers. In order to enable effective analysis and research, we aim to create a comprehensive catalogue of named entities mentioned in this corpus. It contains 240.000 books and newspapers (approx. 4.5 billion tokens) starting from 18th century up to year 2008, with a particular focus on Latvian publications of 1920'ies and 1930'ies. The corpus also includes a number of locally printed historical works in German, Russian and other languages, but the majority (70%) of the data is in Latvian, making it the largest curently available Latvian corpus.

The digitized data (scanned images and OCR results) is publicly available[2] and searchable. However, we consider common full-text indexing systems as not sufficient for enabling analysis of this corpus to the full extent. Linguistic analysis needs to take into account the morphological complexity of Latvian language, and in historical research the proper name spelling in documents would differ from searcher expectations due to historical reforms in Latvian orthography, morphology and also the large number OCR mistakes present in the digitized documents. Recognizing and properly indexing the named entities would provide a unique, valuable publicly available resource for Latvian historical, sociological and linguistic research.

This paper describes current efforts and results in augmenting the raw text corpus by automated tagging of morphological and named entity information, and offering the analysis results as publicly available online services for further research.

---

[1] Corresponding Author.

[2] www.periodika.lv

1. Tools for automated annotation of Latvian corpora

The large volume of this corpus means that any manual processing of documents is not feasible, and we needed to develop technologies to enable automated analysis of these documents. The current pipeline for document processing is structured as follows:

- Pre-processing of digitalised documents – extracting text and metadata from OCR results, and structuring them into subcorpora according to publication date, document type and language;
- Morphological analysis and disambiguation;
- Named entity recognition and classification;
- Identification of names that may refer to the same entity, including transliteration of historical documents to modern spelling, and partial correction of OCR mistakes;
- Indexing the text corpus morphological and named entity information in an online search engine.

A large part of these language software tools weren't readily available for Latvian and needed to be developed for this project. For example, automated transliteration of historic Latvian orthography is a separate research problem, described in [2], and integrating this solution was a required part for analysis of earlier parts of the corpus.

Performance was also a crucial factor for all the tools, requiring specific care and parallelization of software to achieve reasonable processing time for the corpus data – 660 Gb of annotated text.

The most significant new tool development for this purpose was an automated Latvian language named entity analysis system. Named entity recognition for Latvian is a relatively new research topic, with a single recently developed NER system TildeNER[3]. However, although this system is published as part of ACCURAT project, it was not practically usable, as it relies on proprietary language processing components and training data that isn't publicly available. It is also designed for analysing contemporary electronic documents, and there are significant differences in historical Latvian language and text domain, so in any case new training and tuning of the model would bre required. For these reasons we decided to develop a separate named entity classifier, learning from their experience but using domain-adapted training data and features as described in further chapters.

For the remaining parts of linguistic analysis we were able to adapt previously available tools with some modifications – SemTi Kamols morphological analyzer [4] and Bonito/Manatee corpus indexing system [5].

2. Named entity annotation standard

Our chosen taxonomy consists of 7 main types of named entities (person, location, organization, facility, event, product and time) and 21 subtypes of these groups. We have chosen not to annotate amounts – numerals and measurements in this corpus. Many types are further specified using subtypes (illustrated in Table 1) to add semantic precision to the manually annotated data, as it was considered relatively easy to specify more information while annotating and it provides extra options for using this data in further research.

Table 1. Named entity types and subtypes

Type	Subtypes				
PERSON	pers.hum	pers.imag	pers.anim		
LOCATION	loc.geo	loc.gsp	loc.addr	loc.other	
ORGANIZATION	org.gov	org.pol	org.game	org.com	org.other
FACILITY					
PRODUCT	prod.vehicle	prod.brand	prod.art	prod.printing	prod.award
TIME	time.date	time.other			
EVENT					

The types and subtypes are based on commonly used categories of named entities elsewhere, essentially a subset of Sekine's extended named entity hierarchy [6], but adjusting for our corpus (mainly newspapers and literature fiction) and expected use cases, including historical research.

The main difficulties that occurred in mark-up process are: 1) detecting the boundaries of named entities, 2) selecting the correct category type (subtype) in cases of ambiguity, as some named entities can seem to fall under two or more classes.

There are some named entities where it becomes difficult to find the right category, for example, if using a coarse-grained classification system such as MUC-7, then for some frequently occurring cases (universities, hospitals, etc) it is unclear if it should be annotated as an organization or location – it can depend on sentence semantic context. Our annotation guidelines prescribe to mark such cases with a separate 'facility' type as suggested by [6]. To deal with ambiguity the subtype *other* is defined for all types and used when it is not clear from the text which type or subtype to use, instead of subjectively (or randomly) choosing a category.

In an approach similar to [7], to show the structure of extended named entities in cases where parts of name are valid proper names independently (mainly in case of *facilities*, *organizations* and *events*), we are using a hierarchical annotation. For example, *Paula Stradiņa klīniskā universitātes slimnīca* 'Pauls Stradins Clinical University Hospital', is annotated as type *facility*, spanning the whole expression, and also includes an entity of type *pers.hum*, which spans the first name *Paula* and second name *Stradiņa* (name of the hospital's founder). This approach is very convenient to show extended named entities such as events, organizations and facilities that are very common in our corpus, especially the Soviet newspapers that traditionally use extended names for various entities.

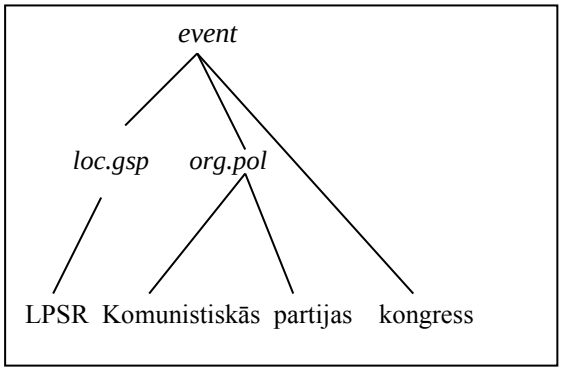


Figure 1. Example of a complex named entity.

While appositives are often used to provide auxiliary information for named entities, we are not annotating them as a part of those entities. For example, in the phrase *skolotājs Jānis Kalniņš* 'teacher Jānis Kalniņš' only the proper name is marked as a named entity. In cases where appositions carry vital information about the entity, for example, the phrase *Latvijas prezidents Kārlis Ulmanis* 'president of Latvia Kārlis Ulmanis' we would still mark only the name *Kārlis Ulmanis* as the entity, expecting to recover the appositions (titles, professions, etc) afterwards from the corpus if necessary.

Prepositions also are not included into the structure of named entities unless they are component of *time.data* or *time.other*, for example, ***no*** *2012. gada janvāra* ***līdz*** *decembrim* 'from the 2012 January till December';

3. Named entity recognition system

We have developed and are currently tuning and improving a named entity recognition machine learning system suitable for analysis of Latvian texts starting from mid-19th century, based upon the Stanford NER conditional random field (CRF) classifier [8] and the findings of TildeNER research in applying the Stanford NER system to Latvian language as published in [3].

For the training and testing the named entity recognition we have manually annotated an 150.000 word corpus of books and newspapers of different representative time periods, shown in Table 2. To facilitate fast annotation of this training corpus, we have developed an online annotation tool based on PHP and jQuery. The annotation process is based on mouse 'painting' the entities within sentence with a unique color for each role, and storing the results in a custom XML markup format suitable for the hierarchical annotation.

The training corpus is annotated according to the standard described in chapter 2, with a hierarchical annotation of overlapping named entities. However, automated annotation of such detailed classification is impractical with CRF methods, as the analysis time and space requirements grow rapidly with increased number of classification classes. Currently used classifier is trained by leaving only the outer, 'parent' named entities in case of nested names, and the full hierarchical tagging is still in development, as described in chapter 5.

Table 2. Manually annotated named entity corpus

Year	Type	Title	Size (words)
1861	Newspaper	Latviešu Avīzes	5 224
1863	Book	Tahiti salas ļaudis	5 612
1882	Newspaper	Arājs	10 346
1918	Newspaper	Baltijas Ziņas	19 152
1928	Magazine	Atpūta	12 354
1934	Book	Madonas vadonis tūristiem	2 471
1935	Newspaper	Mūzikas Apskats	13 129
1942	Newspaper	Sendergruppe Ostland	8 234
1957	Newspaper	Cīņa	15 568
1966	Book	Krusttēvs Oskars : atmiņas	11 505
1988	Newspaper	Padomju jaunatne	15 181
1988	Book	Kārlis Ulmanis	5 724
1999	Magazine	Zīlīte	2 460
2005	Book	Ugāles baznīca	5 206
2007	Magazine	Dadzis	18 791

In addition to common, language independent features provided by the Stanford NER system, we have extended the feature set to include morphological information (lemma, part of speech, and tags with information of gender, number, case and person) and an extensive gazetteer of place and person names.

We have also developed an online database system to manage the names identified in authoritative external sources[3] or by the automated named entity recognition in the corpus. This allows us to define a reference set of entities with links to both authoritative definitions and also to all their mentions in corpus, providing a base for further research on semantic relations between these entities. The system also allows users to specify related names, synonyms and pseudonyms, so that searching or analysis of such entities would identify all the relevant names. For some domains, such as pseudonyms of Latvian authors or historical changes of street names, this data is available in a structured form in the authoritative data sources and the links between names can be created automatically.

For documents in other languages (mainly German and Russian), we are using an off-shelf version of the Stanford NER system, but afterwards we are merging the results (links to documents where entities are mentioned) with the Latvian entity names by using the library authoritative data, which includes the alternative spellings of historical names in German and Russian.

4. Named entity recognition results and evaluation

Preliminary evaluation of the named entity analysis system was performed on two documents (historical novel *Kārlis Ulmanis* and newspaper *Zīlīte*) excluded from the training corpus. Accuracy was counted on the entity level, treating as correct only those entities with matching categories and exactly same borders as the human annotator. For evaluation purposes a three category classification (person, location, organization) was used to cover the most important entity classes and to be comparable with TildeNER system findings in [3].

We observe a significant difference in recognition quality between the two test documents. Our hypothesis from reviewing the annotated documents is that this difference is caused by two separate issues – the worse OCR quality of newspaper text, and the domain differences. These differences depending on document type (newspapers vs. books) seem to apply for the whole corpus. However, further research would be needed to objectively determine the reasons of such differences.

The system accuracy, illustrated in Table 3, is reasonably good when compared to TildeNER system reported F-measure scores of 60-68%, although this is not an exact comparison as the measurements were done on different sets of test data. We attribute the improved performance of our classifier to the larger amount of training data that we have been able to manually annotate. As both systems use same core principles and similar feature sets, there seem to be options for improvements to both systems by reusing and combining data and research where possible.

Table 3. Named entity recognition evaluation

Document	Precision	Recall	F-measure
Kārlis Ulmanis (book)	85.1 %	79.0 %	81.9 %
Zīlīte (newspaper)	75.4 %	63.5 %	68.9 %

[3] National Library authoritative data, Latvian geospatial information database and Wikipedia

We also performed a cursory manual review of automatically annotated documents from various periods for which we did not have a matching human annotation. Evaluation of the mistakes indicates that the most difficult part is the detection of organizations, due to the large number of cases where organization names start with a person or location name. Determination of organization name borders also is a bigger problem than borders for other named entities.

An initial hypothesis was that the oldest part of corpus (up to 1930'ies) would require a separate named entity classifier due to the historical changes in spelling rules. However, our experiments showed that splitting of training data for these periods results in worse accuracy than using all annotated data together. This allows us to perform the spelling changes after the main processing to merge names such as *Wahzija* and *Vācija* (Germany) into a single entity.

5. Conclusions and further work

We have developed a named entity recognition system for Latvian language with a competitive accuracy, and provided an online index of named entities found within a large historical corpus, linking them to authoritative databases of people and organizations.

This is also now by far the largest publicly available[4] morphologically annotated Latvian language corpus. The corpus size – 4.5 billion tokens – is rather large for a relatively small language as Latvian, and the adapted Bonito/Manatee corpus indexing engine enables efficient lexicographical analysis of word usage in this corpus.

Ongoing work includes further development of the named entity analysis system, using bootstrapping of newly identified named entities to extend the gazetteer used by named entity classifier, as recommended by [3] to improve the NER accuracy. Processing and indexing of the whole corpus data is scheduled to be completed by September 2012, at which point the named entity data would be publicly available at the web site of National Library of Latvia.

The main use case for the resulting named entity catalogue is to provide a publicly accessible service that would facilitate the use of National Library digitized corpus for all kinds of research.

Additional use cases and topics for further work include developing this resource into a semantic database of historic and current people and organizations in Latvia, using it together with the text corpus as a source for information extraction about named entity relations.

6. Acknowledgements

The preparation of this paper and development of the software systems has been supported by the European Regional Development Fund under the projects nr. 2DP/2.1.1.2.0/10/APIA/VIAA/011 and LNB/2011/33/ERAF. The authors would like to thank the National Library of Latvia for providing their corpus for analysis.

[4] Some restrictions apply – summarized data, concordances and limited length citations are available everywhere, but full text of some books and newspapers is available only on Latvian library network and for academic purposes due to copyright restrictions.

References

[1] A. Zogla, J. Skilters, Digitalization of Historical Texts at the National Library of Latvia. *Human Language Technologies – The Baltic Perspective (Baltic HLT 2010), Frontiers in Artificial Intelligence and Applications* (2010), 177–184.

[2] L. Pretkalniņa, P. Paikens, N. Grūzītis, L. Rituma, A. Spektors, Making Historical Latvian Texts More Intelligible to Contemporary Readers. *Proceedings of the workshop "Adaptation of Language Resources and Tools for Processing Cultural Heritage Objects" the Eight International Conference on Language Resources and Evaluation (LREC'12)*.

[3] M. Pinnis, Latvian and Lithuanian Named Entity Recognition with TildeNER. *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*.

[4] P. Paikens, Lexicon-Based Morphological Analysis of Latvian Language. *Proceedings of the 3rd Baltic Conference on Human Language Technologies (Baltic HLT 2007)*, Kaunas, 235–240.

[5] P. Rychlý, Manatee/Bonito - A Modular Corpus Manager. *1st Workshop on Recent Advances in Slavonic Natural Language Processing* (2007), 65-70.

[6] S. Sekine, K. Sudo, C. Nobata, Extended named entity hierarchy. *Proceedings of the Third International Conference on Language Resources and Evaluation* (2002).

[7] A. Savary, J. Waszczuk, A. Przepiórkowski, Towards the Annotation of Named Entities in the National Corpus of Polish. *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, 3622–3629.

[8] J.R. Finkel, T. Grenager, C. Manning, Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, 363-370.

PUBLIKĀCIJA V

Making historical Latvian texts more intelligible to contemporary readers

Proceedings of the workshop "Adaptation of Language Resources and Tools for Processing Cultural Heritage Objects" at the Eight International Conference on Language Resources and Evaluation (LREC 2012), 2012

Making Historical Latvian Texts More Intelligible to Contemporary Readers

Lauma Pretkalniņa, Pēteris Paikens, Normunds Grūzītis, Laura Rituma, Andrejs Spektors
Institute of Mathematics and Computer Science, University of Latvia
Raiņa blvd. 29, LV-1459, Riga, Latvia
E-mail: lauma@ailab.lv, peteris@ailab.lv, normundsg@ailab.lv, laura@ailab.lv, aspekt@ailab.lv

Abstract

In this paper we describe an ongoing work developing a system (a set of web-services) for transliterating the Gothic-based Fraktur script of historical Latvian to the Latin-based script of contemporary Latvian. Currently the system consists of two main components: a generic transliteration engine that can be customized with alternative sets of rules, and a wide coverage explanatory dictionary of Latvian. The transliteration service also deals with correction of typical OCR errors and uses a morphological analyzer of contemporary Latvian to acquire lemmas – potential headwords in the dictionary. The system is being developed for the National Library of Latvia in order to support advanced reading aids in the web-interfaces of their digital collections.

1. Introduction

In 2010, a mass digitalization of books and periodicals published from the 18th century to the year 2008 was started at the National Library of Latvia (Zogla and Skilters, 2010). This has created a valuable language resource that needs to be properly processed in order to achieve its full potential and accessibility to a wide audience, especially in the case of historical texts.

A fundamental issue in a massive digitalization of historical texts is the optical character recognition (OCR) accuracy that affects all the further processing steps. The experience of Tanner et al. (2009) shows that only about 70–80% of correctly recognized words can be expected in the case of the 19th century English newspapers. The actual OCR accuracy achieved in the digitalization of the National Library of Latvia (NLL) corpus has not been systematically evaluated yet[1], however, in the case of historical Latvian, at least two more obstacles have to be taken into account: the Gothic-based Fraktur script (that differs from the Fraktur used in historical German) in contrast to the Latin-based script that is used nowadays, and the inconsistent use of graphemes over time.

During the first half of the 20th century, the Latvian orthography has undergone major changes and has acquired its current form only in 1957[2]. The Fraktur script used in texts printed as late as 1936 is not familiar to most readers of contemporary generation. Moreover, the same phonemes are often represented by different graphemes, even among different publishers of the same period. The Latvian lexicon, of course, has also changed over time, and many words are not widely used and known anymore.

This makes a substantial obstacle in the accessibility of Latvian cultural heritage, as almost all pre-1940 printed texts currently are not accessible to contemporary readers in an easily intelligible form.

In this paper we describe a recently developed system for transliterating and explaining tokens (on a user request) in various types of historical Latvian texts.

In the following chapters, we first give a brief introduction to the evolution of the Latvian orthography, and then we describe the design and implementation of the system that aims to eliminate the accessibility issues (to a certain extent). We also illustrate some use-cases that hopefully will facilitate the use of the Latvian cultural heritage.

2. Latvian orthography

The first printed works in Latvian appeared in the 16th century. Until the 18th century the spelling was highly inconsistent, differing for each printed work. Since the 18th century a set of relatively stable principles has emerged, based on the German orthography adapted to represent the Latvian phonetic features (Ozols, 1965).

In 1870-ies, with the rise of national identity, there were first activities to develop a new orthography that would be more appropriate to describe the sounds used in Latvian: long vowels, diphthongs, affricates, fricatives and palatalized consonants (Paegle, 2001). This goes hand in hand with the slow migration from the Fraktur script to the Latin script. The ultimate result of these efforts was an alphabet that in almost all cases has a convenient one-to-one mapping between letters and phonemes, and is almost the same as the modern Latvian alphabet that consists of 33 letters. However, the adoption of these changes was slow and inconsistent, and both scripts were used in parallel for a prolonged time (Paegle, 2008). From around 1923, Latvian books are mostly printed in the Latin script, but many newspapers still kept using the Fraktur script until late 1930-ies due to investments in the printing equipment.

There were additional changes introduced in the modern orthography in 1950-ies, eliminating the use of graphemes ‘ch’ and ‘ŗ’, and changing the spelling of many foreign words to imitate their pronunciation in Russian. This once again resulted in decades of parallel orthographies: texts printed in USSR use the new spelling while texts published in exile resist these changes.

This presents a great challenge, as the major orthographic changes have occurred relatively late and, thus, a huge proportion of Latvian printed texts have been published in obsolete orthographies. Furthermore, the

[1] The expected accuracy is about 80% at the letter level.
[2] http://en.wikipedia.org/wiki/Latvian_language#Orthography

available linguistic resources and tools, such as dictionaries and morphological analyzers, do not support the historical Latvian orthography.

Figure 1 illustrates some of the issues that have to be faced in the processing pipeline if one would semi-automatically convert a text in Fraktur into the modern Latvian orthography. It should be mentioned that, in the scope of this project, OCR is provided by a custom edition of ABBYY FineReader (Zogla and Skilters, 2010).

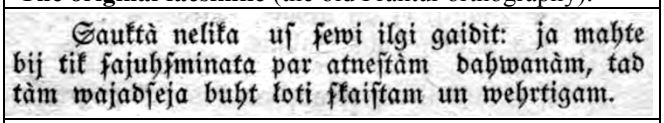
The original facsimile (the old Fraktur orthography):

The actual result of OCR:
Sauktà nelika us sewi ilgi gaidît: ja mahte bij tik sajuhsminata par atnestām dahroanàm, tad tàm roajadseja buht ļoti skaistam un wehrtigam.
The expected OCR result (Latin script, old orthography):
Sauktā nelika uz sewi ilgi gaidīt: ja mahte bij tik sajuhsminata par atnestām dahwanām, tad tām wajadzeja buht ļoti skaistam un wehrtigam.
Transliteration into the modern orthography:
Sauktā nelika uz sevi ilgi gaidīt: ja māte bija tik sajūsmināta par atnestām dāvanām, tad tām vajadzēja būt ļoti skaistām un vērtīgām.

Figure 1: A sample sentence in the historical Latvian orthography and its counterpart in the modern orthography along with intermediate representations.

3. Transliteration engine

We have developed a rule-based engine for performing transliterations and correcting common OCR errors. In this chapter we describe the engine assuming that rules defining the transliteration and error correction are already provided.

To satisfy the user interface requirements[3], the engine is designed to process a single token at a time. The workflow can be described as follows:

- The input data is a single word (in general, an inflected form).
- Find all transliteration rules that might be applied to the given word and apply them in all the possible combinations (thus acquiring potentially exponential amount of variants).
- Find the potential lemmas for the transliteration variants using a morphological analyzer of the contemporary language (Paikens, 2007).
- Verify the obtained lemmas against large, authoritative wordlists containing valid Latvian words (in the modern orthography) of various domains and styles, as well as of regional and historical lexicons.
- Assign a credibility level to each of the proposed variants according to the transliteration and validation results. In an optional step, the transliteration variants (both wordforms and lemmas) can be ranked according to their frequency in a text corpus.

Note that the contextual disambiguation of the final variants (if more than one) is left to the reader.

Below we shall describe most significant parts of the workflow in more depth.

3.1 Types of transliteration rules

Our transliteration engine uses two types of rules: obligatory and optional. The obligatory rules describe reliable patterns (usually for the standard transliteration, but also for common OCR error correction) that are always applied to the given word, assuming that in practice they will produce mistakes only in rare cases. When this set of rules is applied to a target string, only one replacement string is returned (except cases when a target string is a substring[4] of another target string; see Figure 2: 'tsch' vs. 'sch').

The optional rules describe less reliable patterns (usually for OCR correction, but also for transliteration) that should be applied often, but not always. I.e., the optional rules produce additional variants apart from the imposed ones (by the obligatory rules). When a set of optional rules is applied, it is allowed to return more than one replacement string for a given target string.

All rules are applied "simultaneously", and the same target string can be matched by both types of rules (e.g. a standard transliteration rule is that the letter 'w' is replaced by 'v', however, the Fraktur letter 'm' is often mistakenly recognized as 'w').

Figure 2 illustrates various rules of both types (some of them are applied to acquire the final transliteration in Figure 1). Note that OCR errors are corrected directly into the modern orthography (e.g. 'ro' is transformed into 'v' instead of 'w').

```
<rules>
  <obligatory>
    <str find="à" replace="ā"/>
    <str find="ah" replace="ā"/>
    <str find="w" replace="v"/>
    <str find="tsch" replace="č"/>
    <str find="sch" replace="š"/>
    <str find="ees" replace="ies" match="end"/>
  </obligatory>
  <optional>
    <str find="ro" replace="v"/>
    <str find="a" replace="ā"/>
    <str find="l" sensitive="yes">
      <replace>I</replace>
      <replace>J</replace>
    </str>
  </optional>
</rules>
```

Figure 2: A set of sample transliteration rules.

[3] The system will provide back-end services for reading aids (in a form of pop-up menus) in the web-interfaces of the NLL digital collections.

[4] The longest substring not necessarily is the preferable one.

For any rule it is possible to add additional requirements that it is applied only if the target string matches the beginning or the end of a word, or an entire word, and/or that the rule is case-sensitive.

Transliteration rules are provided to the engine via an external configuration file. The current implementation of the engine allows providing several alternative rule sets. An appropriate set of rules can be chosen automatically, based on the document's metadata, e.g. typeface, publication year and type (a book or a newspaper). For the NLL corpus, currently two separate rule sets are being used: one tailored for texts in the Fraktur typeface printed after year 1880, and the other – for texts in the Latin typeface starting from the first item until the transition to the modern spelling in 1930-ies. A work in progress is to develop a set of rules for earlier Fraktur texts of 1750–1880. In future, the rule sets can be easily specialized if it will be experimentally verified that it would be advantageous to remove (or add) some transformation rules, for example, when processing documents of 1920-ies.

3.2 Applying transliteration rules

When the transliteration engine is started, each set of rules is loaded into the memory and is stored in a hash map using the target strings as keys. This gives us the ability to access all the possible replacements for a given target string in effectively constant time[5].

Transformations are performed with the help of dynamic programming and memorization. Each token is processed by moving the cursor character by character from the beginning to the end. In each position we check if characters to the left from the cursor correspond to some target string. In an additional data structure we keep all transformation variants for the first character, for the first two characters, for the first three characters etc. The transformation variants for the first *i* characters are formed as follows (consult Figure 3 for an example):

- For every rule whose target string matches the characters from the *k*-th position till the *i*-th position, a transformation variant (for the *i*-th step) is formed by concatenating each transformation variant from the *k*-th step with the rule's replacement string.
- From each transformation variant in length *i*-1 form a transformation variant in length *i* by adding the *i*-th character from the original token if there is no obligatory rule with a target string matching the last character(s) to the left from the cursor.

When the cursor reaches the end of the string, the obtained transformation variants are sorted in two categories: "more trusted" variants that are produced by the obligatory rules only, and "less trusted" variants that are produced also by the optional rules.

In Figure 3, it appears that "dāroanām" is a more trusted variant than "dāvanām", although actually it is vice versa. The false positive variant is eliminated in the next processing step, while the other one is kept (see Section 3.3).

Input: dahroanàm			
Step 1:	d	Step 5:	dāro, dāv
Step 2:	da, dā	Step 6:	dāroa, dāva, dāroā, dāvā
Step 3:	dā, dā	Step 7:	dāroan, dāvan, dāroān, dāvān
Step 4:	dār	Step 8:	dāroanā, dāvanā, dāroānā, dāvānā
Output (Step 9): dāroanām, dāvanām, dāroānām, dāvānām			

Figure 3: Sample application of transliteration rules. The input comes from Fig. 1 (line 2, token 6). Consult Fig. 2 for the rules applied (producing the underlined strings).

To speed up the transliteration, it is possible for user to instruct the engine not to use the optional rules for the current token.

3.3 Verifying transliteration variants

If transliteration is performed in the way it is described in the previous section, it produces plenty of nonsense alternatives. Thus we need a technique to estimate which of the provided results is more credible. One such estimate is implicitly given by the differentiation between obligatory and optional rules.

Another way to deal with this problem is to obtain a large list of known valid words and check the transliteration variants against it. Typically these would be lists of headwords from various dictionaries, however, due to the rich morphological complexity of Latvian, word lists, in general, are not very usable in a straightforward manner, but we can use a morphological analyzer to obtain the potential lemmas for the acquired transformation variants.

The exploited analyzer (Paikens, 2007) is based on a modern and rather modest lexicon (~60 000 lexemes) – although a lot of frequently used words are the same in both modern and historical Latvian, there is still a large portion of words out of vocabulary. Therefore we use a suffix-based guessing feature of the analyzer to extend its coverage when the lexicon-based analysis fails.

Transliteration variants whose lemmas are found in a list of known words are considered more credible. Currently we use wordlists from two large Latvian on-line dictionaries: one that primarily covers the modern lexicon (~190 000 words, including regional words and proper names), and one that covers the historical lexicon (>100 000 words, manually transliterated in the modern orthography). To extend the support for proper names (surnames and toponyms), we also use the Onomastica-Copernicus lexicon[6].

In the whole transliteration process we end up with six general credibility groups for the transliteration variants:

1. Only the obligatory rules have been applied; lemmatization has been done without guessing;

[5] This is important for the future use-cases where the service will provide probabilistic full-text transliteration.

[6] http://catalog.elra.info/product_info.php?products_id=437

the lemma is found in a dictionary.

2. Only the obligatory rules have been applied; lemmatization has been done by guessing; the lemma is found in a dictionary.
3. At least one optional rule has been applied; lemmatization has been done without guessing; the lemma is found in a dictionary.
4. At least one optional rule has been applied; lemmatization has been done by guessing; the lemma is found in a dictionary.
5. Only the obligatory rules have been applied; the lemma could not be verified by a dictionary.
6. At least one optional rule has been applied; the lemma could not be verified by a dictionary.

For instance, if we take the variants from Figure 3, "dāroanām" is not found in the morphological lexicon and by guessing it might be lemmatized as "dāroana" (noun) or "dāroant" (verb) – none of these nonsense words can be found in a dictionary. However, "dāvanām" is both recognized by the morphological lexicon as "dāvana" ('gift') and is found in a dictionary. A sample of full output data that is returned by the transliteration and lemmatization service is given in Figure 4.

```
<translit input="dahroanàm">
  <group opt_rules="no" guess="no" dict="yes"/>
  <group opt_rules="no" guess="yes" dict="yes"/>
  <group opt_rules="yes" guess="no" dict="yes">
    <variant wordform="dāvanām">
      <lemma form="dāvana">
        <dict id="MEV"/>
        <dict id="SV"/>
      </lemma>
    </variant>
  </group>
  <group opt_rules="yes" guess="yes" dict="yes">
    <variant wordform="dāvānām">
      <lemma form="dāvāna">
        <dict id="MEV"/>
      </lemma>
    </variant>
  </group>
  <group opt_rules="no" dict="no">
    <variant wordform="dāroanām"/>
  </group>
  <group opt_rules="yes" dict="no">
    <variant wordform="dāroānām"/>
  </group>
</translit>
```

Figure 4: Sample output data returned by the transliteration and lemmatization service.

Usually each of these groups contain more than one variant, thus it would be convenient to sort them in a more relevant order, e.g. by exploiting wordform frequency information from a text corpus. For instance, "dāvāna" (in Figure 4) is a specific orthographic form of "dāvana"; it is not used in modern Latvian and is rarely used even in historical texts.

First, a reasonable solution (at the front-end) would be that variants that are verified by a dictionary are given to the end-user before other variants – such approach is justified by our preliminary evaluation (see Section 4). The verified variants that are found in a large on-line dictionary (tagged by 'SV' in Figure 4) can be further passed to the dictionary service to get an explanation for the possible meanings of the word (see Section 5).

Second, a pragmatic trade-off would be that lemmas that are obtained by applying the optional rules and are not found in any dictionary are not included in the final output to avoid overloading end-users with too many irrelevant options (again, see Section 4).

3.4 Alternative sets of transliteration rules

Linguists distinguish several general groups in which Latvian historical texts can be arranged according to the orthography used.

In the current architecture, the transliteration service receives a single wordform per request along with two metadata parameters: publication year and typeface (Fraktur or Latin). Publication type (a book or a news-paper) could be added if necessary.

Taking into account the general groups and the provided metadata, for each case there should be a specific, handcrafted set of transliteration and OCR correction rules. The metadata theoretically could be used for automatic selection of a rule set. However, in practice it cannot be guaranteed (considering an isolated wordform) that the selection is the most appropriate one, if all the parameters overlap between two groups (due to the fact that several historical orthography variants were used in parallel for a prolonged time, and changes were rather gradual). There is also an objective issue caused by the uniform OCR configuration that has been used for all texts in the mass-digitalization despite the orthographic variations. In the result, all potential rule sets would have to extensively deal with OCR errors overgenerating transliteration variants in order to improve recall. Therefore we have defined only two general rule sets: one for the Fraktur script, and one for the early Latin script (see Section 3.1 for more detail).

Theoretically, there are at least two (parallel) scenarios how this issue could be addressed in future. First, a specific OCR configuration (a FineReader training file) could be adjusted for each text group, running the OCR process again and enclosing configuration IDs in the metadata. To a large extent, this could be done automatically, involving manual confirmation in the borderline cases. However, our experiments with Fine-Reader 11 show that this would not give a significant improvement[7] and would not scale well over different facsimiles of the same group, i.e., it would not be cost-effective. Second, a larger text fragment could be passed along with the target wordform, so that it would

[7] For a book (1926) fragment, the accuracy in both cases is about 95% at the letter level and about 75% at the word level.

be possible to detect specific orthographic features by frequency analysis of letter-level n-grams and by analyzing the spelling of common function words. This would allow choosing an optimal set of transformation rules to ensure an optimal error correction and transliteration[8]. More tailored sets of rules should also decrease the amount of nonsense transliteration variants.

3.5 Disambiguation – a future task

The transliteration system, as described above, results in multiple options for possible modern spellings of a given wordform. While this is a usable approach in interactive use-cases for which the system has been initially designed, other applications that require full-text transliteration most likely require automatic disambiguation as well, receiving a single, most probable variant for each wordform.

A naive probability ranking could be obtained by comparing the variants against a word frequency table obtained from a modern text corpus of a matching genre (i.e., newspapers, fiction etc.), according to the metadata of the analyzed text. A more reasonable approach would be exploitation of a POS tagger of modern Latvian[9] to eliminate part-of-speech categories that are contextually unlikely possible. In addition, a word-level n-gram model of modern Latvian could be used, but there might be a lot of rarely used or out-of-vocabulary words, particularly in the case of the NLL newspaper corpus that includes a large number of proper names. The problem of transliteration can be also seen as a problem of machine translation between very similar languages. Statistical phrase-based techniques could be applied, similarly as it has been done for multilingual named entity transliteration (Finch & Sumita, 2009), however, it would require a parallel corpus.

4. Evaluation

The performance of each transformation rule set can be estimated by comparing an automatic transliteration of a historical text with a manually verified transliteration of the same text. We have identified several historical books that have been reprinted in the modern orthography with minor grammatical or lexical changes to the language. We have semi-automatically aligned several book chapters, and we have also manually transliterated several pages from newspapers of various time periods to obtain a small, but a rather representative tuning and test corpus (see Figure 5).

For the current target application – a reading aid for historical texts – we have evaluated the performance of the multi-option transliteration, attempting to minimize the number of variants that are returned while maximizing the accuracy rate – that the known correct variant is among the returned ones.

[8] This would even allow distinguishing more specific rule sets than it is possible by relying only on the (extended) metadata.

[9] e.g., http://valoda.ailab.lv/ws/tagger/ or the one developed by Pinnis and Goba (2011).

Year	Title	Type	Tokens
1861	*Latviešu avīzes*	newspaper, early Fraktur	1025
1888	*Lāčplēsis*	book, early Latin	4308*
			917
1913	*Mērnieku laiki*	book, Fraktur	2880*
			5438
1918	*Baltijas ziņas*	newspaper, Fraktur	1001

Figure 5: A parallel corpus used for tuning (*) and evaluation of transliteration rules.

The tuning corpus identified a number of additional historical spelling variations, and several systematic OCR mistakes that can be corrected with transliteration rules. Figure 6 shows the final performance on the tuning corpus. The results clearly show the importance of the dictionary-based verification and that it would not be reasonable to overload the end-users with the over-generating variants that are acquired by optional rules and that are not verified by a dictionary (`no_dict`, `opt_rules`). The other credibility groups give 97% accuracy on the tuning corpus with 2.77 variants per token.

Credibility group	Accuracy	Variants
`dict`, `no_opt_rules`, `no_guess`	55.6 %	0.63
`dict`, `no_opt_rules`, `guess`	6.1 %	0.14
`dict`, `opt_rules`, `no_guess`	31.1 %	0.73
`dict`, `opt_rules`, `guess`	3.7 %	1.00
`no_dict`, `no_opt_rules`	0.5 %	0.27
`no_dict`, `opt_rules`	1.4 %	30.61
No variant produced:	1.6 %	0

Figure 6: Evaluation on the tuning corpus: an average number of variants and accuracy (contains the correct variant) per credibility group (consult Section 3.3).

These results also indicate a ceiling for the possible accuracy of this method at around 98%, no matter how well the transliteration rules are improved. Manual review of unrecognised words shows that around 1% of words have been irreparably damaged by OCR, and around 1% of words are unique and out of vocabulary: foreign words, rare proper names etc., where many equally likely transliteration options would be possible.

Note that lemmatization by guessing has been necessary "only" in about 10% cases – the common word lexicons of historical and modern Latvian highly overlap.

In the evaluation we are counting only exact spelling matches (including diacritics), and we are counting only word tokens (excluding numbers, punctuation etc.). The evaluation of transliteration accuracy for various texts is shown in Figure 7.

Year	Type	Accuracy	Variants
1861	newspaper, Fraktur	87.7 %	3.12
1888	book, Latin	96.7 %	2.45
1913	book, Fraktur	96.6 %	3.19
1918	newspaper, Fraktur	88.8 %	2.81

Figure 7: Evaluation on the test corpus.

We have observed that the OCR mistakes in the NLL corpus can be tackled by the same means as orthography changes, significantly improving the output quality: from around 75% word-level accuracy in the source texts (books) to around 88% (for newspapers) and 97% (for books) after transliteration. The correlation between font-face changes and orthography developments, as well as the possibility to match the transformation results against a large lexicon allows tackling both problems simultaneously.

However, as the evaluation shows a significant accuracy difference between book and newspaper content, we have analyzed the structure of all identified errors. The errors have been grouped as unrepaired OCR mistakes, unrepaired lexical or spelling differences in the historical language, and errors in transliteration rules, as shown in Figure 8. This indicates that the technique is vulnerable to scanning quality (as the *Baltijas ziņas* facsimile is of a comparatively low quality), and that there is still a future work to be done in improving the lexical change repair rules for the 1860-ies and earlier texts.

Error type	***Latviešu avīzes***	***Baltijas ziņas***
OCR mistakes	28 (23.5%)	83 (74.1%)
Lexical differences	83 (69.8%)	12 (10.7%)
Malfunctioning rules	8 (6.7%)	13 (11.6%)
Other	0	4 (3.6%)
Total	119	112

Figure 8: Error analysis.

5. Dictionary service

On a user request, an unknown word (lemmatized in the modern orthography by the transliteration service) is passed to a dictionary service that is based on a large on-line dictionary of Latvian[10]. The dictionary contains nearly 200 000 entries that are compiled from the Dictionary of the Standard Latvian Language[11] and more than 180 other sources. It covers common-sense words, foreign words, regional and dialect words, and toponyms (contemporary and historical names of regions, towns and villages in Latvia). Explanations include synonyms, collocations, phraseologies and historical senses.

```
<dict id="SV">
  <entry src="KV">
    <word>
      <lemma>dāterēt</lemma>
      <gram>apv.</gram>
    </word>
    <sense>
      <def>Ātri un neskaidri runāt.</def>
    </sense>
  </entry>
</dict>
```

Figure 9: An entry returned by the dictionary service.

A simple entry returned by the dictionary service is given in Figure 9. It gives a meaning of a rarely used historical regional word for which even Google returns no hits (as of 2012-04-01).

6. Use-cases

The initial and primary goal is to integrate these services in the interactive user interface of an on-line digital library of historical periodicals[12], allowing users to get hints on what a selected utterance of a (historical) word means.

A future goal is to facilitate extraction and cataloguing of named entities in historical corpora. For this purpose, the transliteration engine will be integrated in a named entity recognition system that is currently being developed[13]. It will be used while indexing person names and other named entities mentioned in texts by mapping these names to their modern spelling. This will allow searching for proper names regardless of how they might be spelled in the historical documents.

7. Conclusion

We have designed and implemented a set of services that facilitate the accessibility of historical Latvian texts to contemporary readers. These services will be used to improve the accessibility of historical documents in the digital archives of the National Library of Latvia – a sizeable corpus containing about 4 million pages[14].

Our preliminary evaluation shows that the rule-based approach with dictionary verification works well even with a single rule set for all Fraktur texts, returning 2.89 variants in average with a possibility of 92.45% that the correct one is among them. Period-specific tuning of transliteration rules can raise the accuracy up to 96.5% for both books and newspapers.

A future task is to provide an automatic (statistical) context-sensitive disambiguation among these variants.

It has to be noted that the system is designed to be generic and extensible for other transliteration needs by specifying appropriate sets of lexical transformation rules. While currently it is aimed to be used for analysis of historical texts, future work could address the transliteration of modern texts in cases where different spelling is systematically used. For instance, transliteration to the standard language is necessary in the case of user-generated web content (comments, tweets etc.) where various transliteration approaches for non-ASCII characters have often been used in Latvian due to the technical incompatibilities and inconvenience of various systems or interfaces.

Acknowledgments

The preparation of this paper has been supported by the European Regional Development Fund under the project

[10] http://www.tezaurs.lv/sv/

[11] Latviešu literārās valodas vārdnīca. Vol. 1–8. Riga: Zinātne, 1972–1996 (>64 000 entries).

[12] http://www.periodika.lv/

[13] Unpublished work, expected to be ready by the end of 2012.

[14] It is expected that a working demo of these reading aids will be available in May 2012.

No. 2DP/2.1.1.2.0/10/APIA/VIAA/011. The authors would like to thank Artūrs Žogla from the National Library of Latvia and the anonymous reviewers for their comments.

References

Finch, A., Sumita, E. (2009). Transliteration by Bidirectional Statistical Machine Translation. In *Proceedings of the 2009 Named Entities Workshop (NEWS)*, Suntec, pp. 52–56

Ozols, A. (1965). *Veclatviešu rakstu valoda*. Riga: Liesma

Paegle, Dz. (2001). Latviešu valodas mācībgrāmatu paaudzes. Otrā paaudze 1907–1922. In *Teorija un prakse*. Riga: Zvaigzne ABC, pp. 39–47.

Paegle, Dz. (2008). Pareizrakstības jautājumu kārtošana Latvijas brīvvalsts pirmajos gados (1918–1922). In *Baltu filoloģija XVII, Acta Universitatis Latviensis*, pp. 89–102

Paikens, P. (2007). Lexicon-Based Morphological Analysis of Latvian Language. In *Proceedings of the 3rd Baltic Conference on Human Language Technologies (Baltic HLT 2007)*, Kaunas, pp. 235–240

Pinnis, M., Goba, K. (2011). Maximum Entropy Model for Disambiguation of Rich Morphological Tags. In *Proceedings of the 2nd Workshop on Systems and Frameworks for Computational Morphology*, Communications in Computer and Information Science, Vol. 100, Springer, pp. 14–22

Tanner, S., Muñoz, T., Ros, P.H. (2009). Measuring Mass Text Digitization Quality and Usefulness: Lessons Learned from Assessing the OCR Accuracy of the British Library's 19th Century Online Newspaper Archive. *D-Lib Magazine*, 15(7/8)

Zogla, A., Skilters, J. (2010). Digitalization of Historical Texts at the National Library of Latvia. In I. Skadiņa, A. Vasiļjevs (Eds.), *Human Language Technologies – The Baltic Perspective (Baltic HLT 2010)*, Frontiers in Artificial Intelligence and Applications, Vol. 219, IOS Press, pp. 177–184

PUBLIKĀCIJA VI

Automātiskas morfoloģiskas anotācijas izmantojums

Vārds un tā pētīšanas aspekti, 2013.

Pēteris Paikens (LU Matemātikas un informātikas institūts)

AUTOMĀTISKAS MORFOLOĢISKĀS ANOTĀCIJAS IZMANTOJUMS

Raksts apskata lietojumus teksta morfoloģiskajai anotācijai – tā papildināšanai ar vārdu morfosintaktisko īpašību marķējumu, sīkāk apskatot iespējas, kas kļūst pieejamas tad, ja morfoloģisko marķēšanu neveic manuāli, bet gan automātiski ar valodas tehnoloģiju palīdzību. Šādas anotēšanas gaitā tiek identificētas vārdu pamatformas, kā arī norādītas vārdformas īpašības (piemēram, locījums) un galvenās leksiskās īpašības, kas raksturo pašu vārdu un ir kopīgas visām tā vārdformām, piemēram, deklinācija, atgriezeniskums u.c. Homoformu gadījumā, ja vārdformai izolēti būtu iespējami vairāki anotācijas varianti, anotēšanas gaitā tiek izvēlēta pareizā vārda pamatforma un īpašības atbilstoši tā kontekstam teikumā. Rakstā apskatīti arī pieejamie šādi anotēti valodas korpusi – datorizētai analīzei pieejami apjomīgi tekstu kopumi (VPSV 2007, 196) un to lietojumi.

Morfoloģiskās anotēšanas metodes un to ierobežojumi

Kvalitatīvākos morfoloģiski anotētos resursus veido kvalificēti valodnieki, manuāli veidojot katras vārdformas anotāciju. Protams, jebkurā apjomīgā manuālā darbā tiek pieļautas kļūdas, tāpēc laba anotēšanas prakse prasa divu anotētāju piesaisti katra teksta caurskatīšanā, kas ļauj izlabot neuzmanības kļūdas un pamanīt iespējamās neskaidrības anotēšanas principos. Tomēr šāda metode ir ļoti darbietilpīga pat ar rīkiem, kas pēc iespējas atvieglo cilvēka darbu. Latvijas Universitātes Matemātikas un Informātikas Institūtā (turpmāk LU MII) jau daudzus gadus regulāri notiek šāda korpusa papildināšana, sākot ar Kristīnes Levānes-Petrovas aprakstītajiem principiem (Levāne, Spektors 2000), bet joprojām šādi anotēti ir tikai ap 50 000 vārdlietojumu.

Tā kā vairumam lietojumu ir nepieciešams būtiski lielāks korpusa apjoms, praksē daudziem mērķiem tiek lietoti automātiski anotēti korpusi. Morfoloģisko anotāciju var veidot automātiski, balstoties uz latviešu valodas morfoloģijas likumsakarībām un atbilstošu vārdnīcu, piemēram, ar autora agrāk aprakstītajām metodēm (Paikens 2007). Taču ir jāņem vērā, ka latviešu valoda ir morfoloģiski daudznozīmīga un tādēļ līdz šim pieejamos automātiski anotētajos korpusos daļai vārdu tika norādīti vairāki analīzes varianti. Korpusa analīze rāda, ka 50-55% vārdlietojumu nav viennozīmīga morfosintaktiskā interpretācija, apskatot tos bez teikuma konteksta (Paikens, Rituma, Pretkalniņa 2013). Piemēram, kā apraksta Kristīne Levāne-Petrova, Līdzsvarotajā mūsdienu latviešu valodas tekstu korpusā vārdu *sniegs* atradīs gan kā vīriešu dzimtes vienskaitļa 1. deklinācijas lietvārdu nominatīvā, gan arī kā darbības vārda nākotnes formu (Levāne-Petrova 2011).

Jaunums latviešu valodas korpusu veidošanā ir iespēja veikt automātisku daudznozīmības risināšanu[1] ar autora aprakstītajām mašīnmācīšanās metodēm (Paikens, Rituma, Pretkalniņa 2013). Līdzīgi risinājumi ir iepriekš veidoti arī uzņēmumā Tilde (Pinnis, Goba 2011), taču tie nav publiski pieejami. Šīs metodes ļauj datoram izvēlēties ticamāko variantu atbilstoši vārdlietojuma kontekstam teikumā, mācoties no cilvēka anotētajiem paraugdatiem. Attiecīgi tas ļauj rīkam nevis tikai veikt analīzi, palīdzot marķēšanu veikt cilvēkam, bet arī veikt galīgo marķēšanu tās pārbaudes un korekcijas. Šādas anotācijas precizitāte šobrīd ir ap 94%, kas ir būtiski zemāka nekā cilvēka veiktajai anotācijai, tomēr ir pietiekami augsta daudzām praktiskām vajadzībām. Šīs metodes galvenā priekšrocība ir iespēja anotēt lielus korpusus, kas ir mērāmi miljonos vai pat miljardos vārdlietojumu, kā arī pētījumos lietot ne tikai iepriekš sagatavotus korpusus, bet apskatīt arī „pēc pieprasījuma" sagatavotus datus – piemēram, jaunākās ziņas vai šauras nozares dokumentus.

Padziļināts anotācijas līmenis būtu korpusa sintaktiskā anotācija, kas iekļautu arī informāciju par vārdu saistāmību. Šādus korpusus veiksmīgi lieto citu valodu pētīšanā, piemēram, Annas Teilores (*Ann Taylor*) aprakstītais *Penn Treebank* korpusa lietojums angļu valodai (Taylor, Marcus, Santorini 2003) vai čehu valodas *Prague Dependency Treebank*[2]. LU MII pētnieces Laura Rituma un Lauma Pretkalniņa bija uzsākušas šāda korpusa veidošanu latviešu valodai, manuāli anotējot korpusu ar sintakses kokiem (Pretkalniņa, Rituma 2012), taču šādi ir iespējams izveidot tikai ļoti ierobežota apjoma korpusu dažu tūkstošu teikumu apjomā. LU MII šobrīd notiek darbs pie automātiskas sintaktiskās anotācijas rīku izstrādes (Pretkalniņa, Rituma 2013) ar mērķi darīt pieejamus[3] arī liela apjoma sintaktiski anotētus korpusus. Tas ļaus tiešākā veidā meklēt korpusā dažādu sintaktisko konstrukciju realizācijas.

Korpusa lietojums valodas izpētē

Kā jau agrāk aprakstījusi K. Levāne-Petrova, korpusu var labi lietot gramatikas un leksikas izpētē, kā arī valodas apguvē un tulkošanā (Levāne-Petrova 2011). Šajā rakstā vēlētos uzsvērt atšķirības starp iespējām, kuras paver dziļāk anotēti un/vai lielāki korpusi un ilustrēt piemērus pētnieciskām problēmām, kurās var palīdzēt šie resursi.

Pirmkārt, digitāli korpusi (arī neanotēti) ir svarīgs resurss leksikas izpētē. Apskatot kāda vārda lietojumus korpusā, var iegūt pilnīgu priekšstatu par dažādajiem veidiem un apkaimēm, kādās vārds izpaužas. Korpuss kalpo par apliecinājumu tam, kuras vārda nozīmes praksē tiek lietotas, kā arī var kalpot kā avots šo nozīmju klāsta noskaidrošanai – piemēram, kā Pedersenas aprakstītajā pieredzē dāņu valodas leksikogrāfijā (Pedersen 2012). Protams, secinājuma pamatotība ir atkarīga no korpusa apjoma – līdzsvarots korpuss vairāku miljonu vārdlietojumu apjomā raksturos tikai tipiskos valodas līdzekļus, savukārt – ja

[1] Automātiskās anotēšanas modulis pieejams https://github.com/PeterisP/LVTagger
[2] Aprakstīts http://ufal.mff.cuni.cz/pdt2.0/
[3] Jaunumi tiks publicēti www.korpuss.lv

korpuss aptver lielāko daļu no visiem kādā laika periodā iespiestiem tekstiem (kā, piemēram, Latvijas Nacionālās Bibliotēkas digitalizētie dati), var uzskatīt, ka tajā atrodamie piemēri pilnībā raksturo tā laika literāro valodu un korpusā neesošās konstrukcijas tobrīd valodā nelieto.

Otrkārt, piemēru atlase kāda konkrēta vārda vai vārdlietojuma apkaimei tiek lietota gramatikas jautājumu izpētē – atlasot un kvalitatīvi analizējot piemēros redzamās sintaktiskās saites. Šāda meklēšana būtu precīzāka un ērtāka sintaktiski anotētā korpusā, bet kamēr tāds nav pieejams, arī morfoloģiski anotētā korpusā var meklēt un atlasīt paraugus morfosintaktiskajiem šabloniem neatkarīgi no lietotā vārda – piemēram, atlasīt un tālāk pētīt visus teikumus, kuros ir lietots teikuma priekšmets datīvā.

Tāpat arī ilustratīvu valodas paraugu vai piemēru atlasei pētījumiem un teorētiskiem aprakstiem ir ieteicams lietot šādus korpusus. Tas nodrošina piemērus, kas atspoguļo faktisko valodas lietojumu, savukārt intuitīva piemēru veidošana reizēm noved pie 'mākslīgu' konstrukciju piemēriem, kas šķiet ticami taču praksē tomēr gandrīz netiek lietoti, kā arī rada risku ignorēt valodā sastopamas konstrukcijas, kas neatbilst pētnieka individuālajai intuīcijai par tipisku vai vēlamu lietojumu.

Vēl šādi anotētu tekstu var izmantot datorlingvistikas rīku izstrādē (LU MII mākslīgā intelekta laboratorijā vai citur). Korekta vārdu un locījumu noteikšana ir priekšnosacījums gan teikumu sintaktiskai analīzei, gan arī specializētiem rīkiem – piemēram, organizāciju nosaukumu noteikšanai, runas atpazīšanas modeļu izstrādei vai automatizētai teksta semantikas analīzei. Tāpat arī mašīntulkošanas statistisko metožu precizitāti būtiski uzlabo t.s. faktorēto statistisko modeļu lietošana, papildinot tulkojumu paraugdatus ar morfoloģisko anotāciju (Skadiņa, Virza, Pretkalniņa 2012).

Kvantitatīva valodas analīze

Liela apjoma morfoloģiski anotēti korpusi paver iespējas arī dziļākai kvantitatīvai valodas izpētei. Ja apskatāmā parādība vai vārds korpusā parādās dažus desmitus reižu, tad to vislabāk ir analizēt kvalitatīvi, apskatot visus piemērus. Savukārt, ja ir pieejami tūkstošiem un vairāk paraugu, tad ir iespējams kvantitatīvi analizēt to lietojumu valodā, kas var dot informāciju, kuru nevar viegli iegūt kvalitatīvā analīzē.

Pirmkārt, šādos korpusos var veikt korekti kvantitatīvu novērtējumu vārdu lietojuma biežumam. Ja kādā korpusā vārds A ir lietots piecas reizes, bet tā sinonīms B – vienu reizi, tad to var izraisīt arī viena autora vai runātāja valodas specifika; taču ja lielākā korpusā vārds A ir lietots 500 reizes un vārds B – 100 reizes, tad tas jau ir statistiski uzticams rādītājs šo vārdu lietojuma biežumam.

Otrkārt, ja pētāmo parādību var definēt kvantitatīvi – piemēram, to, kādos kontekstos atšķiras vārdu *vispārīgs* un *vispārējs* lietošana – tad tas paver iespējas ar salīdzinoši nelielu laika patēriņu papildināt šādus pētījumus ar apskatāmās parādības lietojuma atšķirībām dažādos žanros un laika periodos. Ja meklēšanas metodes ļauj ātri iegūt skaitlisku apkopojumu no specifiska tekstu

korpusa, tad šādus apkopojumus var iegūt arī no atsevišķiem, šaurākiem apakškorpusiem. Šāda analīze ļauj pētniekam izdarīt objektīvus secinājumus par atšķirīgo un kopīgo šajos valodas lietojumos dažādās runātāju grupās un par valodas izmaiņām laika gaitā.

Treškārt, vārdu apkaimes kvantitatīva analīze ļauj papildus biežākajai apkaimei identificēt arī specifiskāko apkaimi – saistītos vārdus, kas parādās gandrīz tikai šādā apkaimē un tādējādi raksturo vārdu labāk par tā biežāko apkaimi. Šāda rakstura pētījumi tiek plaši veikti citām valodām, bet latviešu valodai šāda analīze pagaidām praktiski nenotiek. Leksikogrāfijā šos pētījumus uzsāka Patriks Henks (*Patrick Hanks*), analizējot angļu valodu un izmatojot korpusa zināšanas objektīvākai vārdnīcu veidošanai (Hanks, Church 1989). Tālāk pētījumu metodoloģiju un izmantošanu ir būtiski attīstījis Ādams Kilgarifs (*Adam Kilgarriff*), un viņa vadībā veidotais *Sketch Engine* rīks (Kilgarriff, Rychly, Smrz, Tugwell 2004) šobrīd ir citām valodām plašāk lietotais korpusa rīku komplekts šādai analīzei. Ņemot vērā tā lietojuma aktualitāti, LU MII 2014. gadā ir plānots projekts šo rīku adaptācijai latviešu valodai.

Vārdu *'skices'* jeb specifiskā apkaime

Par vārda specifisko apkaimi var saukt tos vārdus, kas apkaimē parādās proporcionāli biežāk, nekā ārpus tās. Tādējādi var identificēt vārdiem īpaši raksturīgos saistītos vārdus, arī tad, ja tie ir salīdzinoši reti. Piemēram, korpusā verba *vajadzēt* apkaimē vārds *naudiņa* ir sastopams daudz retāk nekā *laiks* vai *palīdzība*, taču atšķirībā no tiem *naudiņa* praktiski neparādās citu verbu kontekstā, un verbu raksturo daudz specifiskāk.

Sākotnēji šādu apkaimi definēja atbilstoši vārdu kolokācijām – vārdiem, kas tekstā atrodas tuvu (Hanks, Church 1989). Tomēr šāda pieeja identificē arī daudz neatbilstošu vārdu tīri vārdu tuvuma sakritības dēļ. Morfoloģiski anotētu korpusu lietošana ļauj vārda apkaimi meklēt atbilstoši morfosintaktiskiem šabloniem, precīzāk identificējot saistītos vārdus un nošķirot tos pēc lomas – piemēram, analizējot verbam tipiskos teikuma priekšmetus atsevišķi no papildinātājiem.

Būtisks lietojums šādiem specifiskās apkaimes mērījumiem ir apkaimes salīdzināšana starp līdzīgiem vārdiem vai līdzīgiem korpusiem. Piemēram, tas ļauj automātiski identificēt apkaimes, kurās vārdu *vispārīgs* un *vispārējs* lietojums sakrīt un kurās tiek lietots tikai viens no variantiem. Tāpat arī šis paņēmiens ļauj identificēt vārdus un to apkaimes, kas ir raksturīgas specifiski kādai valodas apakškopai – piemēram, portugāļu valodas pētījumos šādi var identificēti vārdus, kuru lietojums atšķiras Eiropā un Brazīlijā lietotajai portugāļu valodai (Kilgarriff, Pomikálek, Jakubíček, Whitelock 2012).

Šobrīd latviešu valodai ir pieejami rīki un korpusi, kas ļauj šādus pētījumus veikt, ja ir pētnieciska interese ieguldīt laiku šādas analīzes veikšanai; taču pagaidām dziļāki pētījumi vēl nav veikti.

Pieejamie anotētie korpusi

LU MII šobrīd galvenokārt tiek lietoti turpmāk uzskaitītie latviešu valodas korpusi, kuriem ir veikta morfosintaktiskā anotācija. Daļa no šiem resursiem nav publiski autortiesību ierobežojumu dēļ, bet ir pieejami pētniecības mērķiem, sazinoties ar LU MII mākslīgā intelekta laboratoriju.

1. Līdzsvarotais mūsdienu latviešu valodas korpuss
   Vispārīgiem lietojumiem ieteicams ir šis korpuss, kurš ir pieejams vietnē www.korpuss.lv. Tajā šobrīd ir ap 4 miljoni vārdlietojumu un tas ir morfoloģiski anotēts. Tā kā korpusa saturs ir līdzsvarots, tad to var uzskatīt par reprezentatīvu mūsdienu valodas paraugu (Levāne-Petrova 2012).
2. Morfoloģiski anotēts paraugkorpuss
   LU MII iekšienē morfoloģijas rīku veidošanai un kvalitātes novērtēšanai tiek lietots ap 50 000 vārdlietojumu liels korpuss ar divu anotētāju pārbaudītu marķējumu.
3. Sintaktiski anotēts paraugkorpuss
   LU MII tiek veidots arī sintaktiski anotēts korpuss (*treebank*). Šobrīd tas satur ap 45 000 vārdlietojumu (3300 teikumu), taču rīki tā publiskai apskatei un meklēšanai vēl ir izstrādes stadijā.
4. Lietuviešu-latviešu-lietuviešu paralēlo tekstu korpuss (LiLa)
   Latvijas un Lietuvas pārrobežu sadarbības programmas 2007.– 2013. gadā atbalstītā projekta „Humanitārās izglītības pētniecības infrastruktūras izveide Austrumlatvijā, Lietuvā" (HipiLatLit) ietvaros ir izstrādāts Lietuviešu-latviešu-lietuviešu paralēlo tekstu korpuss 8 miljonu vārdlietojumu apjomā, kas ir pieejams vietnē www.korpuss.lv.
5. Latvijas Nacionālās bibliotēkas digitalizēto tekstu korpuss
   Vietnē http://korpuss.lndb.lv/ ir pieejami LNB digitalizētie teksti, kas ietver periodiku un grāmatas no 19. gs. līdz mūsdienām. Plaši pārstāvēti ir visi periodi, ar īpašu uzsvaru uz 1918.–1940. gadiem. Šis ir lielākais pieejamais latviešu valodas digitālais korpuss – kopumā virs 3 miljardiem vārdlietojumu. Šis korpuss arī ir morfoloģiski anotēts, taču tajā ir daudz dokumentu skanēšanas un digitalizācijas kļūdu. Tā kā vecākā korpusa daļa nav rakstīta mūsdienu ortogrāfijā, šiem dokumentiem morfoloģiskās anotācijas izmantošana ir stipri ierobežota, taču to var lietot PSRS laika dokumentu izpētei.
6. Tīmekļa tekstu korpuss (*'Web as Corpus'*)
   LU MII vārdu apkaimes analīzē tiek lietots arī ap 70 miljonu vārdlietojumu liels tīmekļa datu izvilkums. Apjoma dēļ tas ir piemērotāks kvantitatatīvai analīzei nekā līdzsvarotais korpuss, taču tajā ir daudz vairāk neformālas un kļūdainas valodas.

Ilustratīvas vaicājumu iespējas

Noslēgumam ir iekļauti paraugi tipiskiem vaicājumiem informācijai, kādu var iegūt no morfoloģiski anotēta korpusa un kas var būt noderīga valodas izpētē. Lietotāja saskarnes detaļas nav norādītas, jo tās ir atkarīgas no konkrētā korpusa meklēšanas programmatūras – bet šie piemēri ilustrē pētījumiem pieejamās iespējas, parādot to, kāda rakstura informāciju var ātri un efektīvi iegūt no esošajiem resursiem.

1. Vārdformas apkaime

   Meklējot konkrētu vārdformu, var iegūt piemērus no korpusa.

*principiāls līderis, kurš apliecinājis uzticību **valstij** gan kaujas laukā, gan valsts dienestā.*
*vienlaikus ar akciju apķīlāšanu par labu **valstij** Stokholmas arbitrāžas tiesā iesniedza pretprasību*
*profesionālim, un to mums, vecākiem, un arī **valstij**, kurā šiem skolotājiem uzticēts mācīt jauno*

2. Vārda apkaime – meklējot pēc tā pamatformas

   Morfoloģiski anotētā korpusā vārdiem ir identificētas pamatformas, kas ļauj meklēt visas vārdformas vienuviet.

*spīdekļi starp viņiem tagad saskaitāmi uz **rokas** pirkstiem: Arturs Karnišovs, Šarūns Jasikevičs*
*un mazāku civilo lidaparātu, un teroristu **rokās** bija kritis vēl viens no tiem.*
*pirkstus vairs nevarēju pakustināt, aptinu ap **roku** kaprona striķi un tā turējos, domādams,ka*

3. Vārdu raksturojošā apkaime pēc morfoloģiskā šablona (piem. apzīmētāji)

   Norādot ierobežojumus blakus vārdu morfoloģiskajam marķējumam (šajā gadījumā īpašības vārda un lietvārda saskaņojumu), var atrast saistītos vārdus.

*Autorei Laimai Muktupāvelai ir **brīvas rokas** interpretēt Emīlijas Benjamiņas dzīvi,*
*mazinātu aizdomas, ka fondu nauda nenonāk **īstajās rokās** un līdzekļu sadale notiek aizkulisēs.*
*No ābeles mātišķajiem zariem uz egles **asajām rokām** pārvietojas āboli.*

4. Sintaktiskās struktūras paraugi

   Bieži arī sintaktiskajām struktūrām var izveidot vaicājumus atbilstoši morfoloģiskajam marķējumam, kā piemērā salīdzinājuma konstrukcijai.

*ieviešanas strādā arī tādi savā jomā atzīti **uzņēmumi kā Hella** , problēmu ir vairāk nekā risinājumu.*
*vēl lielāks, ja nedarbotos tādi bremzējoši **faktori kā inflācija** .*
*izrādīts viens no pirmajiem velosipēdiem - **divritenis kā brīnums**. BMX divriteni, kuru šobrīd*

5. Vārda specifiskā apkaime – tipiskie apzīmētāji

   Šajā piemērā atlasīti tipiskākie apzīmētāji vārdam *'roka'*, norādot to biežumu korpusā.

kreisā	133
labā	352
atplesta	60
kaila	54
palīdzīga	50

6. Vārdu apkaimes salīdzinājums

   Apskatītajā piemērā salīdzināts dažu vārdu *'auto'* un *'mašīna'* apzīmētāju lietojums, norādot attiecīgo piemēru skaitu no korpusa. Šādi var viegli ieraudzīt lietojuma atšķirības, turklāt ir pieejami arī saistāmības

kvantitatīvie rādītāji, kas ļauj izdarīt pamatotus secinājumus arī tad, ja viens no salīdzināmajiem vārdiem ir sastopams daudz biežāk.

	auto	mašīna
smagis/smagā	0	65
skaitļošanas	0	34
ugunsdzēsēju	10	21
kravas	101	133
dienesta	181	19
apvidus	33	3

Protams, precīzos vaicājumus nosaka avotu un pētījuma specifika, taču šie valodas tehnoloģija lietojumu piemēri demonstrē pieejamās iespējas. Jaunākā informācija par publiski pieejamiem korpusiem tiks ievietota tīmekļa vietnē www.korpuss.lv.

Pateicības

Raksts ir sagatavots ERAF projekta 2DP/2.1.1.2.0/10/APIA/VIAA/011 ietvaros un prezentēts ar šī projekta finansiālu atbalstu.

APPLICATIONS OF AUTOMATED MORPHOLOGICAL TAGGING

Summary

Automated morphological tagging allows researchers to better analyze large corpora in language research. During the last few years Institute of Mathematics and Computer Science (IMCS) has created tools to effectively perform corpus morphological annotation and disambiguation, and multiple specific corpora are available for linguistic research.

A particular area of interest is quantitative analysis of word surroundings and collocational behavior (so called 'word sketches') that has proven useful in other languages but has not yet been attempted for Latvian language.

This paper describes the available tools, resources and corpora, aiming to facilitate interest in performing further linguistic research. Common use cases of available data are provided that may be useful in language analysis, including grammar studies and lexicography.

Atsauces

Hanks, Church 1989 – Hanks, Patrick, Church, Kenneth. Word Association Norms, Mutual Information, and Lexicography. *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics,* Stroudsburg, USA : Association for Computational Linguistics, 1989, 76.–83. lpp.

Kilgarriff, Rychly, Smrz, Tugwell 2004 – Kilgarriff, Adam, Rychly, Pavel, Smrz, Pavel, Tugwell, David.
The Sketch Engine. *Proceedings of Euralex,* France : Lorient, 2004, 105.–116. lpp.

Kilgarriff, Pomikálek, Jakubíček, Whitelock 2012 – Kilgarriff, Adam, Pomikálek, Jan, Jakubíček, Miloš, Whitelock, Pete.
Setting up for corpus lexicography. *Proceedings of the 15th EURALEX International Congress*, Oslo, Norway : Department of Linguistics and Scandinavian Studies, University of Oslo, 2012, 606.–612. lpp.

Levāne, Spektors 2000 – Levāne, Kristīne, Spektors, Andrejs. Morphemic Analysis and Morphological Tagging of Latvian Corpus. *Proceedings of the Second International Conference on Language Resources and Evaluation, vol. 2*., Paris : European Language Resources Association, 2001, 1095.–1098. lpp.

Levāne-Petrova 2011 – Levāne-Petrova, Kristīne. Morfoloģiski marķēta valodas korpusa izmantošana valodas izpētē. *Vārds un tā pētīšanas aspekti*: rakstu krājums 15(1). Liepāja : LiePA, 2011, 187.–193. lpp.

Levāne-Petrova 2012 – Levāne-Petrova, Kristīne. Līdzsvarots mūsdienu latviešu valodas tekstu korpuss un tā tekstu atlases kritēriji. *Baltistica VIII priedas,* Viļņa : Vilniaus universitetas, 2012, 89.–98. lpp.

Paikens 2007 – Paikens, Pēteris. Lexicon-based morphological analysis of Latvian language. *Proceedings of 3rd Baltic Conference on Human Language Technologies (HLT 2007),* Vilnius : Vytautas Magnus University, 2007, 235.–240. lpp.

Paikens, Rituma, Pretkalniņa 2013 – Paikens, Pēteris, Rituma, Laura, Pretkalniņa, Lauma. Morphological analysis with limited resources: Latvian example. *Proceedings of NODALIDA 2013*. Oslo : Linköping University Electronic Press, 2013, 267.–277. lpp.

Pedersen 2012 – Pedersen, Bolette Sandford. Lexicography in Language Technology. *Proceedings of the 15th EURALEX International Congress,* Oslo, Norway : Department of Linguistics and Scandinavian Studies, University of Oslo, 2012, 31.–46. lpp.

Pinnis, Goba 2011 – Pinnis, Mārcis, Goba, Kārlis. Maximum Entropy Model for Disambiguation of Rich Morphological Tags. *Systems and Frameworks for Computational Morphology, Communications in Computer and Information Science, 1, Volume 100, The 2nd Workshop on Systems and Frameworks for Computational Morphology (SFCM2011),* Heidelberg : Springer, 2011, 14.–22. lpp.

Pretkalniņa, Rituma 2012 – Pretkalniņa, Lauma, Rituma, Laura. Syntactic Issues Identified Developing the Latvian Treebank. *Proceedings of the 5th International Conference on Human Language Technologies — the Baltic Perspective, Frontiers in Artificial Intelligence and Applications, Vol. 247*, Amsterdam : IOS Press, 2012, 185.–192. lpp.
Pretkalniņa, Rituma 2013 – Pretkalniņa, Lauma, Rituma, Laura. Statistical syntactic parsing for Latvian. *Proceedings of NODALIDA 2013*. Oslo, Norvēģija: Linköping University Electronic Press, 2013, 279.–289. lpp.
Skadiņa, Virza, Pretkalniņa 2012 – Skadiņa, Ingūna, Virza, Madars, Pretkalniņa, Lauma. Angļu-latviešu statistiskās mašīntulkošanas sistēmas izveide: metodes, resursi un pirmie rezultāti. *Baltistica VIII Priedas*. Viļņa, 2012, 155.–168. lpp.
Taylor, Marcus, Santorini 2003 – Taylor, Ann, Marcus, Mitchell, Santorini, Beatrice. The Penn Treebank: An Overview. *Treebanks: Building and Using Parsed Corpora.* Springer Netherlands, 2003, 5.–22. lpp.
VPSV – *Valodniecības pamatterminu skaidrojošā vārdnīca.* Atbildīgā redaktore V. Skujiņa. Rīga, 2007.

PUBLIKĀCIJA VII

Morphological analysis with limited resources: Latvian example

Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013)
NEALT Proceedings Series 16, lpp. 267–278, Oslo, 2013.

Morphological analysis with limited resources: Latvian example

Pēteris PAIKENS, Laura RITUMA, Lauma PRETKALNIŅA
University of Latvia, Institute of Mathematics and Computer Science
peteris@ailab.lv, laura@ailab.lv, lauma@ailab.lv

ABSTRACT

We describe an approach for morphological analysis combining a rule-based word level morphological analyzer with statistical tagging, detailing its application to Latvian language. Latvian is a highly inflective Indo-European language with a rich morphology.

The tools described here include an implementation of Latvian inflectional paradigms, a morphological analysis tool with a guessing module for out-of-vocabulary words, and a statistical POS/morphology tagger for disambiguation of multiple analysis possibilities. Currently achieved accuracy with a training set of only ~40 000 words is 97.9% for part of speech tagging and 93.6% for the full morphological feature tag set, which is better than any previously publicly available taggers for Latvian.

We also describe the construction and methodology of the necessary linguistic resources – a morphological dictionary and an annotated morphological corpus, and evaluate the effect of resource size on analysis accuracy, showing what results can be achieved with limited linguistic resources.

KEYWORDS : morphology, inflective language, POS tagging, Latvian language, morphological corpus.

1 Introduction

For inflective languages, where a large part of grammatical meaning is expressed by the morphological features of words, a wide variety of computational tasks require a way to perform automated part of speech tagging and morphological analysis. It is needed both in specialized use cases such as linguistic research, and also in end-user tasks such as searching within documents or automated spelling correction.

Smaller languages usually have a limited amount of resources and effort available for developing linguistic resources such as annotated corpora or dictionaries, so it is useful to explore analysis methods that would work with smaller amount of resources and allow reusing software tools developed for other, larger languages.

In this paper we describe the construction process of such a toolkit for Latvian language, integrating word-level morphological analysis based on a formalization of Latvian inflection paradigms with a statistical tagger to exploit sentence context. We also provide an evaluation for the effect of resource (annotated corpora and lexicon) size on analysis accuracy.

2 Latvian Morphology

Latvian is an Indo-European language with around 1.5 million native speakers. It is a synthetic inflected language with rich morphology somewhat similar to the commonly analyzed Czech morphology (Hajič, 2000).

Latvian nouns and pronouns have 6 cases in singular and plural in traditional grammar. Nouns are traditionally divided in 6 declensions with different inflectional paradigms. Adjectives, numerals and some participles have 6 cases in singular and plural, 2 genders (masculine and feminine) and separate definite and indefinite forms. In verb conjugation system are 2 numbers (singular and plural), 3 persons, 3 tenses (present, future and past, both simple and compound) and 5 moods, as well as multiple types of participles. Qualitative adjectives and adverbs formed from these adjectives have also degrees of comparison noted in their word form.

The morphology creates more than 200 verb and participle forms derived from each verb lexeme, and more than 100 forms for each adjective. Many of the endings are overlapping, creating homoforms – for example, singular accusative and plural genitive forms are identical for many words.

2.1 Related Work on Latvian Morphological Analysis

The earliest experiments with automated Latvian morphological analysis have been performed in 1970s (Drīzule, 1978), implementing noun and adjective analysis. In 1990s, with the advent of personal computers, there have been multiple attempts to create analysis systems for all parts of speech (Greitāne, 1994; Levāne & Spektors, 2000; Sarkans, 1996, Vasiļjevs, Ķikāne & Skadiņš, 2004) based on linguistic rules for word endings and morphemes.

Systems currently being used in practice for Latvian morphology include lexicon based analysis systems (Paikens, 2007; Skadiņa, 2004) – while requiring more computational and dictionary resources, such systems provide better accuracy than earlier research.

Morphological analysis of Latvian is rather ambiguous – about half of words have multiple valid interpretations if viewed without context, so disambiguation as analyzed in this paper is an important open problem. There exists a recently developed morphological tagger based on Maximum Entropy Model (Pinnis and Goba, 2011), but it is not available to public[1].

3 Development of a Morphologically Annotated Corpus

A morphologically annotated corpus is a key resource for all further work – even for methods that do not require input from a large corpus, it is crucial to have at least a small set of verified data that can be used for testing and evaluation.

3.1 Morphological Annotation Standard

The morphological feature annotation standard used for Latvian corpora was initially (Levāne, 2000) derived from the annotation principles used for other languages in the MULTEXT-East project (Erjavec, 2004). It is a way to represent word annotation with a short tag, each character position representing a separate, independent feature. The meaning of each character position depends on the part of speech (marked in the first character) in order to keep the tag length short enough for human reading.

For an example, Figure 1 illustrates the morphological feature tag for noun *draugam*, the singular dative form of *draugs* (a friend).

Tagset for noun	part of speech	type	gender	number	case	declension
	n	**c**	**m**	**s**	**d**	**1**
	noun	common	masculine	singular	dative	first

FIGURE 1 – Example of a morphological tag for noun *draugam* ('friend') **ncmsd1**

It should be noted that in addition to purely morphological features, the annotation includes also lexical properties (such as type and declension in Figure 1) necessary for other research uses of the annotated corpora. The tag element names and values are matched to the ISOcat standard as recommended by CLARIN project[2].

The annotation process starts with generating the possible readings with an automatic analyzer described in the next section, and then a manual review and entry of missing

[1] The tagger is used in company Tilde proprietary tools - the training data and tagger are not available for other research purposes.

[2] http://www.clarin.eu

features. The speed of annotation is around 300 words per hour for a skilled operator with appropriate software tools.

3.2 Annotated Corpora

There have been multiple efforts on building morphologically annotated corpora of Latvian. Currently publicly available corpora are shown in Table 1. As noted earlier, the corpora were developed for projects of varying goals, and there are some differences between exact annotation standards used.

Corpus	Text source / domain	Tokens	Sentences
Balanced	Latvian Balanced Corpus[3]	50 795	3 940
Legal	EU documents[4]	23 359	1 038
Plāns Ledus	A fiction book	16 708	1 314
Latvijas Vēstnesis	A newspaper	28 956	2 035

TABLE 1 – Morphologically annotated Latvian corpora.

These corpora have been reviewed by a single annotator only. To ensure adequate data quality we performed a second annotator review and correction of the balanced corpus annotation to reduce the number of annotation errors, and serve as a valid 'gold standard' data for analyzer training and evaluation in this paper.

4 Automated Morphological Analysis

Our basic morphological analysis – generation of all possible morphological interpretations of a word form – is based on an earlier publicly available lexicon-based morphological analyzer (Paikens, 2007), extending it with additional lexical data. It is based on matching possible word form endings and the inflectional changes to stems as described in classical linguistic research, and verifies the stem candidates against a lexicon marked with declensions and conjugations of common nouns and verbs.

The currently used morphological lexicon has been assembled from multiple sources, including an electronic version of an inverse dictionary (Soida, 1970), manual review of the closed word classes and words with irregular inflection, scientific terminology data, and updates based on . It is not properly balanced – the contents reflect what resources were available, so coverage may vary depending on the text domain. The lexicon contains 47 000 lexemes.

Even with such lexicon size, 5-6% of test data is still out of vocabulary. Most of these words are formed according to Latvian grammatical rules, so it is still reasonable to deduce morphological properties based on the word ending, and for these cases, a 'guessing' system is implemented that generates a large number of possible analysis

[3] http://www.korpuss.lv

[4] White Paper. Preparation of the Associated Countries of Central and Eastern Europe for Inte-gration into the Internal Market of the Union.

options. This includes the correct reading for all except some 0.5-1% foreign words or brand names that are used literally as inflexive nouns, but happen to have an ending that matches a Latvian flexive form.

5 Statistical Disambiguation Methods

For many languages, pure morphological analysis will have a significant amount of ambiguity. For Latvian, our current analyzer gives multiple interpretations for 50-55% of words, with an average of 3.5-3.8 options for ambiguous words, depending on text domain, and similar amount of ambiguity has been observed in other morphologically rich languages (Yuret & Türe, 2006). The above ambiguity measurement includes morphological features – part of speech, case, number, gender, etc., and also lemmas in case of inflectional homonymity.

We examine two main use cases for disambiguation – choosing the most likely option for a single token, or selecting the most likely morphological tags for a whole sentence, looking at words in context. Single token analysis has less data for accurate disambiguation, but can be used in analysis of incomplete text fragments such as search queries, and is simpler to implement.

5.1 Baseline - Single Token Disambiguation

If there are multiple valid interpretations, clearly some of them are more frequent than others – we can intuitively note that some inflective forms may be more commonly used; or that one of theoretically possible lemmas is a rare, archaic word.

For this scenario, we can count the frequencies in a morphologically disambiguated corpus for two main features – the inflectional paradigm that generated the option, and the lexicon entry (if any) of the source lemma. This allows a quick estimation of the likelihoods, choosing the analysis option with the most likely paradigm and lexeme. While this method is naturally limited, it provides reasonable results with very tiny resources, providing us with a baseline to evaluate more complex options described later.

This is similar to the first stage of a Brill tagger if the surface form was seen in training corpus, but this heuristic generalizes well also to cases where the exact form was not seen before.

5.2 Morphological Tagging Within a Sentence

There are two main directions to use sentence context in disambiguation of homoforms in order to appy the appropriate morphological tags. One approach would be to invoke syntax rules, such as general syntactic analyzers (e.g. Bārzdiņš, Grūzītis, Nešpore & Saulīte, 2007 or Deksne & Skadiņš, 2011) that could also be adapted for morphological disambiguation. On the other hand, it is also possible to obtain these rules directly from an annotated corpus with machine learning algorithms. Our initial experiments with available Latvian syntactic analysers gave poor results due to limited syntactic coverage, driving us to the machine learning direction – although other research (Hajic,

Krbec, Kveton, Oliva & Petkevic, 2001; Hulden & Francom, 2012) suggests that a hybrid approach may bring further improvements.

Further in description we use our currently best performing solution, a conditional Markov model (CMM) based morphological tagging module. We have also trained various other systems, including hidden Markov model (HMM) and conditional random field (CRF) based classifiers, but we achieved better results with CMM.

The CMM module software is a modified version of the Stanford NLP[5] system CMM classifier implementation (Toutanova, Klein, Manning & Singer, 2003). A major difference between our solution and the original Stanford POS-tagger is the integration of the classifier with a rule based morphological analyzer supplying multiple possible analysis options to the classifier for disambiguation.

The standard approach for other languages (Hulden & Francom, 2012; Toutanova, Klein, Manning & Singer, 2003; Gahbiche-Braham, Bonneau-Maynard, Lavergne & Yvon, 2012) is to train a classifier on features directly derived from the word form string, such as letter n-grams, capitalization features, etc. While this may be effective for languages with a smaller range of word forms, this is not optimal for morphologically rich languages, as suggested by research in other languages (Youret & Türe, 2006). Word form specific features would greatly suffer from feature sparsity, as even in a huge training corpus many rarer word forms would not be seen at all; and a large part of word ending inflection rules cannot be adequately captured by letter n-gram features.

However, this morphological knowledge can be exploited by adding as training features the results from rule based morphological analysis described in section 4. That gives a reasonably accurate (contains correct form in 98% cases) list of what tags seem possible for each word. So in addition to the used classifier training features commonly used for other languages, we also supply a list of possible part-of-speech and tag options for the selected word and its closest neighbours. We also provide a 'recommended' POS and tag, calculated as described in section 5.1, which gives ~1% additional boost in accuracy. This change augments the machine learning of ending (letter n-gram) relations with morphological features with the linguistic rules in analyser, and allows to achieve good results with rather small training corpora.

6 Evaluation

6.1 Methodology

We used a morphologically annotated balanced corpus of 50 795 words, using 46 306 of it as training data (5 344 of it for tuning and developing the systems), and a separate set of 4 489 words for evaluation in this paper. Text content is taken as-is from the corpus, leaving intact any spelling issues or insertions of foreign words.

Lexical features such as declension, verb modality, semantic grouping, etc. are discarded for both training and evaluation data, as they can be retrieved afterwards from

[5] http://nlp.stanford.edu/software/tagger.shtml

the lexicon when the lemma is determined. The following morphological features are used for evaluation: part of speech, gender, number, case, person, verb mood, and definiteness for adjectives and participles.

6.2 Rule-based analyzer module evaluation

On our test corpus, the rule-based morphological analysis module includes a correct analysis option for 98.2% words, incorrect analysis for 1.3% words, and no analysis for 0.5% words (mostly insertions from other languages). Rule-based analysis results are unambiguous for 46.6% words, and the ambiguous words have on average 3.8 options each.

6.3 Statistical disambiguation methods

Comparing the results of automatic morphological disambiguation on the evaluation data set shows a tag accuracy level of 87.0% for baseline single token analysis and 93.6% for the best performing CMM model.

Both methods are suitable for analysis of large text corpora, with single token analysis being able to analyze approx. 100 000 words per second per core on a 2.8Ghz processor, and the CMM tagger around 3 000 words per second.

Reviewing the distribution of disambiguation errors by feature category (break-down shown in Table 2) indicates that the most common error is a combination of number and case mismatch, confusing singular accusative and plural genitive forms of nouns or whole noun phrases. These are homoforms for a large portion of Latvian nouns and adjectives, and both accusative and dative may be syntactically reasonable after a verb, indicating respectively the object or recipient of the action. We plan to reduce this class of errors by integration of morphological disambiguation with deeper syntactic analysis (statistical dependency parsers) that should be able to better resolve such ambiguities.

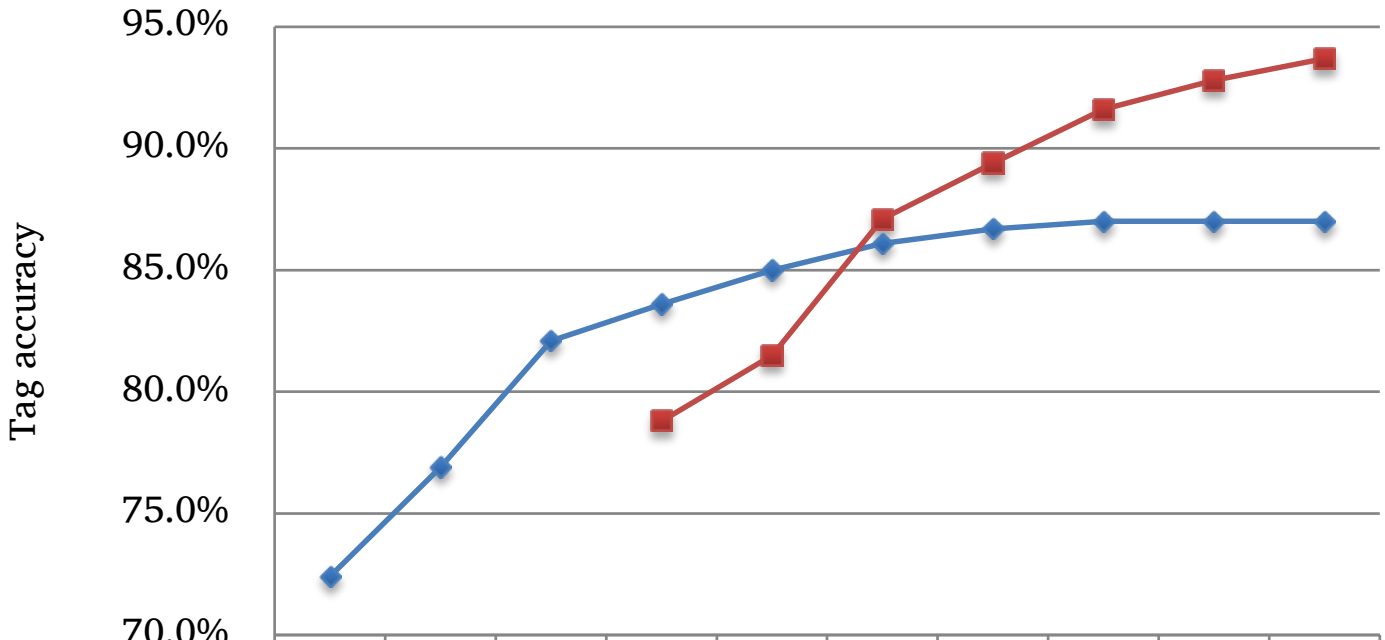
Part of speech	2.1 %
Gender	3.2 %
Number	4.5 %
Case	7.0 %
Verb mood	1.8 %
Person	0.8 %
Definiteness	1.4 %

TABLE 2 – CMM tagger error rates within feature categories.

6.4 Training data size effect on accuracy

Experiments on running the same disambiguation methods with limited training data, illustrated in Figure 2, show that the naive single token disambiguation quickly reaches its limit at around 10 000 words already. The CMM based model would likely provide

better accuracy with additional training data, which is also supported by experiments of Pinnis and Goba (2011) performed on a training set of 117 000 words, but it already provides an improvement above the single-token baseline with even very small training corpus such as 5 000 words.



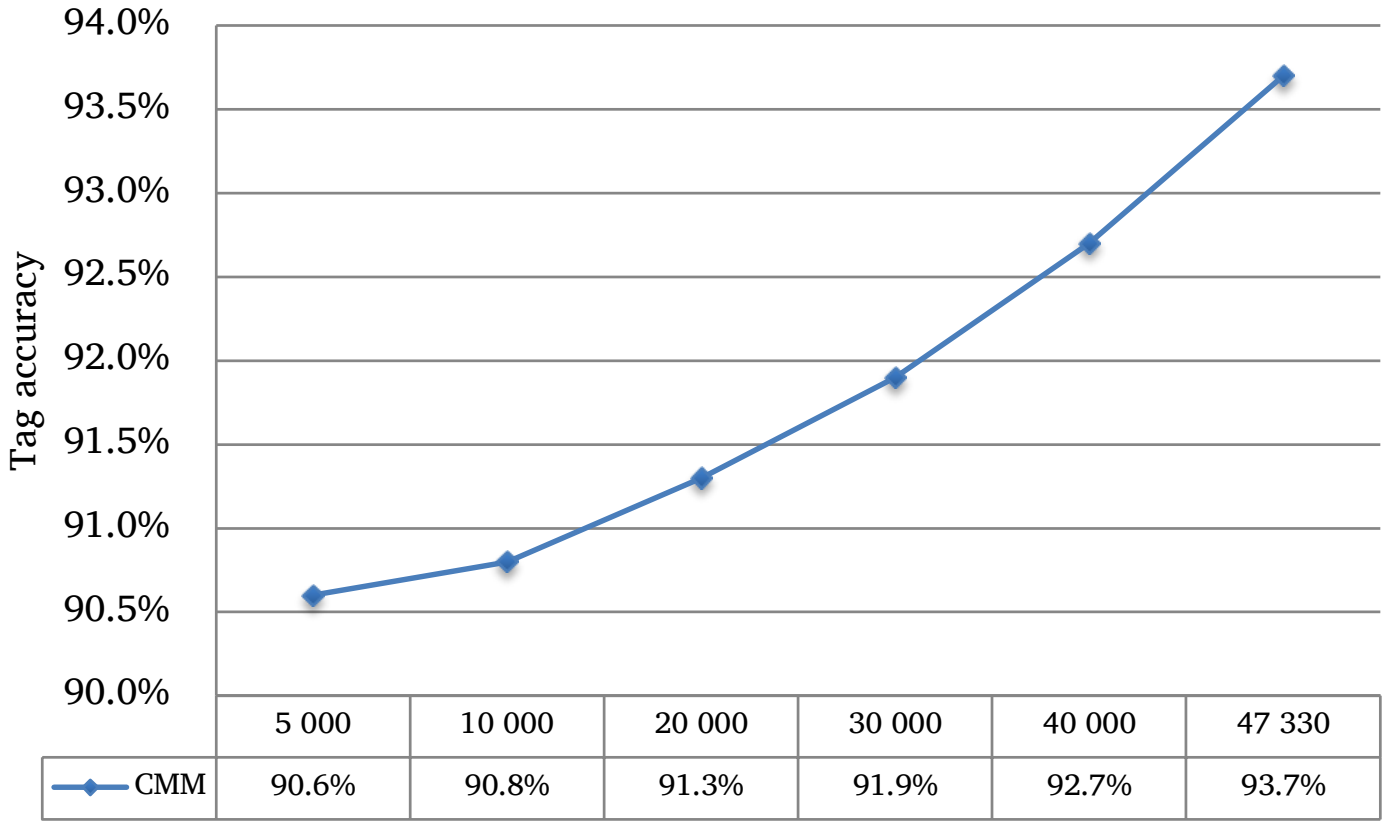
	0	100	500	1 000	2 000	5 000	10 k	20 k	30 k	46 k
Single-token	72.4%	76.9%	82.1%	83.6%	85.0%	86.1%	86.7%	87.0%	87.0%	87.0%
CMM				78.8%	81.5%	87.1%	89.4%	91.6%	92.8%	93.7%

FIGURE 2 – Effect of corpus size on CMM disambiguation accuracy compared to single-token baseline

6.5 Effect of Lexicon Size on Accuracy

To evaluate the necessity of a morphological lexicon (a dictionary annotated with declensions or inflectional paradigms), we performed a series of tests, training and running the CMM classifier with an artificially reduced lexicon. The minimal dictionary contains 5 000 lexemes for the closed word classes – pronouns, conjunctions, prepositions, and irregular verbs, with further experiments measured by randomly adding nouns and verbs from the full dictionary up to the indicated limit.

The evaluation results shown in Figure 3 indicate that a proper lexicon has a strong impact in reducing error rate, however, when considering languages or dialects where large dictionaries are unavailable (such as the Latgalian language closely related to Latvian), it is not strictly necessary since our experiments show a practically usable accuracy of 90.6% even with the minimal lexicon.



	5 000	10 000	20 000	30 000	40 000	47 330
CMM	90.6%	90.8%	91.3%	91.9%	92.7%	93.7%

FIGURE 3 – Effect of corpus size on CMM disambiguation accuracy compared to single-token baseline

7 Conclusion and Outlook

We have developed a freely available morphological analysis and disambiguation solution for Latvian language. The tools, resources and corpora are publicly available under an open source licence.

We demonstrate that morphological analysis and disambiguation for languages with rich morphology can be performed with small amounts of language-specific resources. In particular, if the inflection rules can be formally defined, then a morphological tagging module with a useful accuracy of 90% can be trained even with a small annotated corpus of 10-20 thousand words and a limited dictionary.

We expect to further improve accuracy of the morphological tagger by extending the training data up to 100 000 words and exploring options for integration with syntactic parsers.

A future goal is to attempt to apply this methodology for Latgalian language – a regional language with approx. 165 000 native speakers and very limited digital resources. We also hope that this experience can inspire linguistic tool development for other languages with limited size of corpora, noting that a practically useful accuracy can be obtained with very limited language data.

Acknowledgments

We thank the University of Latvia for the financial support in preparing and presenting this paper, and the anonymous reviewers for their improvement recommendations.

References

Bārzdiņš G., Grūzītis N., Nešpore G. and Saulīte B. (2007). Dependency-Based Hybrid Model of Syntactic Analysis for the Languages with a Rather Free Word Order. In *Proceedings of the 16th Nordic Conference of Computational Linguistics*, pages 13–20, Tartu.

Erjavec, T. (2004). MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'2004)*, pages 1535–1538, Paris.

Deksne, D. and Skadiņš, R. (2011). CFG Based Grammar Checker for Latvian. In *Proceedings of the 18th Nordic Conference of Computational Linguistics NODALIDA 2011*, Riga, Latvia.

Drīzule, V. (1978). Об автоматическом распознавании омонимии флексий латышского языка [On automated recognition of flexive homonymy in Latvian language]. In *LZA Vēstis 1978*, 10, pages 79–87, Rīga, LZA.

Gahbiche-Braham, S., Bonneau-Maynard, H., Lavergne, T. and Yvon, F. (2012). Joint Segmentation and POS Tagging for Arabic Using a CRF-based Classifier. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

Greitāne I. (1994). Latviešu valodas lokāmo vārdšķiru locīšanas algoritmi. (Algorithms for Latvian Form Generation) In *LZA Vēstis 1994*, 1, pages 32–39, Rīga, LZA.

Hajic, J., Krbec, P., Kveton, P., Oliva, K. and Petkevic, V. (2001). Serial combination of rules and statistics: A case study in Czech tagging. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 268–275.

Hajič, J. (2000). Morphological Tagging: Data vs. Dictionaries. In: *Proceedings of the 6th Applied Natural Language Processing and the 1st NAACL Conference*, pages 94-101, Seatle, Washington, U.S.A.

Hulden, M. and Francom, J. (2012). Boosting statistical tagger accuracy with simple rule-based grammars. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

Levāne, K. and Spektors A. (2000). Morphemic Analysis and Morphological Tagging of Latvian Corpus. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, vol. 2, pages 1095–1098.

Levāne-Petrova K. (2011). Morfoloģiski marķēta valodas korpusa izmantošana valodas izpētē. In *"Vārds un tā pētīšanas aspekti": Rakstu krājums 15(1)*, pages 187–193, Liepāja, LiePA.

Paikens, P. (2007). Lexicon-based morphological analysis of Latvian language. In *Proceedings of 3rd Baltic Conference on Human Language Technologies (HLT 2007)*, pages 235-240, Kaunas.

Pinnis, M. and Goba, K. (2011). Maximum Entropy Model for Disambiguation of Rich Morphological Tags. In *Systems and Frameworks for Computational Morphology,*

Communications in Computer and Information Science, 1, Volume 100, The 2nd Workshop on Systems and Frameworks for Computational Morphology (SFCM2011), pages 14-22, Heidelberg, Springer.

Sarkans U. (1996). Morphemic and Morphological Analysis of the Latvian Language. In *Proceedings of the Forth conference on Computational Lexicography and Text Research*, pages 219–225, Budapest

Skadiņa I. (2004). Latviešu valodas morfoloģiskās analīzes sistēma – tās nozīme teikuma pareizrakstības pārbaudē. In *Vārds un tā pētīšanas aspekti 8*, pages 282–290, Liepāja.

Soida, E. and Kļaviņa, S. (1970). *Latviešu valodas inversā vārdnīca*, Rīga, LVU.

Toutanova K., Klein D., Manning C.D. and Singer Y. (2003). Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003*, pages 252–259.

Vasiļjevs, A., Ķikāne, J. and Skadiņš, R. (2004). Development of HLT for Baltic languages in widely used applications. In *Proceedings of First Baltic Conference „Human Language Technologies – the Baltic Perspective"*, pages 198-202, Riga.

Yuret, D. and Türe F. (2006). Learning morphological disambiguation rules for Turkish. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL '06),* pages 328-334, Association for Computational Linguistics, Stroudsburg, PA, USA.

PUBLIKĀCIJA VIII

Coreference resolution for Latvian

Proceedings of LREC 2014, Ninth International Conference on Language Resources and Evaluation, 2014.

Coreference Resolution for Latvian

Artūrs Znotiņš, Pēteris Paikens
University of Latvia, Institute of Mathematics and Computer Science

***Abstract*— Coreference resolution (CR) is a current problem in natural language processing (NLP) research and it is a key task in applications such as question answering, text summarization and information extraction for which text understanding is of crucial importance. We describe an implementation of coreference resolution tools for Latvian language, developed as a part of a tool chain for newswire text analysis but usable also as a separate, publicly available module. LVCoref is a rule based CR system that uses entity centric model that encourages the sharing of information across all mentions that point to the same real-world entity.**

The system is developed to provide starting ground for further experiments and generate a reference baseline to be compared with more advanced rule-based and machine learning based future coreference resolvers. It now reaches 63.9% F-score for end-to-end system and 75.8% for coreference module.

This paper describes current efforts to create CR system and to improve NER performance for Latvian. Task also includes creation of the corpus of manually annotated coreference relations.

***Index Terms*—Coreference resolution, entity centric model, named entity recognition, natural language processing**

I. Introduction

Coreference resolution is the task of grouping all the mentions of entities[1] in a document into coreference chains[2] so that all the mentions in a given chain refer to the same discourse entity [1]. For example, given text (where mention borders are marked with square brackets)

> *[Latvietis$_1$] [Jānis Bērziņš$_1$] ir [jauns zinātnieks$_1$] un [universitātes profesors$_1$]. [Profesors$_1$] ir veicis nozīmīgus pētījumus datorlingvistikā kopā ar [profesoru$_2$] [Pēteri Kalniņu$_2$]. [Viņš$_1$] kopā ar [savu$_1$] [līdzgaitnieku$_2$] [Kalniņu$_2$] uzstāsies konferencē Itālijā.*

> *[Latvian$_1$] [Jānis Bērziņš$_1$] is a [new scientist$_1$] and [professor at university$_1$]. The [professor$_1$], together with [professor$_2$] [Pēteris Kalniņš$_2$], have carried out important research in computer linguistics. [He$_1$], together with [his$_1$] [associate$_2$] [Kalniņš$_2$], will speak in the conference in Italy.*

the task is to group the mentions so that those referring to the same entity are placed together into a coreference chain (represented with same subscripted index).

Latvian is an under-resourced language, with a limited range of language processing tools and resources, and very limited earlier research on coreference resolution [2]. We believe that the described system is the first available implementation of coreference resolution for Latvian language.

Nowadays most coreference systems use knowledge rich features that require extra preprocessing. Typically coreference resolution requires following steps:

- identification of tokens and sentences;
- part of speech tagging;
- parsing;
- named entity recognition;
- mention identification;
- coreference resolution.

While today most state-of-the-art coreference resolvers use machine learning [3], many coreference relations can be resolved using relatively simple rules and recent work has shown that rule based approach can outperform machine learning models for coreference resolution [4], [5]. In this paper we have investigated these approaches and describe our implementation as adapted to Latvian language. In addition, we describe the changes to existing named entity recognition solutions aimed at better coreference resolution accuracy.

II. Proposed Solution

A. Entity Centric Model

LVCoref uses an entity centric model which allows each coreference decision to be globally informed by previously created coreference chains. It allows to use global constraints, e.g., linking two mentions is not allowed if it creates attribute disagreement ("Jānis Bērziņš" and "Pēteris Bērziņš" linked together by a common surname). It also diminishes distance between potential coreferent mentions if the closest same entity mention cannot be correctly linked with current mention based on local features.

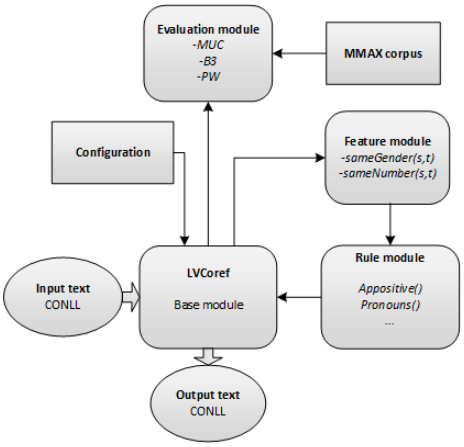


Fig. 1. The architecture of the proposed coreference system

[1] Entity is an object or group of objects, while mention is the reference to an entity.

[2] Coreference chain is a set of coreferent mentions and it corresponds to one entity.

B. System Description

LVCoref base module integrates the other modules described here, and handles input/output formatting and used rules according to a configuration file. An evaluation module uses MMAX [6] format gold coreference links.

The rule module contains available rule sets. These rules are created by combining features from the feature module.

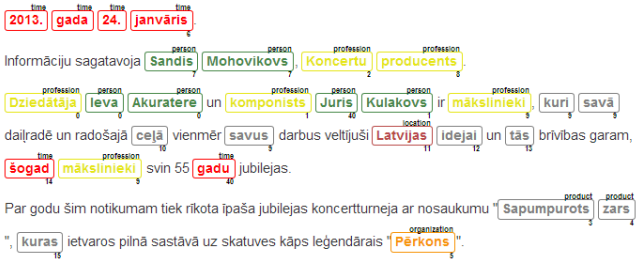


Fig. 2. Automatic coreference annotation with LVCref

C. Preprocessing

The coreference resolution ystem relies on morphosyntactic information produced by the following tools:

The initial step is a statistical morphology tagger which achieves 97.9% accuracy for part of speech recognition and 93.6% for the full morphological feature tag set [7], [8].

Syntactic parsing is done by a parser [9] based on *Maltparser* toolkit [10] and the hybrid dependency-based annotation model used in the Latvian Treebank [11]. Parser is based on dependency grammar approach achieving 72% precision.

In addition, we identify mentions of named entities with a CRF-based NER tool trained for Latvian that provides annotation of person names, geographic locations and organizations, media types, product names currently reaches 76% F-score.

D. Annotated Corpus

Data set was created by manually annotating 6 interviews. The evaluation data was encoded in MMAX format and featured 3 layers: the segmentation layer, the markable layer and the coreference layer.

Corpus was created by annotating 7 mention categories (person, location, media, organization, product, sum and time).

E. Named Entity Recognition

Before this project, there were two available NER systems for Latvian: TildeNER [12] and LVTagger [13] both based upon the Stanford NER condition random field (CRF) classifier. For the purposes of this research we chose to adapt LVTagger, extending it with additional training data for modern news language.

Our chosen taxonomy consists of 7 types of NE (person, location, organization, product, media, sum and time). Nested expression are not tagged as separate NE's, taking in account the longest NE. E.g., whole phrase "*Latvijas Republikas Finanšu un Veselības ministrijas*" (organization) is marked as one entity without marking "*Latvijas Republikas*" (organization) as another entity.

Created corpus (45000 words, 2500 sentences) consists of manually annotated news articles (table I). The corpus can be considered rather small when compared to ConLL corpora which have over 300'000 tokens [14]. While CoNLL corpora use 4 NE types, LVTagger uses 7 types, which makes the data much sparser and therefore the NER task harder.

TABLE I
NAMED ENTITY ANNOTATED NEWS ARTICLE CORPUS

Entity type	Count
location	910
media	63
organization	851
person	512
product	99
sum	245
time	301

The standard CoNLL metric is used, where the output NE is considered correct only if its span and type is exactly the same as the span and type in the gold data.

We have improved gazetteer features for multiword expressions and we are planning new experiments for using semantic database of NE's and frames which is constantly augmented with data from processed news articles to automatically extract high quality gazetteers.

Table II lists current results for NER.

TABLE II
NER EVALUATION RESULTS

Entity type	P	R	F1
location	77,6	85,9	81,5
media	87,9	67,4	76,3
organization	71,8	59,9	65,3
person	88,5	88,8	88,6
product	38,9	8,9	14,4
sum	88,8	89,3	89,1
time	84,2	68,5	75,6
totals	78,9	73,6	76,2

F. Identification of entity mentions

To resolve coreferences, one must first detect the mentions that are going to be linked in coreference chains. Mention identification finds pronouns, common nouns, and named mentions. In general, I take into account noun phrases that are largest possible for their head word, e.g., in phrase "*Kultūras ministrija*" *("the Ministry of Culture"*) only the whole phrase "*Kultūras ministrija*" is marked as a mention and not "*ministrija*". Mentions can be nested, e.g., "[[*Latvijas Nacionālā teātra] direktors*]" *("the [director of the [Latvian National Theatre]]"*).

As mentions are marked all recognized named entities and noun phrases with head word that are listed in gazetteers and acronyms that were not found by NER.

After that non-mentions, e.g., pleonastic "*tas*" ("*it*") in phrases like "*tas nozīmē*" ("*it means*"), are filtered out.

G. Coreference Module

The method is based on applying rules one at a time from the highest to lowest priority, thus in deciding whether two mentions should corefer, system can also consider information about other mentions that previous steps already joined.

1) Exact string match

This rule links two named entity mentions only if they contain exactly the same text by comparing lemmatized phrases.

2) Precise constructions

This rule set links two mentions if any of the conditions below is satisfied:

- *Appositive*. Standard Haghigi and Klein[5] definition to detect appositives is used: one mention is dependent on another, e.g., "[*profesors*$_1$] [*Jānis Bērziņš*$_1$]" *("[professor*$_1$*] [Jānis Bērziņš*$_1$*]"*) .
- *Predicative nominative* are in a subject-object relation being dependent on same verb "*būt*" ("*be"*), e.g., "[*Jānis Bērziņš*$_1$] *ir* [*pasniedzējs*$_1$]" *("[Jānis Bērziņš*$_1$*] is a [professor*$_1$*]"*).
- *Acronym* – mentions are linked if one of them equals the sequence of upper case characters in the other mention, e.g., "*Ekonomikas ministrija*" and "*EM*".

3) Strict head match

Two mentions are linked based on naïve matching of their head words and the second one does not introduce new entity attributes, e.g., "*Latvijas Republikas Augstākā tiesa*" ("*the Supreme Court of the Republic of Latvia*") and "*the Supreme Court of Latvia*".

4) Pronoun anaphora

Pronoun antecedents are searched in three previous sentences using Hobbs' algorithm [15].

Mention compatibility is based on the information about their represented coreference chain. Two mentions are acknowledged as coreferent based on their morphological features (gender, number, case), syntactic constraints (one does not dominate another, *i-within-i* [5]), semantic category and their represented mention chains shared attributes.

III. Results and Evaluation

A. Data Set

Evaluation data came from created coreference corpus using four annotated interviews. Data statistics are listed in table III.

B. Baseline

As a baseline was chosen simple head match, linking all mentions with same head. More sophisticated resolution models have been suggested, but they are rarely compared with this baseline, admitting that it performs better than expected. For the MUC-7 test data Soon's system [16] outperforms head match only by 5%, while Uryupina's system [17] outperforms baseline by 15%.

TABLE III
Data Set Statistics

Number of documents	6
Number of sentences	778
Number of words	13768
Number of mentions	1088
Number of coreference chains	333
Number of singleton mentions	180
The average length of the coreference chain	3.27

C. Evaluation

Coreference module and end-to-end system were evaluated against three well-known coreference resolution metrics: pairwise, MUC [18] and B^3 [19].

MUC is a link based metric which measures how many predicted and gold mention chains need to be merged to cover gold and predicted clusters respectively.

B^3 is a mention based metric which measures the proportion of overlap between predicted and gold mention clusters for a given mention.

The output has the results for each of the three metrics mentioned earlier, both in terms of precision and recall, as well as F-score.

Tables IV and V list the performance of coreference module (with given gold mentions) and end-to-end system. Coreference module outperforms baseline by 7.4%, but end-to-end system by 5.4%.

TABLE IV
Evaluation results for Baseline

	Coreference module			End-to-end system		
	F1	P	R	F1	P	R
MUC	71.9	86.9	61.3	54.7	58.0	51.7
B3	71.8	89.6	59.9	69.1	75.2	63.8
PW	61.5	86.8	47.6	51.6	63.0	43.8
$\overline{\text{MUC. B}^3\text{. PW}}$	68.4	87.8	56.3	58.5	65.4	53.1

TABLE V
Evaluation results for System

	Coreference module			End-to-end system		
	F1	P	R	F1	P	R
MUC	83.9	88.8	79.5	63.8	67.1	60.8
B3	77.1	87.7	68.7	71.2	76.5	66.6
PW	66.5	86.8	53.8	56.7	69.9	47.7
$\overline{\text{MUC. B}^3\text{. PW}}$	75.8	87.8	67.3	63.9	71.2	58.4

Tables VI and VII illustrate the performance of my system as 3 rule sets are incrementally added. Each successive rule set increases system performance by increasing recall and slightly decreasing precision. With respect to individual contributions, this analysis highlights two significant performance increases: exact string match and strict head match. It proves that a large

TABLE VI
Cumulative Performance as Rule Sets are Added to the End-to-end System

	MUC			B^3			PW			$\overline{\text{MUC, B}^3\text{, PW}}$		
	F1	P	R	F1	P	R	F1	P	R	F1	P	R
Exact match	47.1	88.6	32.1	61.2	96.6	44.8	43.0	97.6	27.6	50.4	94.3	34.8
+ Precise construction	49.9	87.7	34.8	62.7	95.9	46.5	43.5	96.9	28.1	52.0	93.5	36.5
+ Strict head match	61.0	70.2	53.9	70.2	81.3	61.8	56.0	78.5	43.5	62.4	76.7	53.1
+ Pronouns	63.8	67.1	60.8	71.2	76.5	66.6	56.7	69.9	47.7	63.9	71.2	58.4

TABLE VII
Cumulative Performance as Rule Sets are Added to the Coreference Module

	MUC			B^3			PW			$\overline{\text{MUC, B}^3\text{, PW}}$		
	F1	P	R	F1	P	R	F1	P	R	F1	P	R
Exact match	54.9	92.7	38.9	63.6	97.2	47.3	47.8	98.5	31.6	55.4	96.1	39.3
+ Precise construction	57.8	91.9	42.1	65.4	96.5	49.5	48.8	96.3	32.7	57.3	94.9	41.4
+ Strict head match	74.4	87.7	64.5	73.7	89.6	62.6	61.7	87.7	47.6	69.9	88.3	58.2
+ Pronouns	83.9	88.8	79.5	77.1	87.7	68.7	66.5	86.8	53.8	75.8	87.8	67.3

percentage of mentions in text are repetitions of previously mentioned entities based on string similarity. Precise constructions give only a slight performance increase because they are relatively infrequent.

D. Error Analysis

To understand the errors in the system, I analyzed two documents from evaluation set and categorized them into distinct groups.

Non-anaphoric constructions. Identifying whether noun phrase is nested mention or part of the stable construction is not a trivial task, e.g., "*Aktieru zāle*" is stable construction and "*Aktieru*" is non-anaphoric construction.

Indefinite noun phrases. Latvian does not explicitly distinguish definite and indefinite nouns, so it is unclear if mention with same head introduces a new entity or refers to a previous mention, e.g., "*Privatizācijas aģentūra*" and "*aģentūra*".

Morphological errors. E.g, singular mentions "*šuvēja*" and "šuvējas" ("tailor") are not linked together because of incorrect grammatical number identification (equal singular genitive and plural nominative forms).

Syntactic errors make it difficult to find appositive and predicative nominative constructions.

Pronoun anaphora resolution. Demonstrative pronoun "*tas*" ("*it*") often refers to event mention, e.g., "*plānot*" ("*to plan*"). This system currently does not mark event mentions, thus missing all mentions that are verbal phrases.

Another considerable source of errors is caused by insufficient semantic information, e.g., pronoun "*mēs*" ("*we*") is used to refer to an organization in an interview.

IV. Conclusions, Application and Further Work

The presented approach offers a useful yet easy to implement baseline for further work and is currently the only available coreference resolution system for Latvian. The implementation is currently used as a part of a newswire text analysis and fact extraction system being developed. We also plan to make an evaluation of the impact of coreference resolution precision on the precision of final fact extraction by the end of this year.

The currently achieved precision – 63.9% for the end-to-end system and 75.8% for coreference module – was satisfactory for use in our text analysis problem and is comparable with results recently achieved for linguistically similar languages ([20], [21] [22]), although their research shows options for future work in improving accuracy. Morphological, syntactic, semantic information and entity centric model provide a noticeable contribution to coreference resolution performance.

Precision of mention identification is one the most important factors that affects the performance of the end-to-end coreference system. Error analysis revealed that the main problems of coreference resolution are related to non-anaphoric constructions, indefinite noun phrases and pronoun coreference resolution.

Currently we are planning first machine learning experiments for coreference resolution and incorporating available semantic database knowledge (facts about popular entities) to support high quality gazetteer maintenance for named entity recognition and to help resolve coreferences using global semantic information.

References

[1] K. van Deemter and R. Kibble, "What is coreference, and what should coreference annotation be?," in *Proceedings of ACL workshop on Coreference and Its Applications, pp. 90-96, Maryland.*, 1999.

[2] G. Barzdins, N. Gruzitis, G. Nespore, B. Saulite, I. Auzina and K. Levane-Petrova, "Multidimensional Ontologies: Integration of Frame Semantics and Ontological Semantics," in *Proceedings of the XIII Euralex Internacional Congress*, Barcelona, 2008.

[3] I. Witten, M. Hall and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, 2011.

[4] H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu and D. Jurafsky, "Stanford's Multi-Pass Sieve Coreference Resolution System at the CoNLL-2011 Shared Task.," 2011.

[5] D. Klein and A. Haghighi, "Simple coreference resolution with rich syntactic and semantic features," 2009.

[6] C. Müller and M. Strube, "Multi-level annotation of linguistic data with MMAX2.," 2006.

[7] P. Paikens, L. Rituma and L. Pretkalniņa, "Morphological analysis with limited resources: Latvian example," 2013.

[8] "Latviešu valodas morfoloģisko pazīmju kopa," [Online]. Available: http://www.semti-kamols.lv/doc_upl/TagSet.pdf. [Accessed 20 02 2013].

[9] L. Rituma and L. Pretkalniņa, "Statistical syntactic parsing for Latvian," 2013.

[10] A. Lavelli, J. Hall, J. Nilsson and J. Nivre, "MaltParser at the EVALITA 2009 Dependency Parsing Task," 2009.

[11] G. Bārzdiņš, N. Gruzītis, G. Nešpore and B. Saulīte, "Dependency-Based Hybrid Model of Syntactic Analysis for the Languages with a Rather Free Word Order.," 2007.

[12] M. Pinnis, "Latvian and Lithuanian Named Entiy Recognition with TildeNER," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, 2012.

[13] P. Paikens, I. Auziņa, G. Garkāje and M. Paegle, "Towards named entity annotation of Latvian National Library corpus," 2012.

[14] T. K. Sang, E.F. and F. De Meulder, "Introduction to the CoNLL-2003 shared task: language-independent named entity recognition," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL*, 2003.

[15] J. R. Hobbs, "Pronoun Resolution," 1976.

[16] W. M. Soon, H. T. Ng and D. C. Y. Lim, "A machine learning approach to coreference resolution of noun phrases," 2001.

[17] O. Uryupina, "Knowledge Acquisition for Coreference Resolution," 2007.

[18] M. Vilain, J. Burger, J. Aberdeen, D. Connolly and L. Hirschman, "A model-theoretic coreference scoring scheme," 1995.

[19] A. Bagga and B. Baldwin, "Algorithms for scoring coreference chains.," in *Proceedings of MUC-7 and LREC Workshop*, 1998.

[20] G. Iakes, A. Olatz, C. Klara, D. d. I. Arantza and J. Amane, "Automatic Coreference Annotation in Basque," in *11th International Workshop on Treebanks and Linguistic Theories*, 2012.

[21] M. Kopeć and M. Ogrodniczuk, "Creating a Coreference Resolution System for Polish," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, 2012.

[22] M. Novák and Z. Žabokrtský, "Resolving Noun Phrase Coreference in Czech," in *8th Discourse Anaphora and Anaphor Resolution Colloquium, DAARC 2011*, 2011.

[23] Recasens, "SemEval-2010 Task 1: Coreference Resolution in Multiple Languages," 2010.

PUBLIKĀCIJA IX

Using C5.0 and exhaustive search for boosting frame-semantic parsing accuracy

Proceedings of LREC 2014, Ninth International Conference on Language Resources and Evaluation, 2014.

Using C5.0 and Exhaustive Search for Boosting Frame-Semantic Parsing Accuracy

Guntis Barzdins, Didzis Gosko, Laura Rituma, Peteris Paikens
Institute of Mathematics and Computer Science, University of Latvia
Rainis Blvd 29, Riga LV-1459, Latvia
E-mail: guntis.barzdins@lumii.lv, didzis.gosko@gmail.com, {laura, peteris}@ailab.lv

Abstract

Frame-semantic parsing is a kind of automatic semantic role labeling performed according to the FrameNet paradigm. The paper reports a novel approach for boosting frame-semantic parsing accuracy through the use of the C5.0 decision tree classifier, a commercial version of the popular C4.5 decision tree classifier, and manual rule enhancement. Additionally, the possibility to replace C5.0 by an exhaustive search based algorithm (nicknamed C6.0) is described, leading to even higher frame-semantic parsing accuracy at the expense of slightly increased training time. The described approach is particularly efficient for languages with small FrameNet annotated corpora as it is for Latvian, which is used for illustration. Frame-semantic parsing accuracy achieved for Latvian through the C6.0 algorithm is on par with the state-of-the-art English frame-semantic parsers. The paper includes also a frame-semantic parsing use-case for extracting structured information from unstructured newswire texts, sometimes referred to as bridging of the semantic gap.

Keywords: FrameNet, semantic role labelling, information extraction

1. Introduction

Development of FrameNet [1] resources for various languages is an ongoing activity (Burchardt at al., 2006; Leenoi at al., 2011). Much of that effort is aimed at only mapping the English FrameNet frames into lexical and syntactic structures of other languages and thus creating a FrameNet annotated corpora for the target language. Meanwhile creation of a Latvian FrameNet was motivated primarily by computational needs of automatic information extraction from natural language texts (predominantly newswire articles). The benchmark methodology for automatic frame-semantic parsing was set at SemEval-2007 (Baker at al., 2007) and specifically - by the best performing LTH system (Johansson & Nugues, 2007). Further improvements to the methodology were implemented in the state-of-art SEMAFOR system (Das at al., 2014).

In this paper we report a novel approach for boosting frame-semantic parsing accuracy through the use of the C5.0 decision tree classifier[2] (Quinlan, 1993) and manual rule enhancement. We also describe a possibility to replace C5.0 by exhaustive search (nicknamed "C6.0") leading to even higher frame-semantic parsing accuracy. This approach is particularly efficient for languages with small FrameNet annotated corpora as is the case for Latvian, which is used in this paper for illustration.

2. Latvian FrameNet

Latvian FrameNet originally was created for a practical information extraction system (described in Section 5) developed for a national news agency to automatically extract biographical data about publicly visible persons and organizations mentioned in the newswire articles. A number of design decisions were taken to strengthen the computational nature of Latvian FrameNet.

First design decision was to preprocess all input texts with a tokenizer and POS tagger (Paikens at al., 2013), an unlabeled [3] dependency parser (Pretkalnina & Rituma, 2013; Pretkalnina et al., 2014), and a NER and co-reference resolver (Znotins & Paikens, 2014) to produce extended CoNLL-style annotations prior to any FrameNet annotation (see Fig.1).

Index	Form (Word)	Lemma	POS	Tag	Parent	Named Entity Type (NER)	Named Entity ID
1	pienākumus	pienākums	n	ncmpa1	3	O	
2	sāks	sākt	v	vmnif13an	3	O	
3	pildīt	pildīt	v	vmnn0t3000n	0	O	
4	pašreizējais	pašreizējs	a	armsnyp	6	O	185
5	Latvijas	Latvija	n	npfsg4	6	location	182
6	vēstnieks	vēstnieks	n	ncmsn1	3	profession	183
7	ASV	ASV	y	y	6	profession	184
8	Ojārs	ojārs	n	n_msn1	9	person	183
9	Kalniņš	kalniņš	n	ncmsn1	7	person	183
10	,	,	z	zc	3	O	

Figure 1: CoNLL style input data for FrameNet tools, a sentence „*Duties began performing current Latvia ambassador to USA Ojars Kalnins."* preprocessed with POS, unlabeled dependency, NER, co-reference parsers

Secondly, a novel FrameNet graphical editor[4] (Fig. 2) was developed (Brediks, 2013) specifically for annotating dependency pre-parsed texts illustrated in Fig 1. The key difference from the legacy phrase-structure grammar based Berkeley FrameNet annotation tool (Ruppenhofer at al., 2010) or the Salto FrameNet annotation tool (Burchardt at al., 2006) is that our tool relies on the dependency-tree to automatically derive filler phrase boundaries once the head-word for the frame element (FE) is selected. This tool was used to create a FrameNet annotated corpus for Latvian. The corpus currently

[1] http://framenet.icsi.berkeley.edu

[2] C5.0 is a commercial version of C4.5 – a decision tree classifier popular for data mining applications, available from http://rulequest.com/see5-info.html

[3] Labeled dependency trees are used in Section 4 to improve the handling of coordination

[4] http://www.ltn.lv/~guntis/FrameMarker.zip

contains almost 5000 sentences from various types of newswire sources.

Third design decision was to use a reduced number of frames – although our methodology is applicable to any number of frames, we have selected just 26 Frames (*Being born, People by age, Death, Personal relationship, Being named, Residence, Education teaching, People by vocation, People by origin, Being employed, Hiring, Employment end, Membership, Change of leadership, Giving, Intentionally create, Participation, Earnings and losses, Public procurement, Possession, Lending, Trial, Attack, Win prize, Statement, Product line*) which were of interest to the national news agency for media monitoring purposes; this use-case dictated also adding or removal of some frame elements (arguments) as shown in Fig. 3.

3. Frame-Semantic Parsing

Thanks to above design decisions it was rather straightforward to adapt the benchmark LTH frame-semantic parser (Johansson & Nugues, 2007) approach to Latvian FrameNet. The original LTH frame-semantic parser uses multiple SVM classifiers to identify frame targets and frame elements. Besides SVM we explored various machine learning approaches, including a log-linear implementation of SEMAFOR [5] system, but the achieved accuracy turned out to be low due to limited size of available FrameNet annotated corpora for Latvian. This problem lead to the key innovation reported in this paper – the C5.0 based manual boosting of frame-semantic annotation accuracy.

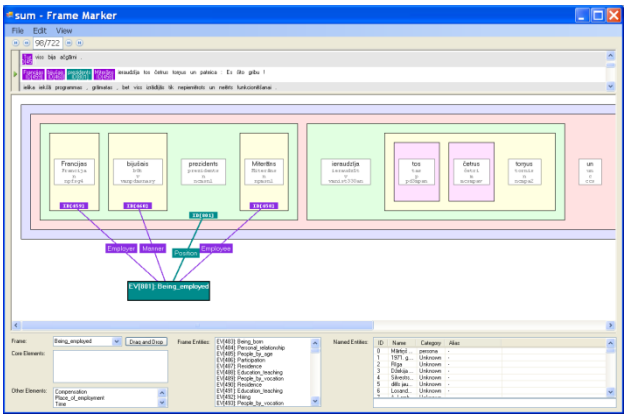


Figure 2: Dependency-tree based FrameNet editor

In terms of classification accuracy C5.0 (C4.5) is comparable to SVM (Shawkat & Smith, 2006) although C5.0 is typically used with lesser training data sets than SVM. Meanwhile the crucial advantage of C5.0 (C4.5) is that the decision tree classifier generated automatically from corpus can be output also in the form of human readable and editable rules like shown below:

```
Rule 1: (5, lift 585.8)
     PreviousLEMMA = euro    (Euro)
     CurrentLEMMA = apgrozījums    (turnover)
     -> class YES (Earnings_and_losses)  [0.857]
Rule 2: (9/1, lift 559.5)
     CurrentLEMMA = peļņa    (profit)
     NextLEMMA = būt    (be)
     -> class YES (Earnings_and_losses)  [0.818]
```

Such classification rules can be easily (effort of approximately 1 hour per frame type) enhanced manually by a human linguist to significantly boost accuracy of frame-semantic parser. Typical rule-changes made by human linguist are adding complete list of month-names, if "January" is mentioned in the rule, or adding more professions, if "plumber" appears in the rule, or discarding some silly rules caused by training data sparsity. Tables 2. and 3. show the actual boosting effect achieved. One can observe that manual boosting results in increased precision (at the expense of slightly reduced recall in case of frame element recognition). It is crucial to note that such manual boosting is quite "cheap" compared to effort required to achieve a similar boost by merely annotating more training data. To achieve simpler classification rules to be read and edited by human, we trained a separate[6] binary (YES/NO) C5.0 classifier for identification of each frame target and frame element type. This is slightly different from the approach taken in LTH frame-semantic parser, which divides the task into the following steps:

1) Identifying the words that should be associated with frames
2) Classifying the frames associated with the word in (1)
3) Identifying the words that should be associated with frame elements (arguments)
4) Classifying the frame elements associated with the words in (3)

In our frame-semantic parser "frame target identification" refers to steps (1) and (2) jointly, as these are handled by one binary C5.0 classifier per frame type, which merely classifies if the current word in the text is (or is not) a target for this specific frame type. Similarly "frame element identification" in our case refers to (3) and (4) jointly and is handled by one binary C5.0 classifier per frame element type.

In our approach positive examples in the resulting training datasets are sparse and require considerable tweaking of C5.0 parameters to produce meaningful rules. We concluded on the following command-line parameter settings:

```
$ ./c5.0 -r -m1 -c100 -f <name>
```

along with associated <name>.costs file heavily penalizing missed YES-rules: "`NO, YES: 100`"

[5] Log-linear or perceptron based approaches have significant drawback (compared to kernelized SVM or C5.0) – besides the list of basic features they require also "feature templates" to handle feature vector value patterns. These feature patterns need to be manually crafted by the domain expert (Das at al., 2014). Use of C4.5 to automate feature template generation (Fernandes & Milidi'u, 2012) was seminal to the approach described in this paper.

[6] Our approach of identifying each frame type separately allows to scale it linearly from 26 frames in Latvian FrameNet to over 1000 frames in the current English FrameNet 1.5

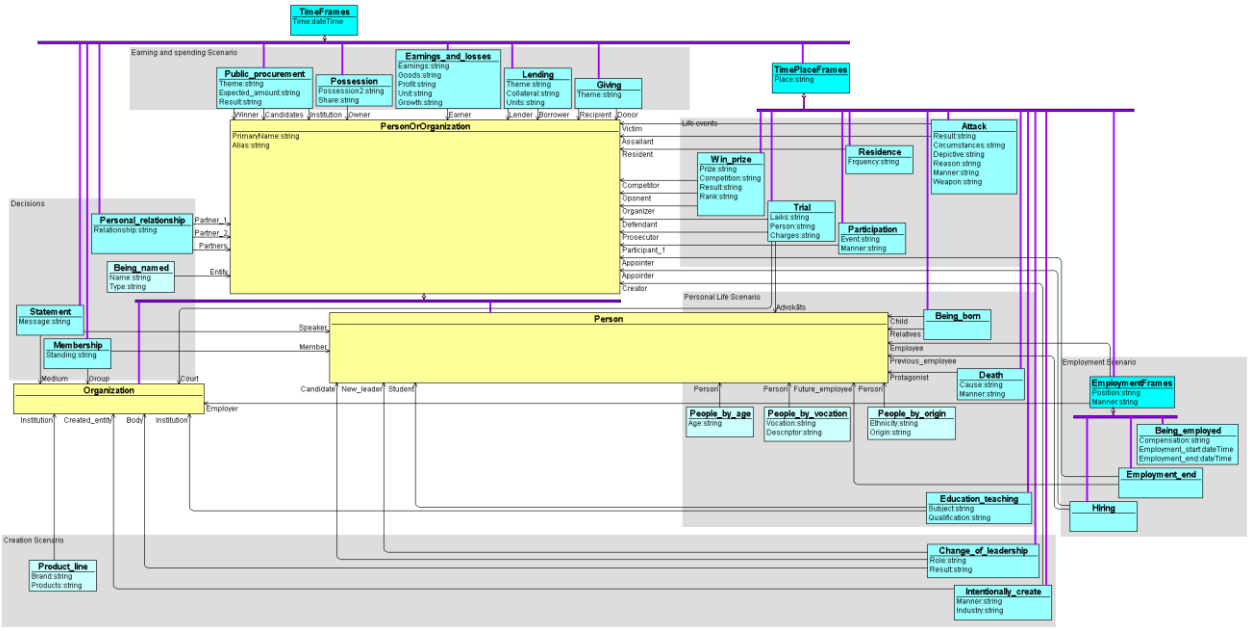


Figure 3: Latvian FrameNet 26 Frames (blue boxes) and frame element filler types / NER categories (yellow).

For fully automatic frame target identification mode rules were cut-off at Laplace ratio 0.1 to avoid target overgeneration due to manipulated costs file. Frame targets are identified first and then frame element candidates are considered only in the radius of 4 words around the identified frame target word according to the dependency tree; only one frame element of a kind is retained for each frame target if C5.0 classifier found multiple candidates (the closest ones to the target according to the dependency tree).

	Latvian FrameNet data	*English SemEval '07 data*
Exemplar sentences	1682	139439
Frame labels (Frame types)	26	665
Role labels (FE types)	80	720
Sentences in test data	199	120

Table 1: FrameNet data sets used for evaluation

Target identification	*Precision*	*Recall*	*F1*
C5.0 fully automatic (Latvian FrameNet data)	51.7	39.5	44.8
C5.0 manual boosting (Latvian FrameNet data)	**55.6**	**43.9**	**49.1**
C6.0 fully automatic (Latvian FrameNet data)	**62.5**	**46.8**	**53.5**
LTH (English SemEval'07 data)	66.2	50.6	57.3
SEMAFOR (English SemEval'07 data)	69.7	54.9	61.4

Table 2: Frame target recognition evaluation results

FE identification	*Precision*	*Recall*	*F1*
C5.0 fully automatic (Latvian FrameNet data)	54.6	43.8	48.6
C5.0 manual boosting (Latvian FrameNet data)	**59.4**	**43.3**	**50.1**
C6.0 fully automatic (Latvian FrameNet data)	**61.3**	**60.7**	**61.0**
LTH (English SemEval'07 data)	51.6	35.4	42.0
SEMAFOR (English SemEval'07 data)	58.1	38.8	46.5

Table 3: Frame element recognition evaluation results

Evaluation of our initial results shows that C5.0 decision tree based approach provides accuracy that is competitive for Frame-semantic parsing, and can also be conveniently combined with manually enhanced rules for accuracy boosting. Comparing Latvian frame-semantic parsing results to state-of-the-art English frame-semantic parser accuracies suggests[7] that C5.0 and smaller size of Latvian FrameNet contributes positively to frame element recognition accuracy, while for frame target recognition corpus size is crucial. It shall be noted that for target identification English frame-semantic parsers actually use two additional information sources not available for Latvian – the list of lexical units known to invoke particular frame (lexical units are part of English FrameNet distribution) and WordNet synsets (Fellbaum, 1998).

4. Exhaustive Search (C6.0)

While experimenting with C5.0 as described in the previous Section, we noted that use of approximate

[7] Evaluation results for English were copied from [4]. Evaluation script used for English is not available online. Our evaluation script counts only exact head-word matches for frame targets and for frame elements in correctly identified frames

entropy-based C5.0 is somewhat obsolete for tasks requiring only binary classifier (e.g. our frame-semantic parser implementation), because the number of hypothetical rules recognizing positive exemplars is merely *number-of-positive-exemplars* × $2^{feature\text{-}count}$, which is a tractable number for exhaustive search up to approximately 20 features (we use 11 features for frame target identification and 13 features for frame element identification). It shall be noted that exhaustive search applies only to the rule learning stage – the runtime application of the learned rules is very fast.

Additional motivation to replace C5.0 was the costs file, which had to be manually tweaked to generate rules from unbalanced training data containing massive amounts of negative exemplars and very sparse positive exemplars. Without costs file C5.0 often gave just single default rule "negative", which is true for 99.9% of training exemplars. Few optimizations allowed cutting down the computation time for exhaustive search below one minute per classifier for the amount of training data available in Latvian FrameNet. The resulting exhaustive search based classifier we nicknamed[8] in this paper "C6.0" since for frame-semantic parsing applications it clearly surpasses the original C5.0 (including also the manually boosted C5.0 rules) – see the initial C6.0 results in Tables 2. and 3. Attempts to further manually boost the rules generated by C6.0 were nearly fruitless and improved accuracy by statistically insignificant values of less than 1%.

```
1 [_, _, _, {peļņa, apgrozījums}, _, _, _, _, _, _, _]     136  31
2 [_, ng, _, zaudējums, _, _, _, _, _, _, _]               10   0
3 [_, _, _, {zaudējums, ienākums}, _, nn, _, _, _, _, _]   12   2
4 [_, _, _, nopelnīt, _, _, _, _, _, x, _]                 6    0
5 [uzņēmums, _, _, _, _, _, _, _, _, vcnpa, _]             2    0
6 [kompānija, _, _, _, _, v_nia, _, _, _, _, _]            2    0
7 [',', _, _, ieņēmums, _, _, _, _, _, _, _]               2    0
```

Figure 4: C6.0 generated target identification rules for frame *Earnings and losses*. Shown are counts in the training corpus for total matches and false positives.

Meanwhile the human-readable, optimal rules generated by C6.0 (it is actually quite insightful to read these machine generated rules, see Fig. 4) opened two other possibilities for boosting the frame-semantic parsing accuracy:

a. Correcting the frame annotation inconsistencies in the training corpus.
b. Spotting the missing features preventing C6.0 from inferring universal rules with high coverage.

Training corpus annotation inconsistencies are particularly easy to spot in the human-readable frame target identification rules generated by C6.0, because these rules substitute for the meaningful lists of lexical units (word senses, included in the English FrameNet distribution) known to invoke the particular frame. Meanwhile frame element identification rules generated by C6.0 correspond to meaningful lexical entries [9] (containing frame element syntactic realization variations in the annotated corpora) in English FrameNet distribution. Tables 4, 5, 6 show the final results after the spotted annotation inconsistencies (mostly they were missed frames) were corrected in the extended training corpus and few missing features were added to the parser. Fig. 5 shows cross-validation of frame target F1 score relative to various split of training and test sets. The evaluation results show that the resulting C6.0 based Latvian frame-semantic parser performs on par with state-of-the-art English frame-semantic parsers despite smaller FrameNet training corpus for Latvian.

	Latvian FrameNet data	*English SemEval '07 data*
Exemplar sentences	4079	139439
Frame labels (Frame types)	26	665
Role labels (FE types)	80	720
Sentences in test data	844	120

Table 4: Extended data sets used for evaluation

Target identification	*Precision*	*Recall*	*F1*
C6.0 fully automatic (Latvian FrameNet data)	**63.5**	**62.7**	**63.1**
LTH (English SemEval'07 data)	66.2	50.6	57.3
SEMAFOR (English SemEval'07 data)	69.7	54.9	61.4

Table 5: Frame target recognition final results

FE identification	*Precision*	*Recall*	*F1*
C6.0 fully automatic (Latvian FrameNet data)	**65.9**	**76.8**	**70.9**
LTH (English SemEval'07 data)	51.6	35.4	42.0
SEMAFOR (English SemEval'07 data)	58.1	38.8	46.5

Table 6: Frame element recognition final results

The final list of features used for frame target identification was:

PLEMMA – previous word lemma
PPOS –previous word morphology tag
PNETYPE – previous word NE type
LEMMA – target word lemma
LEMMA_CLUSTER – target word cluster
POS – target word morphology tag
DEPLABEL – syntax role of the target word
NETYPE – target word NE type

[8] C6.0 is not a universal substitute for the much richer functionality of C5.0 useful in other application domains

[9] Lexical entries in English FrameNet include also valence patterns, defining meaningful frame element subsets and their syntactic realizations observed in the annotated corpora; in our parser meaningful frame element subsets are hardcoded

NLEMMA – next word lemma
NPOS – next word morphology tag
NNETYPE – next word NE type

The final list of features used for frame element identification was:

LEMMA – FE headword lemma
LEMMA_CLUSTER – FE headword lemma cluster
POS – FE headword morphology tag
NETYPE – FE headword NE type
DEPLABEL – syntax role of the FE headword
HLEMMA – parent word lemma
HLEMMA_CLUSTER – parent word cluster
HPOS – parent word morphology tag
HNETYPE – parent word NE type
TARGET_TYPE – frame name
TARGET_PATH2D – sequence of 4-direction moves forming the path in the dependency tree between FE headword and target word
TARGET_PATH2D_SHORT – the path without sequential duplicates
TARGET_NEAR – path length above or below 4

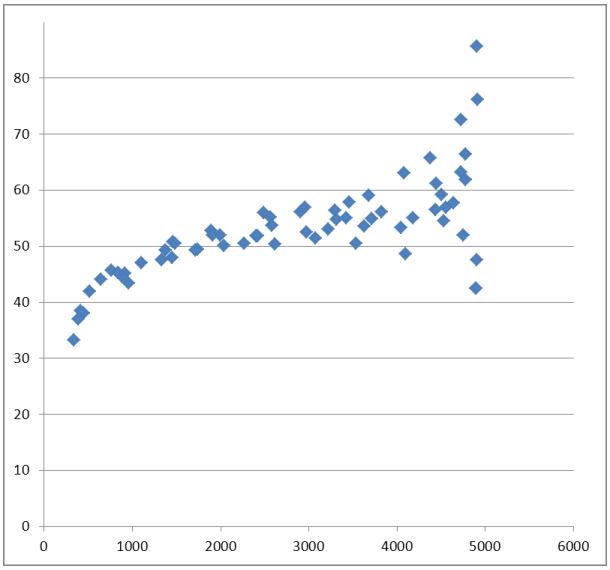


Figure 5: Dynamics of frame target F1 score relative to the number of sentences in the training set versus test set. The total number of annotated sentences is 4923.

The actual implementation of C6.0 algorithm we have developed is slightly more sophisticated than pure relaxation of positive exemplars for exhaustive search of best rules shown in Fig. 4, as algorithm has to decide which of the searched rules form the best rule-set without falling victim to the overfitting/underfitting problem. Overfitting occurs when rules have too high precision at the expense of low recall – such rules perform excellent on the training set, but are not general enough to be useful for unseen data. Underfitting is the opposite extreme, where high recall is achieved at the expense of low precision due to rules being too promiscuous. In C6.0 we use the same approach as C5.0 to address the overfitting/underfitting problem through confidence limits for the binomial distribution or through Laplace ratio. The best F1 scores we achieved with the default Laplace ratio *(n-m+1)/(n+2)* for rule's accuracy estimation, where *n* is the number of exemplars covered by the rule and *m* shows how many of them are false positives (*n* and *m* are the two numbers shown in Fig. 4 for every rule). Meanwhile confidence limits for the binomial distribution gave better recall rates with slight degradation to precision and overall F1 accuracy.

The actual C6.0 implementation [10] includes minor additional fine-tuning options such as tiebreaking strategy for rules with equal Laplace ratio – preferring the most relaxed or the most specific rule (default is choosing the most specific rule) and restricting the maximum number of features appearing in one rule (default is 5, although 3 gives nearly as good results in the fraction of time). C6.0 also includes sieves to minimize the number of overlapping rules and to keep only rules covering more than one exemplar, as fewer rules in the resulting rule-set tend to improve the overall accuracy on unseen data.

5. Discussion

The ability to achieve high accuracy for frame-semantic parsing enables streamlining of information extraction task from natural language texts, such as newswire articles. The goal of such information extraction effectively is populating the ontology[11] shown in Fig.3 (this is OWLGrEd [12] visualization of the actual OWL ontology) with instance data retrieved from the text. To do so, frame-semantic parsing techniques described in this paper (producing instances for the blue boxes in Fig.3) need to be combined with Cross Document Coreference (CDC) techniques (Wick at al., 2013) to automatically determine which mentions in the text refer to the same real-world entity (producing disambiguated instances for the yellow boxes in Fig.3).

We have implemented such integrated information extraction system and populated it with data from approximately 1 million newswire articles. From the practical standpoint it turned out that the bottleneck of the approach is Named Entity discovery and linking accuracy – even at estimated 80% CDC accuracy it too often merged together different real-world entities with similar names or did not link together alternative spellings for the same entity (due to frame elements often being a hierarchy of Named Entities, e.g. "*triju Zvaigžņu ordeņa virsnieks*" in Fig. 6), making the overall results unusable. To mitigate the problem, we deflected to the use of the predefined Knowledge Base of manually disambiguated well-known person, organization, location, product, event names (with their commonly used aliases), which can be identified in the text more robustly using Named Entity linking methods similar to DBpedia Spotlight (Daiber at al., 2013). Of course, this workaround links only frame elements found in the predefined Knowledge Base, leaving other frame element fillers unidentified. The

[10] http://c60.ailab.lv
[11] http://www.ltn.lv/~guntis/FrameNetLV.owl
[12] http://owlgred.lumii.lv

unidentified frame element fillers therefore are stored as simple text strings as they appear in the original sentences (technically they can be stored in the same Knowledge Base, only tagged as "unidentified entities").
From the practical standpoint of information extraction about persons and organizations from the newswire texts this has turned out to be the best solution – link only entities present in the Knowledge Base, but leave all other frame element fillers identified only by the text strings as they appear in the source text. This mixed approach allows for creating a convenient user interface, where instance data from the Knowledge Base in Fig. 3 is verbalized using a light version of (Dannells & Gruzitis, 2014) producing simple sentences as illustrated in Fig. 6 which can further be formatted in the familiar Curriculum Vitae like manner.

Ieva Akuratere bija solista amatā [23]
Ieva Akuratere bija Puķu burves amatā [8]
Ieva Akuratere bija mūziķes un aktrises amatā [5]
Ieva Akuratere bija deputātes amatā Rīgas domē [4]
Ieva Akuratere bija solista amatā Koncertuzvedumā [4]
Ieva Akuratere bija dziedātājas amatā [3]
Ieva Akuratere bija triju Zvaigžņu ordeņa virsnieka amatā Latvijā [3]

Figure 6: Fragment of the automatically generated person profile (verbalization of *Being employed* frame). Linked Named Entities underlined, duplicate counts in brackets.

Although not yet implemented in a practical system, there is a further refinement possible for the above described Knowledge Base and information extraction system – adding the time dimension (in Fig. 3 note that *Time* is the dominant frame element present in almost all frames). For most frames extracted from the newswire texts the time of their occurrence is either explicitly specified in the text and can be retrieved by frame-semantic parser as frame element *Time* or approximate time can be retrieved from the metadata of the newswire article publication date.
Having time associated with all extracted frames opens a possibility (Barzdins, 2011) for structuring the information extracted from the newswire texts – rather than having a mix of seemingly contradictory facts in one Knowledge Base (e.g. *"Peter lives in Paris"* and *"Peter lives in NewYork"*) we can create a whole sequence of Knowledge Base instances (one per every day of history), with each instance containing only the facts which were true on that particular day and thus make these instances non-contradictory (e.g. *"Peter lives in Paris" (in instances for 2001)* and *"Peter lives in New York" (in in instances for 2011)*). Inserting frames extracted from the text by the frame-semantic parser into the proper instance (or sequence of instances) of the Knowledge Base is not an easy task (Murray & Singliar, 2012), as some frames describe an instantaneous event (e.g. frame *Attack*) while other frames describe a state which is true over prolonged period of time (e.g. frame *Being employed*). Nevertheless, resolving the time dimension (and for some sorts of tasks – also spatial dimension) is a vital additional tool for truly bridging the semantic gap in natural language understanding, eventually enabled by the accurate frame-semantic parsing.

Being born	100	Residence	67	Participation	40
Earnings and losses	89	Statement	67	Employment end	33
Death	80	Hiring	62	Product line	33
Education teaching	71	Membership	50	Lending	29
Being employed	70	Possession	48	Personal relationship	25
Change of leadership	67	People by vocation	46	Trial	18
Intentionally create	67	Win prize	45	People by origin	16

Table 7: Target identification F1 scores for some Latvian FrameNet frames.

To evaluate to what extent the information extraction approach described in this paper actually bridges the semantic gap (Ehrig, 2007) between the unstructured newswire input text and the structured output (Knowledge Base or ontology in Fig. 3), Table 7 breaks down the target identification accuracy for various frames. These results illustrate that target identification accuracy varies widely between different frame types, meaning that the current set of features apparently is not sufficient for identification of the low-scoring frames. Another explanation for the low-scoring frames might be that the concept they convey is broader (can be expressed in more ways) and thus bridging of the semantic gap with high accuracy for these frames requires a larger training corpus.

6. Conclusion

The described approach illustrates the possibility of bootstrapping a state-of-the-art frame-semantic parser for a new language by merely hand-annotating approximately 5000 sentences with the frames of interest. In our approach each frame is learned independently, meaning that the result holds for any number of different frames. It is interesting to observe that rules for frame target and frame element identification generated automatically by C6.0 effectively substitute for the manually crafted lexical unit entries which are part of the English FrameNet distribution.
On a more philosophical level, we believe that our C5.0/C6.0 based approach of statistical learning of human readable (and human-editable) rules from a corpus bridges the gap between statistical and rule-based NLP approaches and likely can be extended to other NLP areas such as the MaltParser shift-reduce dependency parsing algorithm (Nivre, et al., 2007), where the SVM classifier could be replaced by C5.0 or C6.0 to achieve a similar manual accuracy boosting effect.
Another notable achievement of C6.0 is practical machine learning by exhaustive search, which is shown to achieve high accuracy even from a small set of exemplars as shown in Fig. 5. We suspect that C6.0 is more accurate than approximate machine learning techniques popular today, but a thorough comparison with other machine learning approaches is beyond the scope of this paper.

7. Acknowledgement

The research was partially supported by Latvian 2010.-2014. National Research Program Nr.2 "Development of Innovative Multifunctional Materials, Signal Processing and Information Technologies for Competitive Science Intensive Products", project Nr.5. We also thank the anonymous reviewers for their improvement recommendations.

8. References

Baker, C., Ellsworth, M., Erk, K. (2007). SemEval-2007 task 19: Frame semantic structure extraction. In *Proceedings of SemEval-2007: 4th International Workshop on Semantic Evaluations*. Prague, pp. 99–104.

Barzdins, G. (2011). When FrameNet meets a Controlled Natural Language. In *Proceedings of NODALIDA*. Riga, NEALT Proceedings Series Vol. 11, pp. 2--5.

Brediks, D. (2013). FrameNet Semantic Annotation Editor with Co-reference Identification, Postgraduate Thesis in Informatics, University of Latvia.

Burchardt, A., Erk, K., Frank, A., Kowalski, A., Pado, S., Pinkal, M. (2006). The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*. Genoa, Italy, p. 6.

Daiber, J., Jakob, M., Hokamp, C., Mendes, P.N. (2013). Improving efficiency and accuracy in multilingual entity extraction, In *Proceedings of the 9th International Conference on Semantic Systems*. ACM, pp. 121--124.

Dannells, D., Gruzitis, N. (2014). Extracting a bilingual semantic grammar from FrameNet-annotated corpora. In *Proceedings of the 9th Language Resources and Evaluation Conference (LREC)*. Reykjavik, *this volume.*

Das, D., Chen, D., Martins, A.F.T, Schneider, N., Smith, N.A. (2014). Frame-Semantic Parsing, *Computational Linguistics*, 40(1), pp. 9--56.

Ehrig, M. (2007). Ontology Alignment – Bridging the Semantic Gap. *Semantic Web and Beyond*, Vol. 4, Springer.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database.* MIT Press.

Fernandes, E.R., Milidi´u, R.L. (2012). Entropy guided feature generation for structured learning of Portuguese dependency parsing. In *Proceedings of the Conference on Computational Processing of the Portuguese Language (PROPOR)*. Lecture Notes in Computer Science, Vol. 7243, pp. 146--156.

Johansson, R., Nugues, P. (2007). LTH: semantic structure extraction using nonprojective dependency trees. In *Proceedings of SemEval-2007: 4th International Workshop on Semantic Evaluations*. Prague, pp. 227--230.

Leenoi, D., Jumpathong, S., Porkaew, P., Supnithi, T. (2011). Thai FrameNet Construction and Tools, *International Journal on Asian Language Processing*, 21(2), pp. 71--82.

Murray, W., Singliar, T. (2012). Spatiotemporal Extensions to a Controlled Natural Language. In Proceedings of the 3rd Workshop on Controlled Natural Language, volume 7427 of LNCS, pp. 61-78.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kubler, S., Marinov, S., Marsi, E. (2007). MaltParser: A languageindependent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Paikens, P., Rituma, L., Pretkalniņa, L. (2013). Morphological analysis with limited resources: Latvian example. In *Proceedings of NODALIDA*. Oslo, pp. 267--278.

Pretkalnina, L., Znotins, A., Rituma, L., Gosko, D. (2014). Dependency parsing representation effects on the accuracy of semantic applications — an example of an inflective language. In *Proceedings of the 9th Language Resources and Evaluation Conference (LREC)*. Reykjavik, *this volume.*

Pretkalnina, L., Rituma, L. (2013). Statistical syntactic parsing for Latvian. In *Proceedings of NODALIDA*. Oslo, pp. 279—290.

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers.

Ruppenhofer, J., Ellsworth, M., Petruck, M.R.L., Johnson, C.R., Scheffczyk, J. (2010). *FrameNet II: Extended Theory and Practice.* Berkeley, CA, USA: International Computer Science Institute.

Shawkat, A., Smith, K.A. (2006). On learning algorithm selection for classification. *Applied Soft Computing*, 6(2), pp. 119--138.

Wick, M., Singh, S., Pandya, H., McCallum, A. (2013). A Joint Model for Discovering and Linking Entities, In *Proceedings of the 2013 workshop on Automated knowledge base construction*. ACM, pp. 67--72.

Znotins, A., Paikens, P. (2014). Coreference Resolution for Latvian. In *Proceedings of the 9th Language Resources and Evaluation Conference (LREC)*. Reykjavik, *this volume.*

PUBLIKĀCIJA X

Latvian newswire information extraction system and entity knowledge base

Proceedings of Human Language Technologies – the Baltic Perspective, 2014.

Latvian Newswire Information Extraction System and Entity Knowledge Base

Pēteris PAIKENS[a,1]
[a] *University of Latvia, Institute of Mathematics and Computer Science*

Abstract. This paper describes an information extraction system designed for obtaining CV-style structured information about publicly mentioned persons, organizations and their relations by analyzing newswire archives in the Latvian language. The described text analysis pipeline consists of morphosyntactic analysis, NER and coreference resolution, and a semantic role labeling system based on FrameNet principles. We also implement an entity linking process, matching the entity mentions in each document to an entity knowledge base that is initially seeded with authoritative information on relevant people and organizations. The accuracy of automated frame extraction varies depending on specifics of each frame type, but the average accuracy currently is 53% F-score for frame target identification, and 61% for frame element role classification. The currently targeted volume of text is the total archives of Latvian newspapers, magazines and news portals, consisting of about 3.5 million articles.

Keywords. information extraction, knowledge base, text summarization

Introduction

Newswire archives contain a huge wealth of information that has been once gathered, verified and published, but is scattered among many separate articles of unstructured natural language text. There is a large demand for extracting this knowledge in a structured and summarized manner, and this is also an important business area for a number of news broker companies. A particular niche of structured information is the profiles of important people and companies. For some languages and locations, the need is well served by open resources such as Wikipedia, but for others, including Latvia, this coverage is not sufficient and there is a market need for providing such data. This is currently done by LETA, the largest Latvian news agency, providing profiles of some 20,000 people and 2,000 organizations. The raw data of news articles is digitized and well accessible, and current technology supports effective search and retrieval of relevant documents, but creating and maintaining profile data still is very labor intensive and requires a significant time investment. This time cost limits the coverage of such profiles to a fraction of all people mentioned in news, and restricts the frequency of reviewing and updating those profiles.

As this type of information can be automatically extracted from article text by state of art information retrieval approaches, a research project was started by University of Latvia IMCS together with LETA, the largest Latvian news agency, with the goal of

[1] Corresponding Author: Pēteris Paikens, University of Latvia, Institute of Mathematics and Computer Science, Raiņa bulvāris 29, Riga, Latvia, LV-1459; E-mail: peteris@ailab.lv.

researching these methodologies, adapting them to the Latvian language and target domain, and building a prototype for a pilot project of extracting profile data about publicly mentioned persons and organizations, as well as their relations, from newswire archives in Latvian.

The described text analysis system is designed to provide news analysts with 'fact candidates' about such entities, linking to the primary sources of those facts for clarification – if this can be done with a sufficient accuracy, it allows to summarize larger amounts of data than is manageable by people using common search techniques. The structured data of relations between people and organizations can also be used for journalist analysis of indirect relations, when represented as a graph in tools that allow to visualize and explore such data. The currently targeted volume of text is the total archives of Latvian newspapers, magazines and news portals, consisting of about 3.5 million articles.

In the first section, we describe the main research problems encountered, and the relevant previous research on solving those problems. Section 2 describes the conceptual and technical architecture of the developed information extraction system. Section 3 provides details on the representation chosen to model the relevant domain facts. Section 4 describes the process of linking entity mentions discovered in text to the appropriate real world entities. Finally, we provide some conclusions and discussion of future work.

1. Problem Description and Related Work

Information extraction is a currently unsolved problem in computational linguistics, and an active area of research. While some of the required components are well-researched, many of them still require improvements to be suitable for practical usage even for well-resourced languages such as English. The main active research issues are the actual semantic data extraction phase, the abstract meaning representation model, and entity linking to the appropriate real world entities.

Implementing information extraction for the Latvian language added extra challenges in developing or adapting tools for the more general text processing stages. The morphosyntactic analysis and named entity recognition parts of this system are a separate problem that is described shortly in the next section, and with more detail in the cited publications.

1.1. Newswire Information Extraction Task

There are recently started projects for other languages with similar aims – the closest such project is NEWSREADER[1] that uses a similar methodology for aggregating notable newswire events, with their current analysis focus on the financial domain of public companies. While that research is still ongoing and was not published before our system development was well underway, their approach is very relevant and offers solutions to potential subtasks, e.g. for scaling the processing[2] if we would want to analyze the corpora in real time or move to larger corpora than only Latvian newswire.

In addition, there are multiple projects that also attempt information extraction from newswire corpora, most notably Europe Media Monitor[3], but they target a different problem scope and the main overlap with this paper is in entity identification.

1.2. Meaning Representation Model

As we need to represent the domain information in a structured manner, the choice of meaning representation determines both the scope of facts that the system will be able to describe, as well as the level of detail and nuance that it will attempt to capture from text. The classic approaches for modeling this data include relational databases and the linked data approach using Resource Description Framework models.

For the purposes of this system, we have chosen to model the domain knowledge according to FrameNet principles[4], as described in section 3. The main reason for this choice was that it is closer to the fact representation as it occurs in natural language sentences; and the advantages of other approaches can be obtained by further automated transformations of this data to RDF or specific database formats.

A notable new relevant approach has been recently published – Abstract Meaning Representation[5], which would potentially be valuable for this use case, but currently still needs more research and tool development for automated text analysis to this representation.

1.3. Semantic Role Labeling

The key component of the text analysis system is the step of mapping the identified text morphosyntactic structure and entities to the semantic representation. We treat the core part of profile data extraction as a semantic role labeling problem, annotating sentence tokens with the frame target and frame element information according to the chosen representation.

The current state of art systems for performing this step, as measured on corpus of Semeval2007 shared task, are LTH[6] and Semafor[7]. Those algorithms are of general purpose and can be adapted to a variety of languages, annotation paradigms and text domains, and were also tested in practice on Latvian data. During our research, we developed a novel, separately described method[8] based on decision tree learning that achieves a comparable accuracy, and also gives a possibility for manual rule review and improvement that is well suited for the properties of this project –preexisting domain knowledge and small number of frame types that makes manual rule review feasible.

There is also significant research on extracting such data by fixed sentence patterns and regular expressions, which we did not consider in depth as such approaches have limited coverage in languages with variable word order such as Latvian.

1.4. Entity Linking

The relevant sub problem of entity linking is the task of matching named entities identified in documents to their real-world counterparts, identifying new entities and resolving ambiguities for multiple people with the same name. A related problem also is cross-document coreference resolution and event coreference linking, which is not currently handled but is a topic for future work

Current related research on entity linking includes Wick et al[9], Han et al[10] and Stoyanov et al[11], based on which we developed an entity linking module tuned for the needs of this project as described in section 4.

2. System Architecture

The main parts of the system are its text analysis pipeline and the entity knowledge base. The text analysis pipeline consists of morphosyntactic analysis[12], [13], named entity and coreference identification[14], and a semantic role labeling system[8]. The latter two components were implemented for the Latvian language specifically for this project, and the morphosyntactic and NER layers were adapted for newswire text domain by creating additional training data and tuning statistical models.

After this document analysis, the entities found in each document are mapped to an entity-based knowledge base, as shown in Figure 1, and appending the newly identified facts. Afterwards, the facts identified in each separate document (often duplicates) are summarized for further applications.

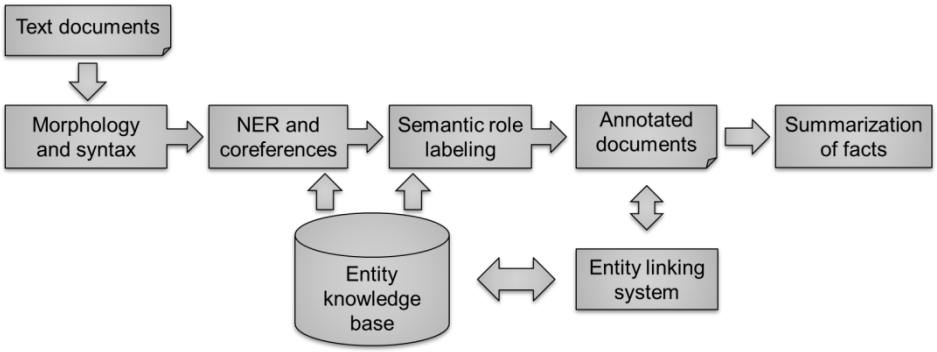


Figure 1. Analysis process flow.

The technical architecture implements each annotation layer as a separate software module capable of running independently and with multiple concurrent copies, suitable for batch processing of large corpora. Data interchange between the modules is done either in columnar tab-delimited format as used in historical CONLL and Semeval shared tasks or a custom JSON format that includes the entity details and semantic frame labeling over the original sentences.

3. Fact Representation and Entity Knowledge Base

For the purpose of analyzing biographical data, we have chosen a narrow subset of English FrameNet – 26 frames – and adapted the frame details for both the Latvian language and targeted domain.

A key challenge was the actual adaptation of semantic frame models. It was not straightforward, as the original FrameNet frames significantly vary in granularity, and domain-specific needs mandated adjustments and additions to the original frames. The currently proposed ontology, shown in Figure 2, stores the identified semantic frames as predicates linking together multiple entities, and allows summarizing/merging multiple frames with identical or overlapping information.

The implementation treats all types of entities as equal, and all information that is particular only to some entities (e.g., people) is stored as separate types of semantic frames. Thus, the entities are reduced to their identities and alternative names, the type and a set of semantic frames that describe this entity and link it to other entities.

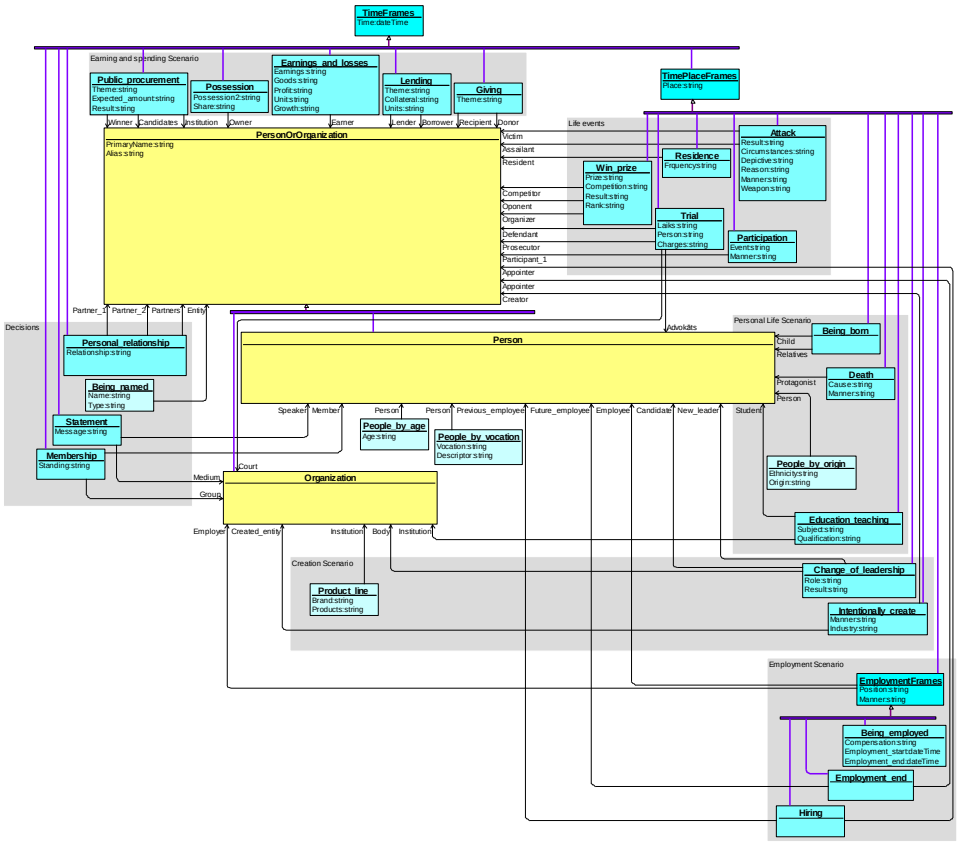


Figure 2. Knowledge representation ontology.

4. Entity Linking

We also implement an entity linking process, matching the entity mentions in each document to an entity knowledge base that is initially seeded with authoritative information on ~25,000 popularly known people and ~35,000 companies, and source data on 'classifiers' such as locations, professions, etc.

Common practice for larger languages such as English is to link entities to the identities listed in large public repositories such as Wikipedia or DBpedia, but for Latvian those resources do not provide a sufficiently high coverage of locally important people and companies. Thus, an internal authoritative list is used, based on data previously aggregated in proprietary systems of LETA. The number of entities rapidly increases by a factor of ten as new, less common entities get added from document analysis – authoritativeness of such entities is expected to be maintained by manually reviewing and correcting newly identified entities with a large number of mentions. The entity knowledge base data is also fed back to the analysis stage in order to improve named entity classification accuracy by using the previously seen entities.

Entity name ambiguity is resolved by a cross-document coreference technique loosely based on the entity linking model used by Wick, Singh et al[9]. It is assumed

that in case of people sharing identical names, the knowledge base would contain a list of those namesakes, and disambiguation can be performed amongst them – and if it is not, then such entities can be flagged for manual review due to conflicting factual data such as different claimed birth years. Three separate signals are used for classification: (a) the components of 'extended names', taking the extended noun phrase parts from document mentions, and appositive items from the known frames such as titles or professions; (b) connected entities – other entities mentioned in the document versus entities sharing a common fact/frame in the knowledge base; and (c) document level context according to a bag of words model, which approximately models document topic –disambiguating an actor and a politician sharing the same name by looking if the document contains words specific to theatre or political reporting.

The similarities between the entity mentioned in newly analyzed document and the candidate 'true' entities are evaluated with a cosine-similarity metric. Similarity is measured against news articles previously marked as mentioning that specific person, or in the minimal data case, against a source 'CV' article from which the relevant properties and context can be extracted.

5. Conclusion

This research shows that information retrieval techniques and natural language processing tools are sufficiently mature that commercially usable information extraction systems for specific domains can be implemented using currently published methodologies and available tools.

The accuracy of automated frame extraction varies depending on specifics of each frame type, but the average accuracy currently is 53% F-score for frame target identification, and 61% for frame element role classification[8]. A prototype of this system has been implemented and at the time of writing this abstract is currently undergoing pilot testing by analyzing news articles mentioning particular individuals, and comparing the automatically extracted data with a human analysis of the same articles. Initial error analysis indicates a strong dependency on the accuracy of initial analysis layers – mistakes in syntactic analysis or named entity recognition cause those elements to be misclassified in the semantic analysis as well.

Surprisingly large part of the system is nearly language independent – while implementing the initial text analysis steps (morphosyntactic structure, entity processing) required a significant amount of language-specific tools, at the stage of semantic role labeling the data is processed in the FrameNet representation, which is domain specific but language neutral. This would enable combining information from sources in multiple languages, if the entity mapping between languages is adequate, joining person and company names in different languages, and also 'classifier' type entities such as professions and family relations.

Adapting the existing system to different domains and languages (assuming that the generic language processing modules are available for the target language) would initially consist of (a) developing a FrameNet model for the desired information mapping and (b) annotating a semantic training corpus according to that model containing approximately 100 examples for each frame.

A specific challenge for Latvian was capturing the notion of semantically similar words in order to reduce the sparsity effect of training data. The published state of art systems for other languages gained a accuracy boost of multiple percentage points by

including WordNet lexical data, which is not available for Latvian. This gap was partially filled by manually developing a number of lists of domain-specific semantic groups of words (synonym sets of targeted verbs, lists of job titles, etc) and including them as features for the classifiers.

6. Acknowledgements

This work has been supported by the European Social Fund with the project "Support for Doctoral Studies at University of Latvia".

The research leading to these results has received funding from the research project "Information and Communication Technology Competence Center" of EU Structural funds, contract nr. L-KC-11-0003, signed between ICT Competence Centre and Investment and Development Agency of Latvia, Research No. 2.7 "Creation of the New Information Archive Access Product based on Advanced NLP".

References

[1] P. Vossen, G. Rigau, L. Serafini, P. Stouten, F. Irving, W. van Hage, NewsReader: Recording History from Daily News Streams. *Proceedings of LREC 2014, Ninth International Conference on Language Resources and Evaluation* (2014).

[2] X. Artola, Z. Beloki, A. Soroa, A Stream Computing Approach Towards Scalable NLP. *Proceedings of LREC 2014, Ninth International Conference on Language Resources and Evaluation* (2014).

[3] R. Steinberger, B. Pouliquen, E. van der Goot, An introduction to the Europe Media Monitor family of applications. *Information Access in a Multilingual World - Proceedings of the SIGIR 2009 Workshop* (2009). Boston, 1–8.

[4] J. Ruppenhofer, M. Ellsworth, M.R.L. Petruck, C.R. Johnson, J, Scheffczyk, *FrameNet II: Extended Theory and Practice. Berkeley*, International Computer Science Institute, California, USA, 2010.

[5] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider, Abstract Meaning Representation for Sembanking. *Proceedings of Linguistic Annotation Workshop* (2013).

[6] R. Johansson, P. Nugues, LTH: semantic structure extraction using non projective dependency trees. *Proceedings of SemEval-2007: 4th International Workshop on Semantic Evaluations* (2007) 227–230.

[7] D. Das, D. Chen, A. F. T. Martins, N. Schneider, N. A. Smith, Frame-Semantic Parsing. *Computational Linguistics, vol. 40:1* (2014).

[8] G. Barzdins, D. Gosko, L. Rituma, P. Paikens, (2014). Using C5.0 and Exhaustive Search for Boosting Frame-Semantic Parsing Accuracy. *Proceedings of LREC 2014, Ninth International Conference on Language Resources and Evaluation* (2014).

[9] M. Wick, S. Singh, H. Pandya, A. McCallum, A Joint Model for Discovering and Linking Entities, *Proceedings of the 2013 workshop on Automated knowledge base construction* (2013) 67–72.

[10] X. Han, L. Sun, A generative entity-mention model for linking entities with knowledge base. *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*(2011) 945–954.

[11] V. Stoyanov, J. Mayfield, T. Xu, D. W. Oard, D. Lawrie, T. Oates, T. Finin, A context-aware approach to entity linking. *AKBC-WEKEX '12 Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction* (2012) 62–67.

[12] P. Paikens, L. Rituma, L. Pretkalniņa, Morphological analysis with limited resources: Latvian example. *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013) NEALT Proceedings Series 16* (2013). Oslo, 267–278.

[13] L. Pretkalnina, L. Rituma, Statistical syntactic parsing for Latvian. *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013) NEALT Proceedings Series 16* (2013). Oslo, 279–290.

[14] A. Znotins, P. Paikens, (2014). Coreference Resolution for Latvian. *Proceedings of LREC 2014, Ninth International Conference on Language Resources and Evaluation* (2014).

PUBLIKĀCIJA XI

Riga: from FrameNet to Semantic Frames with C6.0 Rules

Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval), 2015.

Riga: from FrameNet to Semantic Frames with C6.0 Rules

Guntis Barzdins, Peteris Paikens, Didzis Gosko
University of Latvia, IMCS
Rainis Blvd. 29, Riga, LV-1459, Latvia
{guntis.barzdins,peteris.paikens,didzis.gosko}@lumii.lv

Abstract

For the purposes of SemEval-2015 Task-18 on the semantic dependency parsing we combined the best-performing closed track approach from the SemEval-2014 competition with state-of-the-art techniques for FrameNet semantic parsing. In the closed track our system ranked third for the semantic graph accuracy and first for exact labeled match of complete semantic graphs. These results can be attributed to the high accuracy of the C6.0 rule-based sense labeler adapted from the FrameNet parser. To handle large SemEval training data the C6.0 algorithm was extended to provide multi-class classification and to use fast greedy search without significant accuracy loss compared to exhaustive search. A method for improved FrameNet parsing using semantic graphs is proposed.

1 Introduction

The trend of natural language processing in recent years is shifting towards multilingual natural language understanding based on full-text shallow semantic parsing (e.g., Banarescu et al., 2013). Despite various formalisms proposed, these approaches are characterized by direct extraction of a bi-lexical semantic graph rather than a bi-lexical dependency tree from the surface form of the sentence.

Following the best practice for semantic parsing established already by the SemEval-2014 Task 8 (Oepen et al., 2014) we modified the best-performing closed-track system there (Du et al., 2014) by removing some less essential components while adding a new component of our own. The newly added component is the C6.0 rule-based classifier (Barzdins et al., 2014) used both for graph parsing and for sense labeling. Sense labeling is a novelty of SemEval-2015 Task 18 and was not present in the previous year competition. Semantic frame is comprised of a complete predication combined with the sense identifier of its predicate as shown in Figure 1. Semantic frames are similar to FrameNet (Fillmore et al., 2003) frames, except that FrameNet argument labels are sense-specific – this mismatch can be resolved by feeding the semantic graph (instead of dependency tree) through the regular FrameNet parser.

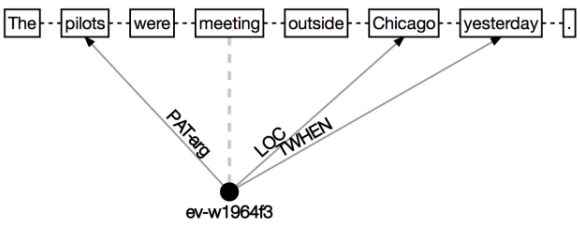


Figure 1. Semantic frame from the PSD corpus.

We participated only in the closed track. Despite ranking third for the semantic graph accuracy, our system ranked first for exact labeled match of complete semantic graphs, and close second for semantic frame accuracy.

2 Baseline Architecture

For semantic graph parsing we started by implementing a straight-forward baseline architecture described on the SemEval-2015 Task-18 evaluation page by the task organizers. The baseline architecture consists of two components: reduction

of the SDP graphs to trees and training the Mate-tools dependency parser (Bohnet, 2010) to produce such trees from the unparsed text. Instead of a destructive reduction of the SDP graphs to trees, we implemented a fully reversible depth-first transformation from the last year best-performing system (Du et al., 2014). This simple approach immediately produced competitive graph parsing results (Table 1) in line with the best-performing systems from the last year.

	in domain			out of domain		
	LP	LR	LF	LP	LR	LF
en.dm	87.34	87.05	87.19	79.95	79.42	79.68
en.pas	90.47	90.03	90.25	85.98	85.48	85.73
en.psd	72.81	71.05	71.92	70.34	67.55	68.92
cs.psd	74.44	71.56	72.97	60.19	57.43	58.78
cz.pas	82.15	81.74	81.94	-	-	-

Table 1. Baseline architecture labeled scores.

For sense labeling in *en.dm* and *en.psd* representations (a new task not present in the previous SemEval-2014 competition) we reused a technique from prior work on FrameNet labeling (Barzdins et al., 2014) based on C6.0 classifier[1]. For this task the C6.0 classifier was modified (see Section 3) to directly produce the multi-class output. By using as the features values from the *form, lemma, POS* columns for the previous, current, and next word, this approach gave good results on the development set: 93.86% accuracy for *en.psd* representation and 94.50% accuracy for *en.dm* representation. We did not try to improve it any further and the same baseline approach was used also for producing senses in the final submitted parses.

In the submitted parses we carried out the graph parsing and sense labeling completely independently, naively combining both annotations afterwards. Later experiments have shown that using graph parsing results as additional features for sense labeling would improve sense accuracy by approximately 0.2%.

3 Sense Labeling with C6.0 Rules

C6.0 rule-based classification algorithm (Barzdins et al., 2014) was inspired by the popular C4.5 decision-tree classification algorithm (Quinlan, 1993) and has been used in the state-of-the-art FrameNet parser.

To accommodate the large training data sets provided in SemEval competition we extended the original C6.0 algorithm with support for the multi-class classification and with the fast greedy search as a replacement for the exhaustive search in the original C6.0 version.

Given k training examples of the form:

$$(a_{11}, a_{12}, a_{13}, \ldots a_{1n}, class_1)$$
$$(a_{21}, a_{22}, a_{23}, \ldots a_{2n}, class_2)$$
$$\ldots$$
$$(a_{k1}, a_{k2}, a_{k3}, \ldots a_{kn}, class_k)$$

where features a_{ij} and $class_i$ are arbitrary character strings, C6.0 classifier builds a list of rules (illustrated in Figure 2) for predicting the class of unseen examples. The left side of the rule is a pattern where any feature position may contain a specific character string to be matched or an unspecified value denoted by "_".

	lemma	POS		Predicted sense	*p*	*n*	*Laplace ratio*
if(	the,	DT	*)then*	q:i-h-h	*227*	*0*	*0.996*
if(	_,	CD	*)then*	card:i-i-c	*147*	*9*	*0.937*
if(	_,	DT	*)then*	q:i-h-h	*336*	*31*	*0.913*
if(	trade,	_	*)then*	n_of:x-i	*13*	*1*	*0.875*

Figure 2. Classification rules generated by C6.0. Rule quality is estimated by the Laplace ratio based on positive *p* and negative *n* matching training examples.

The greedy search algorithm for building a multi-class classifier can be described as follows.

Training data is converted to a pool of classifier training examples. Each training example is considered positive for the class it belongs to, and negative for any other class. A candidate rule is matched against all positive and negative training examples relative to its class. The count of matched positive and negative training examples allows to calculate rule's Laplace ratio *(p+1)/(p+n+2),* where *p* is the number of matching positive training examples and *n* is the number of matching negative training examples. The rules with higher Laplace ratio are better.

For each training example a set of rules correctly classifying this training example is generated by incrementally adding to the left side of the rule feature values from this training example. Fast

[1] Available at http://c60.ailab.lv

greedy search one-by-one adds the features in such order that the resulting rule has the highest possible Laplace ratio in every feature adding iteration. This is contrary to the original C6.0 exhaustive search strategy which tried all feature relaxation combinations instead. The greedy approach eliminates exponential complexity of C6.0 with respect to feature count and when tested, yielded as good results as the exhaustive search on SemEval data.

All generated rules (regardless of the class they predict) are sorted by the highest Laplace ratio. The resulting list of rules is a multi-class classifier which can be considered consisting of multiple binary classifiers (individual rules). For unseen examples the class is assigned by the matching rule with the highest Laplace ratio.

Fig. 2 shows some classification rules for predicting the sense column value in *en.dm* training dataset from two features. The actual production classifier for sense labeling uses more features (listed in Section 2) and generates several thousand rules.

4 Semantic Graph Parsing

We tried three approaches described below to improve the graph parsing results above the baseline.

4.1 Peking and MateTools Graph Parser

The primary approach chosen for semantic graph parsing is to implement a fully reversible transformation between the semantic graph and a tree representation that encodes the extra information in edge labels. It allows training a dependency parser (Bohnet, 2010) on the labeled tree data, and using it to parse text to structures that can be converted back to a semantic graph.

For reversible graph to tree transformation we have implemented the depth-first search transformation and the auxiliary label system used by last year's best-performing Peking system (Du et al, 2014). The auxiliary labels encode:

- A separator to indicate multiple original edges encoded in this label;
- Ancestor-number indicating that in the original graph, an edge with this label is drawn from the dependent to the n-th ancestor instead of the direct parent of this tree edge;
- A reverse-edge symbol to indicate edges that have reversed direction compared to the original graph.

For the multi-root sentences that appear in some of the datasets, we choose the first root (according to word order in sentence) as the main tree root, and iteratively link all the other sentence fragments to the nearest node in the accumulated tree according to the number of words between them; in case of ties preferring the leftmost node. When creating the transformed tree, we also used special labels to distinguish the secondary root nodes of other fragments, so that the transformation is reversible for graphs with multiple root nodes.

After parsing, a tree may contain labels that are invalid according to the principles of this transformation – i.e., a reference to the grandparent of a node that does not have one. In this case, we draw an edge with the appropriate label to the closest possible node.

In this approach the cyclic graph structures are transformed to the different tree branch topologies depending on the traversal order. Traversal order thus affects the likelihood of the parser to correctly reconstruct these cyclic graph structures. To improve cyclic graph structure reconstruction we developed multiple parser variations for ensemble voting based on the following traversal orders for each node:

- Linear distance of linked words, starting with the closest words and preferring the left node in case of ties;
- Frequency of the edge labels, prioritizing the most frequent labels;

In addition, we also applied the same transformations for sentences with reversed word order to provide further variation. The resulting parsers have comparable accuracy, but produce different mistakes, making them useful for ensemble voting. Simple ensemble voting improves graph parsing accuracy over the baseline (Table 2).

	in domain			out of domain		
	LP	LR	LF	LP	LR	LF
en.dm	88.63	87.12	87.87	81.75	79.61	80.67
en.pas	91.46	90.01	90.73	87.55	85.71	86.62
en.psd	75.25	71.29	73.22	73.28	67.52	70.28
cs.psd	78.66	71.73	75.04	64.27	57.72	60.82
cz.pas	83.10	81.85	82.47	-	-	-

Table 2. Ensemble method labeled scores.

4.2 C6.0 Rule Based Graph Parser

We also applied our C6.0 rule-based classifier (described in Section 3) for semantic graph parsing through exact dependency phrase matching. Due to low recall rate it provided only a tiny positive boost to the final ensemble voting result (Table 4) despite the high precision of the rules method (Table 3). Here we considered only edges of length up to 4 and C6.0 rules with Laplace ratio above 90%. Due to low recall we signaled "abstain" vote for the edges not covered by these rules.

	in domain			out of domain		
	LP	LR	LF	LP	LR	LF
en.dm	92.80	33.47	49.20	91.84	19.78	32.56
en.pas	92.94	35.53	51.40	92.58	28.07	43.08
en.psd	88.34	18.76	30.94	86.70	11.34	20.05
cs.psd	95.29	16.70	28.42	80.46	8.13	14.77
cz.pas	90.97	22.91	36.60	-	-	-

Table 3. Labeled scores for the rules method.

4.3 Other parsing approaches

Experiments with transition based parsers (MaltParser/MaltOptimizer) showed approximately 2% lower accuracy than Mate-tools on the same transformed tree data. This is consistent with findings made by others during the earlier SemEval-2014 Task-8. We chose not to use those parsers for the final submission.

5 Final Results

We submitted two runs but report results only for run-1, because run-2 was discovered to include a corrupted Mate-tools dataset.

Our final semantic graph and semantic frames parsing results are shown in Tables 4 and 5. Semantic frames results measure overall sense labeling and graph parsing accuracy, which is the novelty of this year SemEval task.

	in domain			out of domain		
	LP	LR	LF	LP	LR	LF
en.dm	88.57	87.24	87.90	81.69	79.72	80.69
en.pas	91.50	90.02	90.75	87.56	85.72	86.63
en.psd	75.25	71.52	73.34	73.23	67.71	70.37
cs.psd	78.66	71.84	75.10	64.29	57.83	60.89
cz.pas	83.12	81.84	82.47	-	-	-

Table 4. Labeled scores for the submitted result.

	in domain			out of domain		
	FP	FR	FF	FP	FR	FF
en.dm	58.45	57.79	58.12	42.62	41.17	41.88
en.psd	52.48	52.59	52.54	40.60	40.93	40.76

Table 5. Semantic frame scores for the submitted result.

Table 6 shows ranking of averaged SemEval scoring metrics for the best runs of the systems participating in the closed task. Although we ranked third for the semantic graph (labeled dependencies) metric, our system ranked close second for semantic frame accuracy, and first for labeled exact match of the complete semantic dependency graphs. These results suggest that the C6.0 rule accuracy for sense labeling and for exact match semantic graph parsing was able to compensate for slightly lower overall graph parsing accuracy.

System	LF	LM	PF	SF	FF
Peking	**80.51**	21.14	62.64	69.45	**48.70**
Lisbon	80.42	20.05	**63.59**	--	--
Riga	78.68	**21.84**	61.29	73.76	48.33
Minsk	78.18	15.04	56.40	**79.40**	47.32

Table 6. Ranking of scores averaged over all available datasets for the best runs of the systems in the closed track: labeled dependencies (LF), labeled exact match of the complete semantic dependency graphs (LM), complete predications (PF), sense identification (SF), semantic-frames (FF).

6 Conclusions

Variations of Peking depth-first reversible graph-to-tree conversion algorithm in combination with state-of-the-art dependency parser is still a competitive graph parsing approach.

C6.0 rule-based classifier provides competitive sense labeling accuracy and some improvement also for graph parsing accuracy.

An ensemble method with "abstain" voting option for joining outputs of various graph parsing approaches boosts the results by ironing out the weaknesses of individual parsers. Required computational resources are the main limitation here.

Acknowledgments

This work was supported by the Latvian National research program SOPHIS under grant agreement Nr.10-4/VPP-4/11. We thank Lauma Pretkalniņa for the experiments with transition-based parsers.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for Sembanking. In: *Proc. Linguistic Annotation Workshop (SIGANN-2013).* Association for Computational Linguistics, pp. 178-186.

Guntis Barzdins, Didzis Gosko, Laura Rituma, and Peteris Paikens. 2014. Using C5.0 and Exhaustive Search for Boosting Frame-Semantic Parsing Accuracy. In: *Proceedings of the 9th Language Resources and Evaluation Conference (LREC),* pp. 4476-4482.

Bernd Bohnet. 2010. Very High Accuracy and Fast Dependency Parsing is not a Contradiction. *The 23rd International Conference on Computational Linguistics (COLING 2010),* Association for Computational Linguistics, pp. 89-97.

Charles J. Fillmore, Christopher R. Johnson, and Miriam R.L. Petruck. 2003. Background to FrameNet. *International Journal of Lexicography,* 16, pp. 235-250.

John R. Quinlan. 1993. C4.5: Programs for Machine Learning. *Morgan Kaufmann Publishers.* 302 p.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajic, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 Task 8: Broad-Coverage Semantic Dependency Parsing. *Proceedings of the 8th In-ternational Workshop on Semantic Evaluation (SemEval-2014)*. Association for Computational Linguistics, pp. 63-72.

Yantao Du, Fan Zhang, Weiwei Sun, and Xiaojun Wan. 2014. Peking: Profiling Syntactic Tree Parsing Techniques for Se-mantic Graph Parsing. *Proceedings of the 8th In-ternational Workshop on Semantic Evaluation (SemEval-2014)*. Association for Computational Linguistics, pp. 459-464.

PUBLIKĀCIJA XII

Tezaurs.lv: the Largest Open Lexical Database for Latvian

Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC), 2016.

Tezaurs.lv: the Largest Open Lexical Database for Latvian

Andrejs Spektors, Ilze Auzina, Roberts Dargis, Normunds Gruzitis, Peteris Paikens, Lauma Pretkalnina, Laura Rituma, Baiba Saulite
University of Latvia, Institute of Mathematics and Computer Science
Raina blvd 29, Riga, Latvia
name.surname@lumii.lv

Abstract

We describe an extensive and versatile lexical resource for Latvian, an under-resourced Indo-European language, which we call Tezaurs (Latvian for 'thesaurus'). It comprises a large explanatory dictionary of more than 250,000 entries that are derived from more than 280 external sources. The dictionary is enriched with phonetic, morphological, semantic and other annotations, as well as augmented by various language processing tools allowing for the generation of inflectional forms and pronunciation, for on-the-fly selection of corpus examples, for suggesting synonyms, etc. Tezaurs is available as a public and widely used web application for end-users, as an open data set for the use in language technology (LT), and as an API – a set of web services for the integration into third-party applications. The ultimate goal of Tezaurs is to be the central computational lexicon for Latvian, bringing together all Latvian words and frequently used multi-word units and allowing for the integration of other LT resources and tools.

Keywords: Lexicon, Dictionary, Thesaurus, Morphology, Latvian, API

1 Introduction

Tezaurs,[1] a machine-readable lexicon and an online dictionary for Latvian, one of the 24 official EU languages, has been around for a while. The initial human-oriented version of this resource was made publicly available in 2009, comprising more than 125,000 entries that were consolidated from around 40 sources: modern and historical dictionaries, mostly available in a printed form. Since then, Tezaurs has been updated once every three months, and so far it has grown to more than 250,000 entries referring to more than 280 sources.

Tezaurs has attracted a large end-user base[2] and an increasing interest from third-party application developers, however, this work has not been published before.

The ultimate goal of Tezaurs is to be the central open computational lexicon for Latvian, allowing for the integration of other resources and tools for language technology (LT). An analogy can be drawn to SALDO (Borin et al., 2013), a lexical database for Swedish, the central component in an integrated infrastructure for computational lexical resources.

The idea, theoretically, is to bring together all the Latvian words and frequent multi-word units, along with their morpho-syntactic features and meaning, that have been used in the written texts. A secondary aim is to create and maintain a reliable source for language users, where they can verify and learn word forms, senses, and the lexical and grammatical valency.

For the language users, Tezaurs is already a highly popular online reference dictionary.[3] In addition to the fact that it is derived and consolidated from existing sources, Tezaurs provides added value: inflectional tables, phonetic transcriptions, synonym sets, and corpus examples. All the data and the accompanying web services are open-source and open-access.

[1] http://tezaurs.lv

[2] Around 195,000 unique visitors (78% returning) over the last 12 months; around 67,500 sessions per month.

[3] More than 4.5 million page (entry) views per year. (There are about 2 million Latvian speakers worldwide.)

2 Wordlist

Tezaurs is already a useful LT resource even only as an extensive authoritative vocabulary with (optionally) additional attributes for each word: the homonym index, the part-of-speech (POS) category, the inflectional paradigm, the phonetic transcription, domains of usage, stylistic markers and usage restrictions (dialecticism, archaic, colloquial, slang, vulgarity, child speech, etc.), as well as references to the sources.

The additional features allow for calling the Tezaurs web services, e.g. to generate a table of possible word forms based on the lemma and the inflectional paradigm, and for selecting a sub-vocabulary depending on the particular use case and application. Tezaurs has already been used as a source of general-purpose or customized wordlists in various text analysis pipelines that tend to have conflicting requirements on inclusion or exclusion of e.g. slang, archaisms or specific domains. To mention a few examples, Tezaurs' wordlists have been exploited in a newswire information extraction system (Paikens, 2014), in the transliteration and correction of OCR errors in historical texts (Pretkalnina et al., 2012), in an open-source spell checker, in various word games like Scrabble, and in other smaller research and commercially oriented applications.

Currently, a list of headwords along with their homonym indices, part-of-speech categories, inflectional paradigms and source references is available in the public repository of Tezaurs open data.[4] The remaining word attributes are under revision.

The wordlist is available also a web service that returns either the whole wordlist[5] or a detailed set of the above mentioned attributes for a particular word[6] along with homonyms, if any.

3 Morphological Information

The current end-user interface integrates a morphological web service, an extension of an open-source morpholog-

[4] https://github.com/LUMII-AILab/Tezaurs

[5] http://api.tezaurs.lv/v1/words/

[6] http://api.tezaurs.lv/v1/words/doma

ical analyzer for Latvian (Paikens et al., 2013), as a way of generating inflection tables for the lexical entries. Consequently, it also supports the inclusion of the Tezaurs wordlist (or a subset of it) as a lexicon for POS and morphological tagging and for lemmatization.

Although the source dictionaries do not include the morphological information of the headwords, or they include only a partial information, we can semi-automatically detect the POS category and the inflectional paradigm for each word. In most cases this can be done automatically, although quite a few cases have a chance for errors or uncertainty until the particular word groups are manually reviewed.

The main challenge is due to the tradition in the Latvian lexicography, which typically does not specify the POS category (a consequence of a highly inflected language). As of authors knowledge, the only Latvian dictionary that consistently includes POS tags is the Dictionary of Modern Latvian Language, MLVV.[7] MLVV is only now being transformed into a machine-readable form. When this is done, it will cover about 20% of entries in Tezaurs. Thus, in cases where the POS category cannot be unambiguously determined by the formal indications such as the word ending, the detection of the POS category and the specific inflectional paradigm of that category requires taking the meaning of the word (homonym) into account.

Another challenge is the need for manual reviewing of entries that include hints for non-standard inflectional paradigms, particularly in case of archaic and dialectal words whose inflection might not be aligned with the modern (standard) grammar, e.g. they can lack some word forms. Note that Tezaurs includes more than 90,000 dialectal and archaic words.

The morphological features of each word form included in the inflection table (returned by the web service) are only partially included in the end-user interface. The service provides the detailed morphological descriptions either in a form of MULTEXT-East morphosyntactic tags (Erjavec, 2004) or as an ISOcat feature matrix (Windhouwer and Wright, 2012) which is exemplified in Figure 1. The web service can be integrated in third-party applications in combination with the features provided by the Tezaurs wordlist (particularly, the inflectional paradigm).

4 Phonetic Transcription

In most cases, there is a one to one mapping between graphemes and phonemes in Latvian. Therefore the source dictionaries typically do not include information about the pronunciation of headwords, except in rare cases. Such cases include, for instance, words with contrastive syllable tones which can change the meaning of orthographically identical words, e.g. *zāle*: [zãle] (level tone) 'hall, large room' vs. [zâle] (broken tone) 'grass, herb'. However, two specific graphemes – 'e' pronounced as 'e' or 'æ', and 'o' pronounced as 'u͡o' (as in *doma* 'thought'), 'ɔ' or 'ɔː' – require an informed choice to pronounce the word correctly, and the pronunciation may vary across inflectional forms, even with the same spelling.

Our recent research on Latvian speech processing has resulted in a rule-based system that captures the pronunciation patterns and generates a machine-readable phonetic transcription for the given isolated word (Auzina et al., 2014). The system is now accessible as a Tezaurs web service[8], and it is being integrated in the Tezaurs website and the data sets (starting with the wordlist). In combination with a text-to-speech service (Pinnis and Auzina, 2010), this will make Tezaurs a more useful resource for language learners.[9] The transcription service, however, occasionally makes mistakes in case of the 'e' and 'o' graphemes. Again, after processing and integrating the MLVV data, this issue will be fixed at least for frequently used words.

In future, the morphological service (Section 3) can be extended by the transcription service to generate inflectional tables that are enriched with the phonetic transcriptions. Note that for verbs the pronunciation of the stem may change across inflectional forms.

5 Dictionary Entries

Another primary facet of Tezaurs: it is an extensive explanatory online dictionary. An entry generally represents a partial morphological information of the headword, usage restrictions (if any), the sense split, multi-word units and idioms, and source references. Homonyms and homographs (for more than 4,500 words) are given as separate entries with different indices.

Entries are internally organized by word senses (around 325,000 senses in total; 1.3 senses per headword). Each sense is explained by a full definition or a synonymous cross-reference. Morphological and stylistic restrictions can be specified also at the sense level. Senses often include embedded micro-entries of multi-word units along with their usage restrictions and glosses (around 32,000 in total). Some entries embed also idiomatic micro-entries (more than 11,000 in total) which are related to the whole entry. Usage examples are generated on-the-fly from a balanced corpus, where possible, as described in Section 7.

An example entry, as presented for the end-user, is given in Figure 2.

There is a web service available[10] that returns the dictionary entries in the LMF format, the standard interchange format for lexical resources (Hayashi et al., 2013).

6 Semantic Relations

Last but not least, Tezaurs is an extensive source for synonyms and other related concepts. Currently, we have put the focus on the synonymy relations which are automatically extracted from the implicit cross-references in the glosses which in turn follow traditional lexicographic guidelines. An issue is that although the sense split is obvious for the outgoing synonym sets (synsets), the incoming

[7] *Mūsdienu latviešu valodas vārdnīca*. University of Latvia, Institute of Latvian Language, 2004–2014 [http://tezaurs.lv/mlvv/]

[8] http://api.tezaurs.lv/v1/transcriptions/doma?encoding=ipa

[9] http://api.tezaurs.lv/v1/pronunciations/doma

[10] http://api.tezaurs.lv/v1/entries/doma/1

```
[{
  "lemma" : "doma",
  "grammaticalGender" : "feminine",
  "declension" : "4",
  "partOfSpeech" : "noun",
  "wordForms" : [
    {"wordForm" : "doma",  "case" : "nominativeCase", "grammaticalNumber" : "singular"},
    {"wordForm" : "domas", "case" : "genitiveCase",   "grammaticalNumber" : "singular"},
    {"wordForm" : "domai", "case" : "dativeCase",     "grammaticalNumber" : "singular"},
    {"wordForm" : "domu",  "case" : "accusativeCase", "grammaticalNumber" : "singular"},
    {"wordForm" : "domā",  "case" : "locativeCase",   "grammaticalNumber" : "singular"},
    {"wordForm" : "doma",  "case" : "vocativeCase",   "grammaticalNumber" : "singular"},
    {"wordForm" : "domas", "case" : "nominativeCase", "grammaticalNumber" : "plural"},
    {"wordForm" : "domu",  "case" : "genitiveCase",   "grammaticalNumber" : "plural"},
    {"wordForm" : "domām", "case" : "dativeCase",     "grammaticalNumber" : "plural"},
    {"wordForm" : "domas", "case" : "accusativeCase", "grammaticalNumber" : "plural"},
    {"wordForm" : "domās", "case" : "locativeCase",   "grammaticalNumber" : "plural"},
    {"wordForm" : "domas", "case" : "vocativeCase",   "grammaticalNumber" : "plural"}
  ]
}]
```


Figure 1: A slightly simplified representation of `http://api.tezaurs.lv/v1/inflections/doma?paradigm=7` (‘thought’).

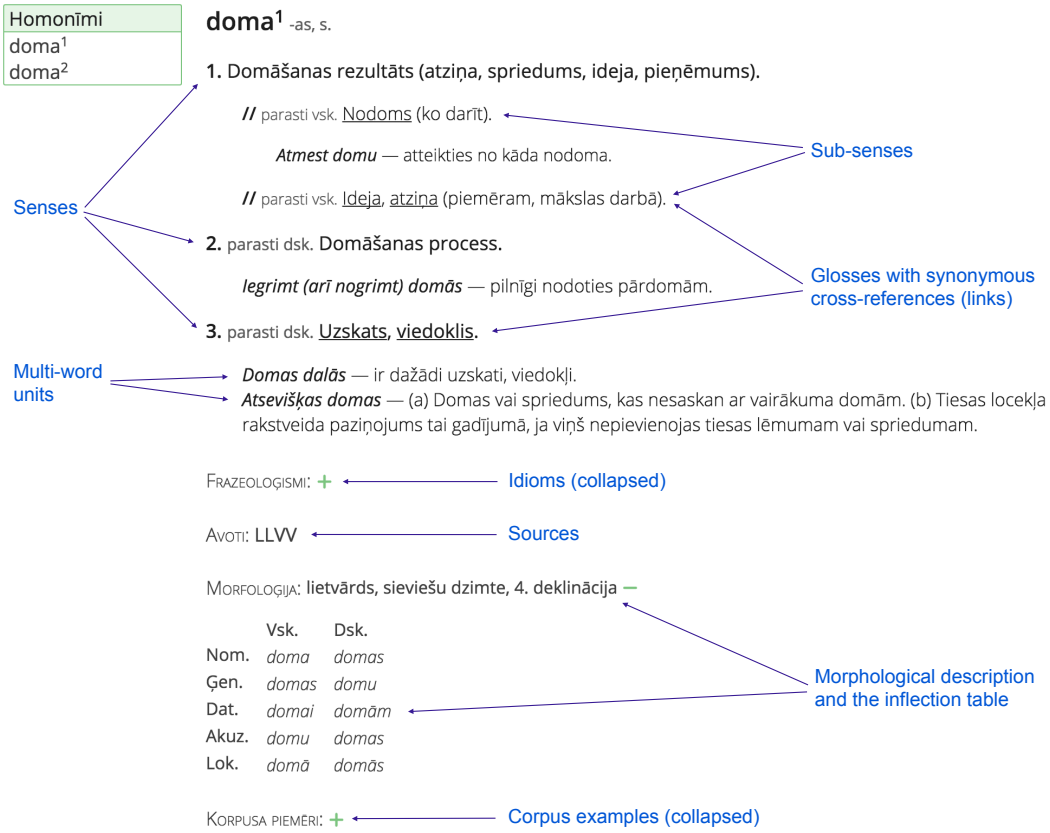


Figure 2: A slightly simplified end-user presentation of the entry `http://tezaurs.lv/#/sv/doma/1` (‘thought’).

sense is usually not specified in the glosses and, in general, has to be decided heuristically. In the long term, this will be a motivation to fix the ambiguous glosses manually.

The extracted synsets will be provided as open data along with the Tezaurs wordlists. We also intend to provide a corpus-driven list of semantically related words based on the *word2vec* approach (Mikolov et al., 2013). This does not necessarily reveal synonyms, but is interesting for human exploration and also as a feature for NLP tools.[11]

7 Corpus Examples

Availability of usage examples helps in understanding the meaning and customary usage of the words, however, appropriate sample sentences have generally not been available. Many source dictionaries do not include them, and for those that do, there are various problems that preclude directly using this data in Tezaurs - copyright issues, outdated usage, unavailability of the primary sources.

We currently provide[12] usage examples automatically retrieved from a balanced text corpus (Levane-Petrova,

[11] A demo of the already acquired vectors for Latvian is available at `http://api.tezaurs.lv/v1/embeddings/`

[12] `http://api.tezaurs.lv/v1/examples/doma`

2012), which provides adequate examples of contemporary usage for common words. The major issue that we encounter is the handling of homographs: morphological tagging and automatic word sense disambiguation helps, but is not perfect and needs manual review of such results.
While this provides useful results for common words, the coverage is limited by the size of corpus and for rare words usage examples are arguably even more important. This is an active direction of ongoing work to integrate data available from large unbalanced corpora of varying quality and/or web searches.

8 Sources

The primary source that has been used to derive the Tezaurs entries is the Dictionary of Standard Latvian Language, LLVV.[13] Almost 65,000 entries have been derived from LLVV (more than 25% of all Tezaurs entries).
There are about 20 secondary sources, each of them used in at least 1% of all entries (in total, around 149,000 entries refer to the secondary sources). The rest is a long tail of about 260 peripheral sources, each of them used in less than 1% of all entries; about 62,000 entries in total. Among them, less than 60 sources are used in 0.1–1.0% of all entries (each); about 55,000 entries in total.

9 Conclusion and Future Tasks

Tezaurs has acquired an important role for the human consumption (incl. professional translators, students, researchers, terminologists). We have also used this data set internally in the development of NLP tools, e.g. to extend the coverage of the POS-tagger (Paikens et al., 2013), to validate the correction of OCR errors (Pretkalnina et al., 2012), etc. We are anticipating an interest from researchers and application developers in the Tezaurs open machine-readable data and web services. The database attracts more and more interest from third-party application developers, both open-source and commercial, e.g. to be integrated in information retrieval systems, spellcheckers, style checkers, language games etc.
Future tasks include separate research problems that can be addressed based on this work. To mention some of them:

- Integration with a verb valency lexicon for Latvian (Nespore et al., 2012). The mapping of particular word senses to verb valencies needs to be done manually, which is feasible for the frequently used verbs.
- Providing corpus-based typical collocation information for each word.
- Further development of the semantic relations between word senses towards a WordNet-like semantic network.
- Integration with Linked Open Data to allow for word-sense grounding etc.
- Linking corpus usage examples to specific word senses by using word embeddings or similar techniques.

[13] *Latviešu literārās valodas vārdnīca*. 1.–8. Riga: Zinātne, 1972–1996 [http://tezaurs.lv/llvv/]

Acknowledgements

This work has been partially supported by Latvian State Research Programmes: Letonika (Project No. 3), NexIT (Project No. 1) and SOPHIS (Project No. 2).

References

Auzina, I., Pinnis, M., and Dargis, R. (2014). Comparison of rule-based and statistical methods for grapheme to phoneme modelling. In *Human Language Technologies – The Baltic Perspective*, volume 268 of *Frontiers in Artificial Intelligence and Applications*, pages 57–60. IOS Press.

Borin, L., Forsberg, M., and Lonngren, L. (2013). SALDO: a touch of yin to WordNet's yang. *Language Resources and Evaluation*, 47(4):1191–1211.

Erjavec, T. (2004). MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*, pages 1535–1538, Lisbon, Portugal.

Hayashi, Y., Monachini, M., Savas, B., Soria, C., and Calzolari, N., (2013). *LMF as a foundation for servicized lexical resources*, pages 201–213. Wiley.

Levane-Petrova, K. (2012). The balanced corpus of modern Latvian and the text selection criteria. *Baltistica*, VIII Priedas:89–98.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS)*.

Nespore, G., Saulite, B., Gruzitis, N., and Garkaje, G. (2012). Towards a Latvian valency lexicon. In *Human Language Technologies – The Baltic Perspective*, volume 247 of *Frontiers in Artificial Intelligence and Applications*, pages 154–161. IOS Press.

Paikens, P., Rituma, L., and Pretkalnina, L. (2013). Morphological analysis with limited resources: Latvian example. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA)*, pages 267–277, Oslo, Norway.

Paikens, P. (2014). Latvian newswire information extraction system and entity knowledge base. In *Human Language Technologies – The Baltic Perspective*, volume 268 of *Frontiers in Artificial Intelligence and Applications*, pages 119–125. IOS Press.

Pinnis, M. and Auzina, I. (2010). Latvian text-to-speech synthesizer. In *Human Language Technologies – The Baltic Perspective*, volume 219 of *Frontiers in Artificial Intelligence and Applications*, pages 69–72. IOS Press.

Pretkalnina, L., Paikens, P., Gruzitis, N., Rituma, L., and Spektors, A. (2012). Making historical Latvian texts more intelligible to contemporary readers. In *Proceedings of the LREC Workshop on Adaptation of Language Resources and Tools for Processing Cultural Heritage Objects*, Istanbul, Turkey.

Windhouwer, M. and Wright, S. E., (2012). *Linking to Linguistic Data Categories in ISOcat*, pages 99–107. Springer.

PUBLIKĀCIJA XIII

Deep Neural Learning Approaches for Latvian Morphological Tagging

Proceedings of Human Language Technologies – the Baltic Perspective, 2016.

Deep Neural Learning Approaches for Latvian Morphological Tagging

Pēteris PAIKENS[1]
University of Latvia, Institute of Mathematics and Computer Science

Abstract. This paper describes ongoing research on improvements of morphological analysis, disambiguation and POS tagging for the Latvian language. Authors apply recent advances in sequential tagging with neural networks and word embeddings calculated from unlabeled corpus to improve morphological tagging accuracy. These approaches allow to reduce the fine-grained morphological tag word error rate from 7.9% of earlier best systems to 6.2%, and coarse-grained POS tag error rate from 3.6% to 2.2%.

Keywords. morphology, tagging, deep learning, neural networks

1. Introduction

Morphological analysis and tagging is a commonly required key stage in most natural language processing systems, especially for morphologically rich languages such as Latvian. Currently various morphological taggers are available for Latvian, but their accuracy lags behind the larger languages such as English. While it's reasonable to expect lower accuracy to distinguish between the many tags possible in a morphologically rich language, even for the coarse part of speech categories the best previously reported accuracy scores for Latvian have an error rate twice as large as the state of the art taggers for English – 5% vs 2.5% [1,2].

This is caused in part by the comparably much smaller amount of available annotated training data. However, recent advances in deep neural network machine learning have not only shown the potential to improve supervised learning tasks, but also can learn powerful representations from unlabeled data, e.g. word embeddings [3] highlighting one possibility to partly cross this accuracy gap.

In this paper we describe the ongoing experiments to apply neural network based approaches to the task of fine-grained morphological tagging of Latvian text. In addition to the linguistic resources used in earlier systems – annotated corpora, lexical resources and output of a rule-based morphological analyzer – we now also augment the system with additional word embedding data from a large unlabeled corpus [4]. In order to evaluate these results, we compare the new system with the current state-of-art taggers publicly available for Latvian.

[1] peteris@ailab.lv

2. Problem Description

For the purposes of this task, we attempt to solve the problem of fine-grained morphological tagging – obtaining a tag that specifies the morphosyntactic properties of each word, while also evaluating the accuracy of the coarse-grained POS tagging.

We implement the following hypothetical improvements in order to evaluate their effect on the accuracy of morphological analysis of Latvian:

- Word embedding data, calculated from a large untagged corpus;
- Various neural network approaches – convolutional neural networks, bidirectional LSTM networks with a CRF layer which has shown excellent results for English [2] and 'wide and deep' structures [5];
- Different representations of morphosyntactic information – including data from paradigm-based morphological analyzer and replacing the classic approach of distinct tags with separate sets of output neurons for each morphosyntactic property, trained together.

3. Related Work and Evaluation Methodology

Current published work on Latvian morphological tagging includes two comparable taggers. One of baseline systems is a conditional Markov model statistical tagger based on Stanford CoreNLP system [6] as described in [7], and the other is based on averaged perceptron as described in [1]. The source code of both these systems is available on GitHub with a permissive license, and their accuracy is comparable – [1] reports 93.60% vs 93.67% accuracy scores on the same set of test data.

There is also earlier work that has been used in Tilde proprietary systems [8], but that is closed source and has been superseded by the newer systems, so it was not replicated and evaluated in this paper.

Current most relevant related work for tagging methods, achieving best results when evaluated on standard English datasets, is the research on LSTM-CRF combination [2]. There is an interesting recent implementation [9] that claims even better results, but at the moment of writing this paper the full details are not yet available.

3.1. Training data

For training and evaluation, we use the current versions of data from the contemporary balanced corpus of Latvian [10] and the Latvian treebank [11]. The designated split of data contains 95 012 tokens as the training corpus and 7 293 tokens as development corpus for tuning and testing the system, and for the work-in-progress evaluations and comparisons of various strategies. A separate evaluation corpus of 7 020 tokens was set aside and used at article submission time for the final evaluation and system comparison.

The data is split in these partitions on a per-document basis, as there is significant intra-document overlap of rare vocabulary and proper nouns, which generally are harder to analyze, and in sentence-based randomized splitting those words are shared between training and evaluation data. Because of this effect, document-based split of training and evaluation data would be a more accurate metric of how the taggers would generalize to new documents. The sentence-based randomization produces an artificially elevated metric, because the system has seen the majority of every document during training.

Due to this change in training and testing data, the numeric results are rather different and not directly comparable with other papers using earlier versions of the same corpus, so the earlier methods were also re-trained and re-evaluated on the current test set.

3.2. Baseline systems

The new results are compared with the two existing systems described above – Paikens-2012 and Ņikiforovs-2015. We use the latest version of code as available on GitHub, but re-train the models on the abovementioned set of training data to ensure a fair comparison. This test set appears to more difficult in part due to the change from sentence-based split to document-based split between training and evaluation data and the numeric results are not directly comparable with earlier papers.

In addition, we also calculate a naïve baseline, obtaining by simply picking the most frequently seen tag out of the tag candidates supplied by the morphological analyzer.

4. System Architecture

For the purposes of this paper, a large variety of neural network structures were tested during system development, but limiting all of them to pure neural network architectures with no post-processing. All experiments shared a common structure of input and output (evaluation) data and were implemented in Tensorflow for GPU-based machine learning.

For input, we use the following features:

- a one-hot encoding of the word form according to the vocabulary of training corpus with rare words treated as out of vocabulary;
- pre-calculated word embedding model [4];
- one-hot encodings of suffix letter n-grams up to length of 4;
- an n-hot vector showing which of the possible candidates for morphosyntactic tags are considered valid for this word, taken from a morphological analyzer based on lexicon and inflectional paradigms [12].

For output, we considered three different vector encodings – a one-hot vector of the possible coarse-grained part of speech categories (13 options), a one-hot vector of the fine-grained morphosyntactic tags (430 options), and an encoding representing each possible morphological attribute-value pair separately (70 elements); functionally equivalent to the fine-grained tag as each can be constructed from the other.

The currently best performing system (labeled “Full NN system” in evaluation) is a combination of various elements with the structure illustrated in Figure 1. It starts with fully connected neural network layers calculating a compressed representation of the comparably wide (~5000 units each) word form and suffix encodings, followed by a drop-out layer to facilitate generalization. This is concatenated together with the other input vectors and fed to a convolution layer that combines features from neighboring words to capture the close-range relations. Convolution window size of just 3 words was found sufficient, as farther relations are captured by a bidirectional layer of long short-term memory (LSTM) cells as initially proposed by [13], thus encoding both the forward and backward context. The final classification is done by a logistic function on the output of LSTM layer (after dropout) combined with the full, wide content of all input features as suggested by [5]. A concatenation of all three output types is used in training to minimize the cross-entropy between network output and expected values using Adam

optimizer algorithm [14] and applying standard regularization to network coefficients. The network converges in 20 epochs in less than 2 hours on a NVidia TitanX GPU based system.

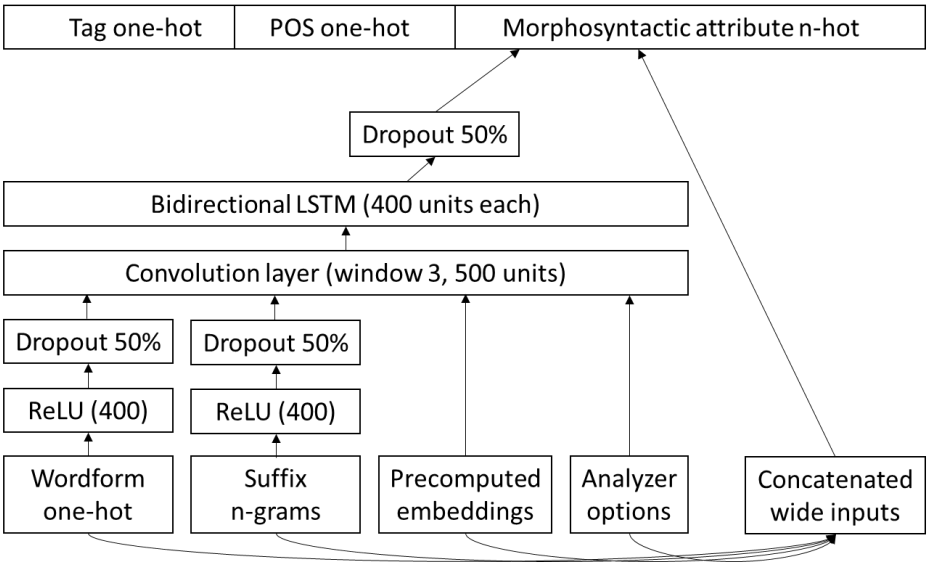


Figure 1. Network structure.

We also evaluate a minimalistic neural network structure, consisting of the abovementioned input layers, a single bidirectional LSTM layer with 200 cells, and the logistic output layer on top of that.

A large variety of other network structures were explored in experiments, but not exhaustively evaluated to verify and prove the effects of each separate factor. Nonetheless, the following observations and experience may be useful to the reader.

The choice of output encoding was highly significant. Using only the one-hot representation of tags (which seems to be the most commonly used approach in literature) without the separate attribute-value encoding lost about 1 full percentage point of accuracy.

Deeper network architectures beyond the proposed structure did not improve accuracy. We performed numerous experiments to explore various depths (up to 12) and layouts of recurrent and fully connected layers but these yielded the same or lower accuracy despite a much higher learning time or, in some configurations, performed significantly worse due to overfitting issues.

From the perspective of accuracy, the initial ReLU (rectified linear unit) layers after word form and n-gram encoding could better be replaced with a layer over the whole input vector set, however, they were necessary for performance reasons as the combination of wide inputs (11000-25000 neurons per word depending on vocabulary filtering) with larger sizes of further recurrent or convolutional layers result in operations that are impractical to train even on current top-end GPUs due to memory limitations.

5. Evaluation and Error Analysis

We compare the developed system against the baseline systems described in section 3.2, re-training them on the same set of updated corpora. The evaluation is shown in Table 1. In addition, we also consider three limited options:

- A much simpler NN model, consisting of only a single LSTM layer with 200 units between the input and output layers described earlier;
- A system which omits the morphological analyzer information while otherwise being identical to the full recommended system.
- A system trained without the attribute-value output, using only tag and POS.

Table 1. System evaluation

System	Full tag accuracy	POS accuracy
Naïve baseline	71.9%	88.6%
Paikens-2012	91.4%	95.1%
Ņikiforovs-2015	92.1%	96.4%
Simple NN model	93.2%	97.6%
No analyzer	92.8%	97.7%
No attribute encoding	92.7%	97.7%
Full NN system	**93.8%**	**97.8%**

When run on a dataset with per-sentence split of training and evaluation data, the same dataset used in earlier experiments [1,7], the full NN system scores 95.4% for the full tag accuracy and 98.3% for POS accuracy. However, we don't consider those metrics as appropriate for evaluation because of issues described in section 3.1.

After performing the evaluation, a classification of errors of the best performing system on the test set was performed. The most popular errors (repeating 10 times or more) are shown on Table 2 and the per-feature error rates are shown in Table 3. Words that are out of vocabulary (with respect to training corpus) were found to have just slightly lower accuracy than average – 91.1% for full tag and 96.4% for POS.

Table 2. Popular errors

Feature	Predicted value	Annotated value	Number of cases
Number	Singular	Plural	87
Number	Plural	Singular	42
Case	Genitive	Accusative	35
Case	Accusative	Genitive	33
Gender	Feminine	Masculine	32
Gender	Masculine	Feminine	32
Case	Genitive	Nominative	25
Case	Nominative	Genitive	20
POS	Residual	Noun	17
POS	Noun	Abbreviation	16
Definiteness	Definite	Indefinite	14
POS	Adjective	Verb	12
POS	Verb	Adjective	11
POS	Noun	Residual	10
POS	Residual	Abbreviation	10

As in earlier systems, the most popular error is the confusion between singular accusative and plural genitive, which are homoforms for many nouns and adjectives and whose disambiguation requires determining the case of a long noun phrase. The tagging errors for gender are in cases of contextual gender of pronouns and participles, where determining the 'correct' gender requires deciding to which noun this word refers.

The part of speech errors, on the other hand, seem to be caused by problems in corpus annotation. Names of foreign companies in newswire documents are variously

tagged as inflexive nouns, residuals (foreign words) or abbreviations, causing confusion in such cases; and words which morphologically are derived from verbs but have obtained an independent adjective meaning are also inconsistently tagged as either verbs (participles) or adjectives, and thus show up as tagging errors.

Table 3. Feature error rates

Feature	Error rate (for POS having this feature)
Part of speech	2.2%
Case	4.2%
Number	3.1%
Definiteness	3.0%
Pronoun type	2.3%
Gender	1.8%
Verb mood	0.6%
Residual type	0.4%

6. Conclusions and future work

Current experiments already noticeably outperform the baseline systems, showing that the approach is viable for improving morphosyntactic analysis of Latvian language, obtaining a significant increase in accuracy – the 1.7 percentage point improvement in tag accuracy amounts to a word error rate reduction from 7.9% to 6.2%, solving 20% of earlier system errors.

As in many use cases the next step in text processing is syntactic parsing, the dominant types of errors raise a peculiar Catch-22 situation – correct morphological tagging in these situation requires knowing the correct syntactic interpretation, while syntactic parsing requires morphological information and in these situations receiving wrong tags would result also in wrong syntactic dependencies. This suggests that further improvements in accuracy of morphological tagging might require doing syntactic parsing at the same time, as done by e.g. SyntaxNet [15].

As future work, it would be interesting to explore possibilities for replacing the morphological analyzer data with a character-level recurrent neural network, attempting to learn from unlabeled corpus the information that is currently taken from inflectional paradigms and lexical resources. Currently the system is usable without the morphological analyzer data, but suffers a noticeable decrease in tag accuracy.

It is interesting to note the surprisingly large effect of output representation as separate attributes instead of a list of tags. It may be worth exploring this effect in a more focused manner and check if it also holds for other morphologically rich languages.

The code and data for the final experimental system is freely available in GitHub at https://github.com/PeterisP/tf-morphotagger. Additional future work is expected in packaging this tagger for public use. While the earlier systems were easily distributable as Java and C# packages respectively, distributing Tensorflow systems to nontechnical end-users in a convenient way is difficult.

7. Acknowledgements

This research has been supported by Latvian State Research Programme SOPHIS (Project No. 2).

References

[1] Ņikiforovs, P. *Latviešu valodas morfosintaktiskais marķētājs.* Bachelor thesis, University of Latvia, 2015.

[2] Huang, Z., Xu, W., Yu, L. *Bidirectional LSTM-CRF Models for Sequence Tagging*. arXiv preprint arXiv:1508.01991 [cs.CL], 2015.

[3] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS)*, 2013.

[4] Znotiņš A. Word Embeddings for Latvian Natural Language Processing Tools. In this volume, 2016.

[5] Cheng, H. T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., ... & Anil, R. Wide & Deep Learning for Recommender Systems. arXiv preprint arXiv:1606.07792, 2016.

[6] Toutanova K., Klein D., Manning C.D. and Singer Y. Feature-Rich Part-of- Speech Tagging with a Cyclic Dependency Network. *Proceedings of HLT-NAACL* (2003), 252–259.

[7] Paikens, P., Rituma, L., and Pretkalniņa, L. Morphological analysis with limited resources: Latvian example. *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA)* (2013), 267– 277.

[8] Pinnis, M. and Goba, K. Maximum Entropy Model for Disambiguation of Rich Morphological Tags. *Systems and Frameworks for Computational Morphology, Communications in Computer and Information Science, 1, Volume 100, The 2nd Workshop on Systems and Frameworks for Computational Morphology (SFCM2011)*, Heidelberg, Springer (2011) 14–22.

[9] Choi, J.D. Dynamic Feature Induction: The Last Gist to the State-of-the-Art. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (NAACL'16)* 2016.

[10] Levāne-Petrova K. Morfoloģiski marķēta valodas korpusa izmantošana valodas izpētē. *"Vārds un tā pētīšanas aspekti": Rakstu krājums 15(1)*, Liepāja, LiePA (2011) 187–193.

[11] Pretkalniņa L., Nešpore G., Levāne-Petrova K., and Saulīte B. Towards a Latvian Treebank. *Actas del 3 Congreso Internacional de Lingüística de Corpus. Tecnologias de la Información y las Comunicaciones: Presente y Futuro en el Análisis de Corpus*, eds. Candel Mora M.Á., Carrió Pastor M., (2011) 119–127

[12] Paikens, P. Lexicon-based morphological analysis of Latvian language. *Proceedings of 3rd Baltic Conference on Human Language Technologies (HLT 2007)*, (2007) 235–240.

[13] Graves, A., Mohamed, A., Hinton, G. Speech Recognition with Deep Recurrent Neural Networks. arXiv preprint arXiv:1303.5778 [cs.NE], 2013.

[14] Kingma, D., & Ba, J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[15] Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., ... & Collins, M. Globally normalized transition-based neural networks. arXiv preprint arXiv:1603.06042, 2016.

PUBLIKĀCIJA XIV

SUMMA at TAC knowledge base population task 2016

Proceedings of the Ninth Text Analysis Conference (TAC 2016), 2016.

SUMMA at TAC Knowledge Base Population Task 2016

Peteris Paikens[*] **Guntis Barzdins**[*] **Afonso Mendes**[†] **Daniel Ferreira**[†]
Samuel Broscheit[#] **Mariana S. C. Almeida**[†] **Sebastião Miranda**[†] **David Nogueira**[†]
Pedro Balage[†] **André F. T. Martins**[#,†]
[*]University of Latvia / News agency LETA
[†]Priberam Labs, Alameda D. Afonso Henriques, 41, 2º, 1000-123 Lisboa, Portugal
[#]Instituto de Telecomunicações, Instituto Superior Técnico, 1049-001 Lisboa, Portugal
{amm,mla,summa}@priberam.pt, Peteris.Paikens@leta.lv

1 Introduction

Our submission to the NIST TAC-KBP-2016[1] is an initial attempt to apply our ongoing research on text analysis within SUMMA project[2] to TAC shared tasks. The goal of SUMMA is to develop a scalable and extensible media monitoring platform with an automatic knowledge base construction and cross-lingual capabilities, thus having a significant overlap with TAC-KBP tasks. For this first TAC participation, our system was only run for the Entity Discovery and Linking (EDL) and Cold Start Knowledge Base Population (KBP) tasks as a way to evaluate our initial system. In the next edition of TAC-KBP, we expect to participate with a more mature system.

The paper is organized as follows. Section 2 and Section 3 describe our contribution to the EDL and to the Cold Start KBP tracks, respectively. Experimental results are reported in Section 4, and Section 5 concludes the paper.

2 Entity Discovery and Linking

2.1 Submissions

Two systems were submitted to the first evaluation window of the EDL track. The first system, *summa1*, is an initial implementation of a language independent approach. The system is based on an implementation of SVM-Rank (Herbrich et al., 2000) trained with "universal" features, namely features obtained from pre-trained cross-lingual representations (Ferreira et al., 2016). Despite having a cross-lingual framework, due to evaluation window time constraints we submitted our results only for English. The second submission, *summa2*, is a ruled-based system for English, that evaluates the impact of several steps into the linking quality.

Since *summa2* outperformed *summa1* in the first evaluation window, in the second evaluation window we focused on an improved version of *summa2*, by adding a candidate ranking step based on nearest-neighbours retrieval and a novel cross-document coherence step. Ahead, this section provides a description of our final submission – *summa3*.

2.2 Entity Recognition and Labeling

Model and features. For detecting and labeling mentions, we use the named entity recognizer (NER) available within TurboParser[3] (Martins et al., 2013). This NER implements a linear sequential model whose features are based on the Illinois Entity Tagger (Ratinov and Roth, 2009).

Training data. As training set, we use the whole TAC-KBP 2015 training data and roughly one third of the Ontonotes. We use the Ontonotes' entity types corresponding to the TAC data (PER, ORG, FAC, LOC and GPE) plus the NORP type. Later, at the end of the linking phase, NORP mentions are assigned a TAC entity type, by mapping the DBpedia info of the selected entity to the five types of the task or, for NIL mentions, by setting the entity type to GPE.

We only focus on named entity mentions (NAM)

[1] https://tac.nist.gov/2016/KBP/
[2] http://www.summa-project.eu/
[3] http://www.cs.cmu.edu/~ark/TurboParser/

mentions, therefore we did not develop a strategy for detecting nominal (NOM) mentions.

Post-processing. As a post-processing step, we force to detect mentions that are marked in the text as being the authors of the articles, and we tag them with the PRE type.

We also apply a string matching procedure to capture mentions that were not recognized by the sequential model. In particular, we extract mentions with the exact same surface form as those previously detected in the document. These new mentions are then tagged with the types of the old ones, according to a voting procedure that is biased towards the PER label.

Later, at the end of the linking stage, some of the entity types are also reassigned in order to promote label agreement after both the co-reference and the linking steps (see details in section 2.3).

2.3 Linking System

The mentions detected in Section 2.2 are linked to database entries according to the strategy described in Algorithm 1.

Algorithm 1 Linking System

1: Simple string match coreference
2: Candidate generation
3: Candidate rank: NN-search + prior statistics
4: Re-rank (top 8 candidates) accounting for coherence
5: NIL detection
6: Cross-document coherence

Coreference. First, we perform a high-precision coreference step at the document level, by linking all the mentions whose surface forms are substrings of other mentions' forms. For preserving the agreement within the coreference clusters, some entity types are then heuristically reassigned with a voting strategy.

Candidates generation. For each mention, the candidates are generated using the less ambiguous mention (defined as the one with the largest span) in the corresponding coreference cluster. Then, the candidates generation itself is performed based on the probability of an entity given a mention, $p_{\text{wp\_wl\_conll\_TAC}}(e|m)$, computed from statistics of the anchors in the following datasets: Wikipedia, Wikilinks[4], AIDA-CoNLL2003[5] and TAC-KBP training data. In addition to that, and for mentions with fewer candidates (less than 60), we also consider as candidates the entities whose titles have all the words of the query mention.

Candidates rank. Step-3 of Algorithm 1 starts by ranking the candidates using a nearest-neighbours (NN) search criterion. To this end, a query feature vector q_i is built for each mention m_i from the body of the source document, considering lemmas, heads and root words. Then, a similarity search operation is executed on a search index with the Wikipedia entities indexed with their corresponding Wikipedia body and Wikilinks text. Considering c_{ik} as the k^{th} nearest-neighbour candidate of mention m_i, this operation approximately computes $s_{sim}(c_{ik}, m_i)$, which is the similarity between q_i and c_{ik} in the feature space, using a ranking function based on Okapi BM25.

Based on the search similarity $s_{sim}(c_{ik}, m_i)$, we compute a preliminary ranking score

$$s_0(c_{ik}, m_i) = s_{sim}(c_{ik}, m_i) \cdot (1 + s_{\text{wp\_wl\_conll}}(c_{ik})), \quad (1)$$

where $s_{\text{wp\_wl\_conll}}(c_{ik})$ is a score related with the likelihood of candidate c_{ik}, computed as follows:

$$\begin{aligned} s_{\text{wp\_wl\_conll}}(c_{ik}) = & k_1 \cdot \log p_{\text{Wikipedia}}(c_{ik}) \\ & + k_2 \cdot \log p_{\text{Wikilinks}}(c_{ik}) \\ & + k_3 \cdot \log p_{\text{CoNLL-03}}(c_{ik}), \end{aligned}$$

where $p_{\text{Wikipedia}}$, $p_{\text{Wikilinks}}$ and $p_{\text{CoNLL-03}}$ are the probabilities of candidate c_{ik} extracted from the statistics in Wikipedia, Wikilinks and AIDA-CoNLL2003 corpora, respectively; and k_1, k_2 and k_3 are tunable positive parameters.

Finally, based on score s_0, step-3 of Algorithm 1 sorts the candidates of each mention using the following ranking score:

[4] http://www.iesl.cs.umass.edu/data/wiki-links

[5] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/aida/downloads/

$$
\begin{aligned}
s_1(c_{ik}, m_i) =& s_0(c_{ik}, m_i) \\
&+k_4 \cdot p_{\text{TAC}}(c_{ik}|m_i) \\
&+k_5 \cdot p_{\text{TAC}}(c_{ik}|m_i) \cdot s_0(c_{ik}, m_i) \\
&+k_6 \cdot p_{\text{wp\_wl\_conll}}(c_{ik}|m_i) \\
&+k_7 \cdot p_{\text{wp\_wl\_conll}}(c_{ik}|m_i) \cdot s_0(c_{ik}, m_i),
\end{aligned}
$$

where c_{ik} is the k^{th} candidate of mention m_i; k_4, k_5, k_6 and k_7 are real valued positive parameters; $p_{\text{TAC}}(c_{ik}|m_i)$ is the conditional probability of candidate c_{ik} given its mention m_i, computed from the statistics in the TAC-KBP training dataset; and $p_{\text{wp\_wl\_conll}}(c_{ik}|m_i)$ is the same conditional probability computed from the concatenation of Wikipedia, Wikilinks and AIDA-CoNLL2003 corpora.

Re-rank for coherence State-of-the-art methods for entity linking use coherence models that favor solutions in which the entities within a same document are related with each other. The inference of a fully collective model, however, is NP hard (Kulkarni et al., 2009), since one must consider all possible combination of mentions candidates. To tackle this problem of complexity, prior work typically relax the general collective formulation either by using continuous formulations (Kulkarni et al., 2009) or by identifying sets of mentions or entities that are somehow involved in a semantic relation (Hoffart et al., 2011; Ratinov et al., 2011; Sil et al., 2015; Pan et al., 2015).

In step-4 of Algorithm 1, we focus on the top 8 candidates obtained in step-3 and re-rank them to favor coherence. In contrast to previous work, our coherence model resolves each mention independently. To achieve coherence, the score of a mention's candidate is influenced by its coherence with all the candidates of the other mentions in the text:

$$
s_2(c_{ik}, m_i) = s_1(c_{ik}, m_i) \cdot \Big(1 + \sum_{j \neq i, l} s_c(c_{ik}, c_{jl})\Big), \tag{2}
$$

where $s_c(s_{ik}, c_{jl})$ is a score that accounts for the coherence between the candidate under evaluation (s_{ik}) and the l^{th} candidate of other mention m_j (s_{jl}), and which is given by:

$$
s_c(c_{i,k}, c_{j,l}) = \begin{cases} \frac{1}{p_{ij}}, & c_{ik}, c_{jl} \text{ share a link} \\ \frac{1}{2p_{ij}}, & \text{otherwise,} \end{cases} \tag{3}
$$

where p_{ij} is the position of candidate c_{jl} according to the previous ranking score $s_1(c_{jl}, m_j)$. This coherence score was empirically designed to consider both coherence (as the existence or absence of a link) and information regarding previous candidate order.

Our coherence model, in (2), is similar to the model that was independently proposed by Globerson et al. (2016).

NIL detection For documents with at least 10 mentions, we accomplish NIL detection by verifying that a mention has no coherence (measured as the existence or absence of links) with any of the other mentions in the text. After that, some of the NILs are linked to database entries, depending on the links of other mentions in the same coreference cluster. This NIL detection is latter improved in the cross-document coherence step.

Cross-document coherence Finally, step-6 of Algorithm 1 builds on top of step-5 to promote a new type of coherence that works at a corpora level. The underlying idea of this step is to promote coherence along the entities that co-occurred (with the same mention+candidate pair) in different documents.

Let, for each mention m_i, $\mathcal{D}(m_i)$ be the set of the entities to which the other mentions in the document ($m_{j \neq i}$) link to (according to step-5). For each entity e_{ik} to which the surface of mention m_i links to in the full corpus, let $\mathcal{C}(e_{ik}, m_i)$ be the set of entities that co-occur in documents where the surface form of m_i connects to e_{ik}. We define the cross-document coherence score as

$$
s_3(e_{ik}, m_i) = J(\mathcal{D}(m_i), \mathcal{C}(e_{ik}, m_i)), \tag{4}
$$

where $J(.)$ is the Jaccard similarity:

$$
J(\mathcal{A}, \mathcal{B}) = \frac{\mathcal{A} \cup \mathcal{B}}{\mathcal{A} \cap \mathcal{B}}. \tag{5}
$$

Each mention m_i is finally linked to the entity, e_{ik*}, with the highest cross-document coherence score, in (4).

At the end of the linking system, we map the DBpedia labels of the selected entities to the five NER types of the task, and use them to reassign the types of the corresponding NORP mentions.

2.4 Future Directions

Our EDL system consists of several steps that were successfully engineered for the task, and whose parameters can be hand tuned. In the future, we expect to include machine learning in the EDL system. This would allow us to automatically learn the best configuration of parameters and to be able to easily use and test more features.

In a complementary line of research, we plan to use and develop new language-independent features in order to reach a final system which, in line with our *summa1* submission, would be suitable to process documents in different languages.

We also plan to improve our NER module, which appears to be an important bottleneck of the final EDL system.

3 Cold Start KBP

3.1 Motivation and system structure

Our motivation for the Cold Start KBP task was to test a hypothesis that the slot filling problem can be solved by general purpose semantic parsers without specific training data or parser customization. Due to our earlier experience with semantic parsing with the Abstract Meaning Representation (Banarescu et al., 2013) formalism, we apply the top performing AMR parser that we developed for Semeval-2016 competition (Barzdins and Gosko, 2016) and attempt to map its output to the slots specified in Cold Start KB construction task as described in Algorithm 2.

Algorithm 2 KBP slot filling

1: Preprocessing and sentence extraction
2: AMR parsing with a CAMR parser
3: Entity Detection and Linking system
4: Mapping the AMR concept instances to the EDL entities
5: Mapping the AMR predicates to appropriate slot fillers

3.2 Submissions

We submitted two runs obtained by an identical process but differing only in the set of EDL data used. The *summa_KB_ENG_1* run was obtained by using the EDL data from submission *IBM1*, which we believed to be a state of art EDL result from other competitors; and we built the *summa_KB_ENG_2* run from our own team *summa2* EDL submission described in section 2.1.

3.3 AMR parsing

AMR parsing is done by a customized version of the CAMR parser (Wang et al., 2015b; Wang et al., 2015a) as used in our earlier experiments on the Semeval challenges. No extra training data or additional tuning of the AMR layer were used for this task, as the goal was to evaluate potential applications of general purpose semantic parsing. AMR parsing was performed on a sentence-by-sentence level, with no intra-document coreference resolution. It was expected that integrating the EDL system results would link these references but this did not materialize (especially for nominal mentions and pronouns), resulting in significantly lowered accuracy. For future submissions this would be a key issue that needs a solution as rather frequently the required answer was not reached because this lack of intra-document linking of AMR nodes.

3.4 Entity mapping between AMR and EDL data

As the AMR annotation results in a very different set of entities than the EDL guidelines, the entity mapping is not trivial.

AMR entities The initial set of KBP entities is populated by the instances of AMR concept classes listed in Table 1. In most cases these entities are linked to a particular set of tokens, however that is not always true - often AMR identifies entities that have particular role in some predicate, but are not explicitly mentioned in that part of the sentence and would require a document level coreference resolution between AMR graphs of the document (like multi-sentence AMR construct).

Entity linking Linking of these entities with the appropriate entities identified by the EDL systems

is currently done based on boundary overlap - an exact boundary match is not required. In many cases no appropriate entities are found, so we insert new entities that were detected in semantic parsing but were not present in EDL data. Entities from EDL data that could not be linked to appropriate nodes in the AMR graphs were not included in the KBP submission under assumption, that they are not relevant to the relations in this slot filling task; therefore the recall measure of entities in the official scoring is low and reflects only the entities identified by the AMR parser.

AMR concept class	TAC entity type
person	PER
country	GPE
state	GPE
province	GPE
city	GPE
town	GPE
organization	ORG
religious-group	ORG
company	ORG
government-organization	ORG

Table 1: AMR entity mapping

3.5 Predicate mapping between AMR and KBP slots

The actual slot filling is performed by scanning AMR data for a specific subset of AMR concepts that 'trigger' one of the targeted slot filling sets. For this set of concepts we developed a heuristic transformation that scans surrounding nodes of the semantic graph and maps the identified AMR links to particular types of knowledge base slots. The actual mappings are illustrated in Table 6.

As the AMR parser model is generic and not adapted to the particular needs of TAC KBP slot filling task, some slots have no corresponding concepts in AMR data and thus cannot be filled by this approach. However, the more popular types of data such as employment and relationships have a good match between these systems.

It should be noted that the resulting mappings generally are 1-to-n, as the AMR predicates are n-ary relations (similar to the annotation concept used in the Event Nugget track) and can imply multiple different binary relations because both relationship directions need to be considered separately (e.g. the symmetric relationships of employee and employer) and also because the KBP slot filling annotation marks otherwise identical slots differently depending on the entity type.

The transformation process needs to consider additional information from the whole predicate. For example, a employment relationship between a person and a company may result in filling either the slot org:employees_or_members or org:top_members_employees, and the distinction can be made by considering the position label of that AMR predicate. In a similar manner, the predicate for personal relationships has a field (ARG2 in the annotation) describing the type of relation, so it can be transformed to the appropriate choice of KBP slot.

Certain slots can be filled by considering relations that in the AMR parse graph are syntactic predicates as opposed to semantic ones - for example, the residence slots often are described by having a country or location entity as possessive modifier of the person or company.

4 Results

4.1 Entity Discovery and Linking

NER evaluation. This section evaluates the impact of the NER post-processing step into the quality of the CRF model, using the TAC-KBP 2015 test set. Table 2 shows two official metrics for NER[6], computed for the output of the base NER (using a CRF model), after the coreference step of Section 2.3, and at the end of the linking system.

The mentions that were bootstrapped in the post-processing of Section 2.2 lead to a considerable increase (5.4%) of the NER F_1 score. This improvement was mainly (but not only) due to a strong increase in the recall. NERC measure suffered a considerable boost due to a better detection of mentions and a suitable reassignment of types based on the coreference clusters. This measure is also improved at the end of the system, when the NORP tags are reassigned based on the linking results. Overall, we got a positive impact of

[6] see (Ji et al., 2015) for details.

more than 5% in both NER measures, validating our system design options.

	NER	NERC
CRF model	73.8%	67.9%
+expansion+coreference	79.2%	72.2%
+NORP reassignment	**79.2**%	**73.9**%

Table 2: Impact, of NER post-processing steps, in both NER and NERC F_1 scores, using EN and NAM filters.

Step ablation. To evaluate the impact of each step of Algorithm 1 in the final linking system, Table 3 reports various metrics (with EN and NAM filters) at the end of steps 3 4 5 and 6. Each system step leads to cumulative improvements in the global measure NERLC, which accounts for mentions detection, type classification and linking. We also verify that, for all the measures, the final stage of the algorithm is the one with the highest F_1 value. For this outcome, we point out the final stage of cross-document coherence, which has a consistent positive effect into all of the metrics.

Steps 4 and 5 do not always lead to improvements, by themselves. In spite of that, we have experimentally verified that these steps have a final positive impact, even when a local evaluation may indicate them to be disadvantageous. One situation where it is easy to understand this effect, is the decrease of the NENC F_1 score after NIL detection. This decrease is mainly due to an increase in the number of the NIL mentions (lowering the precision), some of which are further relinked to the correct entity when accounting for cross-document coherence.

Regarding coreference evaluation, measure CEAFm suffers a considerable improvement in step-5, when, after detecting NIL mentions, some of them are resolved to entities based on the co-reference clusters. Finally, our cross-document step is also useful for coreference resolution.

System evaluation. Table 4 evaluates our system performance on TAC2016 test data, using the EN-NAM filter. Regarding mention detection, whose quality is reflected in metrics NER and NERC, we only scored 8^th out of 11 teams. Despite of starting with this large disadvantage, our scores increase considerably (improving three positions in the classification rank) when we account for the linking quality (see metrics NERLC, KBIDs and CEAFm). This fact indicates that we have a high performing linking system. To validate this intuition, we run our linking step on top of the mentions detected by the USTC system (Liu et al., 2016). USTC team achieved the highest scores in most of the metrics of the shared task, including those regarding mention detection. From this comparison (whose results are in the last columns of Table 4), we conclude that our linking system is on a pair with the best systems in the competition.

4.2 Cold Start KBP

The official results of KBP evaluation are shown on Table 5, ranking at 13^{th} place out of 19 teams. The low recall rate is rather disappointing, however, the error analysis indicates that this is largely caused by faults in the linking process between AMR graph nodes and EDL entities as discussed in 3.4. On the other hand, the system achieves good precision, so with appropriate fixes it could be competitive in the next iteration of TAC KBP.

5 Conclusions

This paper described the contribution of SUMMA submissions to the NIST TAC-KBP 2016. In this first year, we competed in the EDL and cold start KBP tracks.

Regarding the EDL track, our main submission was a rule-based system, whose steps were empirically validated. As main contribution to the track, we point out our coherence step that treats each mention independently and the impact of an original corpora-level coherence score, which favours agreement between bags-of-entities along a corpus. We also attempted to submit a language independent system to the EDL track, but we did not have time for making a final competitive submission.

Regarding cold start KBP, we establish a proof of concept that the KBP slot filling task may be approached by using general purpose semantic parsing models. While current results indicate a number of technical challenges in transformations between these very different semantic models, this

	Basic Rank (step-3)	Intra-Doc. Coherence (step-4)	NILL Detect. (step-5)	Cross-Doc. Coherence (step-6)
NERLC	61.1%	62.3%	62.4%	**64.7**%
NELC	61.6%	61.2%	62.3%	**63.9**%
NENC	59.8%	65.1%	64.7%	**66.7**%
KBIDs	68.4%	68.1%	70.4%	**70.8**%
CEAFm	58.2%	57.3%	69.7%	**71.7**%

Table 3: Evaluation, on the TAC-KBP 2015 test data, of our system at the end of steps 3, 4, 5 and 6. Each row shows F_1 scores for an official measure (Ji et al., 2015), computed using EN and MAN filters.

	Summa3	rank	USTC (2016)	*Summa3* (USTC mentions)
NER	83.1%	8^th	**90.6**%	**90.6**%
NERC	76.1%	8^th	**87.8**%	**87.8**%
NERLC	66.4%	5^th	79.2%	**79.8**%
KBIDs	70.8%	6^th	**81.1**%	81.0%
CEAFm	74.4%	5^th	83.2%	**83.3**%

Table 4: EDL evaluation on TAC-KBP 2016 test data, using EN-NAM filter. First column: our final system; second column: position of our system in the competition; third column: USTC system; last column: our linking system using USTC mentions.

	Prec	Recall	F1
LDC MAX 0 hop	45.0%	2.9%	5.5%
LDC MAX ALL	40.0%	2.2%	4.1%

Table 5: Accuracy of the resulting knowledge base SUMMA2 submission.

approach shows potential and we expect to provide a significantly improved implementation in the next issue of TAC KBP.

Acknowledgments

This work was mainly supported by the SUMMA project – this project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688139. This work was also supported by Fundação para a Ciência e Tecnologia (FCT) through contracts UID/EEA/50008/2013, GoLocal project (grant CMUPERI/TIC/0046/2014) and the Latvian state research programme SOPHIS (Project No. 2).

References

L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of Linguistic Annotation Workshop*.

G. Barzdins and D. Gosko. 2016. Riga at semeval-2016 task 8: Impact of smatch extensions and character-level neural translation on amr parsing accuracy. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval)*.

Daniel Ferreira, André Martins, and Mariana S. C. Almeida. 2016. Jointly learning to embed and predict with multiple languages. In *Annual Meeting of the Association for Computational Linguistics - ACL*, August.

Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2016. Collective entity resolution with multi-focal attention.

Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 2000. Large margin rank boundaries for ordinal regression.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing – EMNLP*, pages 782–792.

Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of tac-kbp2015 tri-lingual entity discovery and linking. In *Text Analysis Conference*.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th*

ACM SIGKDD international conference on Knowledge discovery and data mining, pages 457–466.

Dan Liu, Wei Lin, Shiliang Zhang, Si Wei, and Hui Jiang. 2016. The USTC NELSLIP Systems for Trilingual Entity Detection and Linking Tasks at TAC KBP 2016. In *Proc. Text Analysis Conference (TAC2016)*.

André F. T Martins, Miguel B. Almeida, and Noah A. Smith. 2013. Turning on the turbo: Fast third-order non-projective turbo parsers. In *Proc. of the Annual Meeting of the Association for Computational Linguistics*.

Xiaoman Pan, Taylor Cassidy, Ulf Hermjakob, Heng Ji, and Kevin Knight. 2015. Unsupervised entity linking with abstract meaning representation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics–Human Language Technologies*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning - CoNLL*, pages 147–155.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384.

Avirup Sil, Georgiana Dinu, and Radu Florian. 2015. The ibm systems for trilingual entity discovery and linking at tac 2015. In *Proc. Text Analysis Conference (TAC2015)*.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015a. Boosting transition-based amr parsing with refined actions and auxiliary analyzers. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers)*, pages 857–862.

Chuan Wang, Nianwen Xue, Sameer Pradhan, and Sameer Pradhan. 2015b. A transition-based algorithm for amr parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–375.

AMR concept	TAC KBP slot
have-org-role-91	per:title
have-org-role-91	per:employee_or_member_of
have-org-role-91	org:employees_or_members
have-org-role-91	gpe:employees_or_members
have-org-role-91	per:top_member_employee_of
have-org-role-91	org:top_members_employees
leader	per:top_member_employee_of
leader	org:top_members_employees
mod	per:countries_of_residence
mod	gpe:residents_of_country
mod	per:cities_of_residence
mod	gpe:residents_of_city
mod	per:statesorprovinces_of_residence
mod	gpe:residents_of_stateorprovince
mod	org:countries_of_headquarters
mod	gpe:headquarters_in_country
mod	org:cities_of_headquarters
mod	gpe:headquarters_in_city
mod	org:statesorprovinces_of_headquarters
mod	gpe:headquarters_in_stateorprovince
have-rel-role-91	per:spouse
have-rel-role-91	per:siblings
have-rel-role-91	per:children
have-rel-role-91	per:parents
have-rel-role-91	per:other_family
study-01	per:schools_attended
study-01	org:students
shareholder	per:holds_shares_in
shareholder	org:holds_shares_in
shareholder	org:shareholders
die-01	per:date_of_death
die-01	per:city_of_death
die-01	per:country_of_death
die-01	per:stateorprovince_of_death

Table 6: AMR predicate mapping