Proceedings of the Second International Workshop on Computational Linguistics for Uralic Languages

Editors:

Tommi A. Pirinen

Eszter Simon

Francis M. Tyers

Veronika Vincze

2016

Workshop Programme

10:00-10:15 Opening

10:15-11:00 Invited talk
András Kornai: *Computational linguistics of borderline vital languages in the Uralic family*

11:00-11:30 Poster boasters

11:30-12:00 Demo and poster session & coffee break
Jeremy Bradley: *Transcribe.mari-language.com: Automatic transcriptions and transliterations for Mari, Tatar, Russian, and more*
Lene Antonsen, Trond Trosterud, Marja-Liisa Olthuis and Erika Sarivaara: *Modelling the Inari Saami morphophonology as a finite state transducer*
Thierry Poibeau and Svetlana Toldova: *Exploring Natural Language Processing Methods for Finno-Ugric Languages*
Kristian Kankainen: *Demonstration of Minority Translate, a tool for making small Wikipedias bigger*
Tommi A Pirinen, Antonio Toral and Raphael Rubino: *Rule-Based and Statistical Morph Segments in English-to-Finnish SMT*
Axel Wisiorek and Zsófia Schön: *Obugric Database: Corpus and Lexicon Databases of Khanty and Mansi Dialects*

12:00-13:00 Session 1
Peter Smit, Juho Leinonen, Kristiina Jokinen, Mikko Kurimo: *Automatic Speech Recognition for Northern Sámi with comparison to other Uralic Languages*
Johannes Dellert: *Uralic and its Neighbors as a Test Case for a Lexical Flow Model of Language Contact*

13:00-14:00 Lunch

14:00-15:00 Session 2
Kadri Muischnek, Kaili Müürisep, Tiina Puolakainen and Krista Liin: *Parsing Estonian: Tools and Resources*
Francis Tyers, Tommi A Pirinen: *Intermediate representation in rule-based machine translation for the Uralic languages*

15:00-15:30 Demo and poster session & coffee break

15:30-16:30 Tutorial 1
Trond Trosterud: *Building grammatical analysers for Uralic languages*

16:30-17:30 Tutorial 2
Veronika Vincze, Filip Ginter, Tommi Pirinen, Francis Tyers: *Universal Dependencies for Finno-Ugric Languages*

17:30–18:30 SIGUR meeting & closing remarks

Programme Committee

Timofey Aleksandrovich Arkhangelsky, Natsional'ny Issledovadel'sky Universitet "Vysshaya shkola ekonomiki"
Lars Borin, Göteborgs universitet
János Csirik, Szegedi Tudományegyetem
Mark Fishel, Tartu ülikool
Mikel L. Forcada, Universitat d'Alacant
Mans Hulden, University of Colorado at Boulder
Heiki-Jaan Kaalep, Tartu ülikool
Tommi A. Pirinen, Dublin City University
Aarne Ranta, Chalmers tekniska högskola
Michael Rießler, University of Freiburg
Eszter Simon, MTA Nyelvtudományi Intézet
Trond Trosterud, UiT Norgga árktalaš universitehta
Francis M. Tyers, UiT Norgga árktalaš universitehta
Veronika Vincze, Szegedi Tudományegyetem

Workshop Organizers

Tommi A. Pirinen, Dublin City University
Francis M. Tyers, UiT Norgga árktalaš universitehta
Eszter Simon, Hungarian Academy of Sciences, Research Institute for Linguistics
Veronika Vincze, Hungarian Academy of Sciences, Research Group on Artificial Intelligence
Ágoston Nagy, University of Szeged
Csilla Horváth, University of Szeged

ISBN Number

ISBN 978-963-306-504-4

Table of Contents

Computational linguistics of borderline vital languages in the Uralic family

András Kornai
HAS Computer Science Research Institute
H-1111 Kende utca 13-17, Budapest, HUNGARY
andras@kornai.com

January 8, 2016

Abstract

In this survey we apply the methodology of [1] to the Uralic family with the specific goal of triage, to help the community decide where the effort is best placed. As in battlefield triage, where the relatively lightly wounded and the very heavily wounded are treated last, here we suggest to direct the very limited resources of the computational linguistics community towards the middle class of *borderline* languages where neither vital nor still/heritage status can be established. The talk will complement from the digital perspective the survey of [2].

Acknowledgments

We thank Marianne Bakró-Nagy (HAS Research Institute for Linguistics, Johanna Domokos (Universität Bielefeld), and Anna Fenyvesi for support and ideas. Katalin Pajkossy (Budapest University of Technology and Economics) was responsible for crawling the data, and her work is gratefully acknowledged here. The viewpoint expressed here is that of the author, as is the responsibility for all remaining errors.

References

[1] András Kornai. Digital language death. *PloS ONE*, 8(10):DOI 10.1371/journal.pone.0077056, 2013.

[2] Lyle Campbell and Bryn Hauk. Language endangerment and endangered Uralic languages. In Harri Mantila, Kaisa Leinonen, Sisko Brunni, Santeri Palviainen, and Jari Sivonen, editors, *Proc. Congressus Duodecimus Internationalis Fenno-Ugristarum*, pages 7–38. University of Oulu, Oulu, 2015.

Modelling the Inari Saami morphophonology as a finite state transducer

Lene Antonsen[1], Trond Trosterud[1], Marja-Liisa Olthuis[1] and Erika Sarivaara[2]

[1]Department of Linguistics, UiT The Arctic University of Norway
[2]Faculty of Education, University of Lapland

December 28, 2015

Abstract

The article presents a set of morphophonological problems coming up when making a transducer for Inari Saami, a language with a complex and not too well documented morphophonology. The sound alternations in the inflection system are governed by an intricate combination of phonological and morphophonological factors. Modelling the grammar as a finite state transducer gives more insight into the Inari Saami morphophonology, and the resulting program will be the foundation of all future Inari Saami language technology applications. The transducer compiles both with `xfst` and `hfst`.

1 Introduction

The article presents the morphophonological part of a morphological transducer for Inari Saami. The transducer consists of 28,000 stems, morphological inflection and compounding for all inflecting parts of speech, and a morphophonological component. The coverage is at present 92.0 % of the tokens in a 1.6M general corpus of Inari Saami text.

The motivation for the transducer was to provide a machine translation system from North to Inari Saami with a morphological generator. The transducer should thus meet the following demands: It must have good coverage, handling the bilingual dictionary of the MT system, and it must be able to generate one and only one form for every inflected, derived or composed wordform of the system.

The article is organised as follows: Section 2 presents the context for which the transducer was built, and gives an introduction to relevant parts of Inari Saami morphophonology. Section 3 presents an analysis of a set of suprasegmental morphphonological problems, first for consonant and then for vowel alternations. Finally comes a conclusion and some thoughts on future work.

2 Inari Saami

Inari Saami has a quite typical Uralic morphological structure, with nine cases for nouns and adjectives and with 3 x 3 person-number inflection and four moods for the verbs. The tense system is the one found in the Northern European Sprachbund (North Germanic, Baltic Finnic and Saami), with two tenses and participles for marking perfectivity. The negation verb is inflected for person, but not for tense. The orthography is phonological in nature, giving priority to represeting wordform rather than word structure, which, again, makes the treatment of inflection more complicated.

Some aspects of Inari Saami morphophonology are treated in [5]; for an overview on central inflectional paradigms, see [6]. The most comprehensive treatment of Inari Saami lexicon and grammar is the 3-volume dictionary [7].

3 Modelling morphophonology

3.1 Empirical foundation

There is no reference grammar for standard Inari Saami with the level of detail needed for building a comprehensive morphological model of the language. We thus had to make such a description.

We built an automated test suite, containing inflection paradigms for 660 nouns, 117 verbs and 39 adjectives, all in all 17,614 pairs of lemma + analysis and inflectional forms, in the form of test files in order to get an overview of the morphology, and to have regression testing. The testing was conducted in both

directions, ensuring that the FST both generated the intended wordforms from analyses, and gave the correct analysis for each wordform.

In addition, in the stem files we have 12,090 more nouns and 5,328 more verbs in base form, most of them taken from the Inari Saami-Finnish dictionary [8]. For many of the lemmas, this dictionary also contained information on inflectional forms, and for the verbs these were turned into a test set of 5,460 wordforms for 4,910 lemmas. The publicly available Inari Saami corpus[1] has been used for testing and lexicon completion, and has been of utmost importance for the development.

The transducer was built using the `twolc`[1] and `lexc` programming languages, as presented in e.g. [2]. For development, the code is compiled with the *Xerox* compilers[2], but for use in the actual machine translation engine, it is compiled with the Helsinki finite state transducer `hfst`[3]. The `lexc` files contain the stems and continuation lexica, where the lower level (intermediate "forms") contain stems, archiphonemes and affixes, which have to be combined with the `twolc` rules to give the final orthographic word form. There are three places to put complexity: in the stems, in the continuation lexica or in the `twolc` file.

`Twolc` models the morphophonology as a relation between two representations, or levels (hence the name *two-level compiler*). The rules and their contextual triggers are thus not ordered, but rather conditions for relations between the two levels. The competing `xfst` model, on the other hand, is a series of rewrite rules, feeding and bleeding each other. In this respect, `xfst` resembles classical generative phonology, but we still chose to use twolc, as we found that it would be too difficult to keep track of the rule ordering of such a complex morphophonology including long-distance assimilisation. Earlier treatments of consonant gradation in `twolc` for Saami languages include [3] and [4].

At present, the Inari Saami transducer provides at least one analysis for 92 % of the words in the reference corpus. Of the unanalysed words, 14 % gets an analysis from our Finnish analyser, mostly Finnish citations rather than loanwords, and 6 % are words that begin with capital letter, mostly proper nouns. Links to both the source code and yaml test files are available online[4].

[1] The Inari Saami corpus is accessible at `http://gtweb.uit.no/korp/`

[2] The compilers are available at `http://fsmbook.com`

[3] The hfst webpage is `http://hfst.sourceforge.net`

[4] `http://giellatekno.uit.no/doc/lang/smn/j-smn.html`, the article uses revision 124755

Trigger	changing	how	comments
^RLEN	vowel centre	lengthening	e.g. *a* to *aa*
^RVSH	vowel centre	shortening	e.g. *aa* to *a*
^VBACK*	vowel centre	quality	e.g. *ä* to *a*
^VHIGH*	vowel centre	quality	e.g. *á* to *i*
i2*	vowel centre	quality	e.g. *iä* to *e*
^CLEN	cons.centre	lengthening	e.g. *h* to *hh*
^WG	cons.centre	gradation	e.g. *tt* to *đ*
^SLEN	stem vowel	lengthening	e.g. *e* to *ee*
^SVSH	stem vowel	shortening	e.g. *ee* to *e*
^SVLOW	stem vowel	quality	e.g. *u* to *o*
^ÁI	stem vowel	quality	with ^EWG: *á* to *i*
^ÁE	stem vowel	quality	with ^EWG: *á* to *e*
u2*	stem vowel	quality	e.g. *u* to *o* and
	vowel centre	quality	*uá* to *oo*
^CSH	cons.centre	shortening	e.g. *tt* to *t* and
	vowel centre	shortening	*aa* to *a*
^EWG	cons.centre	gradation	e.g. *tt* to *đ* and
	stem vowel	quality	*á* to *i* or *e*
^EA	vowel centre	quality	*e* to *iä* and
	stem vowel	quality	*i* to *á*
^FCD	final consonant	deletion	e.g. delete *t*

Table 1: Triggers used to govern morphophonological changes. Adding an extra vowel or consonant to the word is done by combining the trigger and an archiphoneme in the stem. Triggers marked with * are used for verbs only.

3.2 Handling the complexity

In Inari Saami morphophonology it is especially complicated to handle alternations at the consonant centre, the vowel centre and the second syllable (first syllable nucleus and coda, respectively).

The `twolc` rules don't read the upper level of the morphological input, and will therefore function in the same way for all word classes. As the case and number morphology is shared across the nominal classes, the continuation lexica for nouns are reused for proper nouns, adjectives and numerals. New, unassimilated loanwords and foreign names inflect for the same morphological categories

as native words, but with fewer morphophonological alternations.

We have followed the principle that adding a new consonant or vowel to the word, is always done by means of an archiphoneme. Thus, the morphophonological rules never insert new segments from zero, as this would run the risk of an unbounded insertion of elements. The final realisation of an archiphoneme is governed by its context, usually specified by a specific trigger, see Table 1. The archiphonemes are: ^RV (in vowel centre), ^RC (in consonant centre), ^SV (in stem vowel), ^SC (in stem consonant), ^V (suffix vowel), cf. Tables 3 and 4.

We assigned each entry a lexc-stem with the non-reduced vowels and the consonant centre in the strong grade. This stem then formed the basis for all morphophonological alternations. In some cases, the combination of strong grade and full vowel is not attested simultaneously in the paradigm, and the lexc-stem will thus not correspond to an existing stem in the language.

Since at the outset we did not know the whole linguistic landscape, we added triggers to almost all alternations. As the `twolc` rule set grew, the triggers also made it easier to prevent conflicts. At the time being the `twolc` file consists of 106 rules with 274 contexts, the phonotactic division is presented in Table 2.

As we have much documentation in the yaml-files and stems-files, we can consider which triggers we could have done without. Making rules from the context without using triggers gives us more insight into the Inari Saami morphophonology.

There are 109 continuation lexica for nouns and 63 verb lexica. This is comparable to a similar transducer for Lule Saami (119 and 50 lexica), but far less than for a similar transducer for North Saami, which has been worked on for several years, and is in use for a spellchecker as well. We expect the number of lexica to shrink to some extent with a more generalised morphological analysis, but to grow as we take more marginal inflectional patterns and derivations into account.

vowel centre	**consonant centre**	**stem vowel**	**stem consonant**	**suffix**	**adding hyphen**
154	75	27	10	7	1

Table 2: The phonotactic division of the 274 contexts in the 106 rules in the `smn-phon.twolc-file`.

3.3 Processes in the consonant centre

In this section we present some of the most important alternations in the consonant centre, and how we model them.

3.3.1 ^WG – consonant gradation weak grade

Consonant gradation is the most central morphophonological alternation in Saami grammar. The consonant centre in the foot preceding the suffix boundary is realised in one out of two or three forms in a doublet or triplet, as shown in the columns for nominative (strong grade) and genitive (weak grade) in table 3[5]. This alternation runs through the whole inflectional system, and influences surrounding vowels as well.

SgNom	**SgGen** ^WG	**SgIll** ^CSH	**SgCom** ^WG^CLEN	**Essive** ^CLEN	`lexc-` stem	**trans lation**
päkki	*päähi*	*páákán*	*pahhijn*	*päkkin*	`pä^RVkk4i`	'bur'
mecci	*meeci*	*miäcán*	*meccijn*	*meccin*	`me^RVcci`	'terrain'
luákká	*luáhá*	*luákán*	*luáhháin*	*luákkán*	`luákk4á`	'class'
kukká	*kuká*	*kuukán*	*kukkáin*	*kukkán*	`ku^RVkká`	'flower'
sukká	*suhá*	*suukán*	*suhháin*	*sukkán*	`su^RVkk4á`	'sock'
fáddá	*fáádá*	*fáádán*	*fáddáin*	*fáddán*	`fá^RVddá`	'topic'
kissá	*kisá*	*kiisán*	*kissáin*	*kissán*	`ki^RVssá`	'cat'
eeči	*eeji*	*iäčán*	*eijijn*	*eeččin*	`eeč^RCi`	'father'

Table 3: Examples showing how the triggers work on the `lexc`-stems for bisyllabic nouns, and also the interaction between the triggers.

The `lexc`-stems (the form to which all morphological material is added) is always given in the strong grade. Weak grade forms are generated with a ^WG trigger, invoking a consonant gradation rule in `twolc`. Some alternating patterns are ambiguous, like *kk:h* vs. *kk:k*, as shown for *päkki* and *kukká* in table 3. In such cases we leave the default alternation unmarked, and mark the special case, here with *k4*, thus getting two alternations. The `twolc`-rule changes *k4* to *h* if it is follwed by a vowel and the ^WG trigger:

```
k4:h ⇔ Vow: k: _ Vow: ^WG: ;
```

Another rule removes *k* in the same context.

[5]The consonant series are presented in [9]. For a phonological analysis, see [10]

3.3.2 ^CLEN – consonant lengthening

^CLEN can be used alone or in combination with other triggers. When it appears alone, it lengthens a single consonant in strong grade, like *tiätu* 'knowledge' > *tiättun* (essive). Weak grade may cooccur with lengthening of the consonant centre, ^WG^CLEN. This will give different outcomes for quantitative and qualitative consonant gradation. The quantitative gradation is excemplified by *kissá*, and the qualitative by *päkki*, in table 3. Since we have quantitative consonant gradation (*ss:s*) for *kissá*, lengthening the weak grade (*s:ss*) will give a form identical to the strong grade, whereas the qualitative gradation *kk:h* for *sukká* gives a gemination *h:hh* different from the strong grade. Both types are undergoing the same processes in the transducer, though. In addition to modelling the linguistic process as such, this analysis also makes it possible to assign the different stem types *sukká, kissá* to the same continuation lexicon.

3.3.3 ^CSH – consonant shortening, interacting with vowel length

The trigger ^CSH shortens a long central consonant. For quantitative change, the ^WG and ^CSH triggers thus have the same effect.

In qualitative change, on the other side, the two triggers have different effect, as ^WG gives rise to change, whereas ^CSH does not. Cf. *sukka* in table 3, where *suuha* is qualitative consonant gradation and *suukán* is consonant shortening. For *kukká*, the distinction is not visible.

The trigger ^CSH was introduced when working with trisyllabic nouns and was used for shortening both the consonant centre and the vowel centre. In bisyllabic nouns this trigger is used for singular illative and it always appears together with ^RLEN, which instead lengthens the the vowel centre. ^CSH thus plays a different role for bi- and trisyllabic stems. Methodologically, the best approach would have been to start out with triggers having only one function, and then possibly unify cooccurring triggers at the end of the analysis.

3.4 Vowel centre and stem vowel interaction

Some of the vowel centre alternations are governed by different triggers, like ^RLEN (vowel centre lengthening) and ^RVSH (vowel centre shortening). Table 4 contains the examples handled in this section.

3.4.1 ^EA – alternations in bisyllabic nouns

The suffix for singular illative for bisyllabic and trisyllabic nouns is *n*. In a bisyllabic noun, the stem vowels *i* and *e* will both change to *á*. The exception is when the vowel centre is *a*, in which case the stem vowel changes to *a*, like for *saje:sajan* (see table 4 for translation of the example words). In cases where the stem vowel is *e*, and the consonant centre consists of two consonants, the stem vowel alternation conditions vowel centre lengthening, like *alge:aalgan*. Otherwise the stem vowel alternation conditions vowel centre change according to the following pattern: *ä:áá ää:áá ee:iä e:iä ie:iä ye:uá*, as is seen for the words *eečči:iäčán, äšši:ääšán*. This metaphonical (sound) alternation is in our system governed by the trigger ^EA, which is given to all continuation lexica for bisyllabic nouns. The only exceptions are the lexica for newer loanwords, which do not follow this pattern.

The consonant centre in the singular illative form is shortened by the trigger ^CSH, as explained in section 3.3.3, but the vowel centre shortening governed by this trigger is blocked by ^EA.

After having built all the continuation lexica for both verbs and nouns, it turned out that this vowel alternation could be governed by the linguistic context without triggers. The suffix *n* is otherwise used as suffix for essive, but by combining the suffix with the ^CSH-trigger, the condition will not be fulfilled in essive, and the alternation will not be realised. The context will not occur for verbs either. In this case the ^EA trigger blocks the phonological rule, and it would be better to formalise the metaphonical alternation as a general rule, and add a blocker for loanwords, which do not follow this pattern. This requires adding ^CSH also to continuation lexica for words where the consonant centre consists of a single consonant or a consonant cluster, which never can be shortened.

3.4.2 ^EWG, ^ÁI, ^ÁE – alternations in trisyllabic nouns

For trisyllabic nouns the singular genitive form is used as stem in the `lexc`-file as this is the form where we find the strong grade, and the form to which most suffixes are added. The singular nominative form of many of these nouns have both weak grade and vowel alternations, often both for vowel centre and stem vowel. The stem vowel *á* can change to *i* or *e*, both for nominative, essive and partitive forms. The triggers ^ÁE and ^ÁI are added to these stems, and the stem vowel changes to *i* or *e* accordingly. Words with *á:i* alternations are e.g. *eebir, kyevtis* (see table 4 for translation of example words), their genitive

SgNom	SgGen	SgIll	Essive	lexc	transl.
alge	*alge*	*aalgan*	*algen*	`aˆRVlge`	‘boy’
saje	*saje*	*sajan*	*sajeen*	`saˆRVjˆRCe`	‘place’
äšši	*ääši*	*ääšán*	*äššin*	`äˆRVšši`	‘issue, case’
eebir	*iäbbár*	*iäbbárân*	*ebirin*	`iäbbárˆÁI`	‘bucket’
kyevtis	*kuáhtás*	*kuáhtásân*	*kyevtisin*	`kuáhtásˆÁI`	‘duo’
ores	*orrääs*	*orásân*	*oresin*	`orrääsˆÁE`	‘male’
lyeme	*luámmán*	*luámánân*	*lyemeen*	`luámmáˆSVnˆÁE`	‘cloudberry’
riegis	*riäggás*	*riäggás*	*riegisin*	`ri5äggás`	‘circle, wheel’
uápis	*uáppás*	*uáppásân*	*uáppásin*	`u5áppásˆÁI`	‘acquaintance’

Table 4: Examples of vowel alternations for singular nominative, genitive, illative and essive, for the words used as examples in Chapter 3.4.1 and 3.4.2. The word *riegis* has a contracted stem, and therefore not the illative suffix *n*.

and essive forms are *iäbbár, ebirin, suáđđás, suáđisin*, and the `lexc`-stems are *iäbbárˆÁI, kuáhtásˆÁI*. Examples with *á:e* alternations are *ores, lyeme*, with genitive and essive forms *orrääs, oresin, luámmán, lyemeen*. The `lexc`-stems are *orrääsˆÁE, luámmáˆSVnˆÁE*. The vowel change is triggered by ˆEWG in the affix file, which also triggers the weak grade for the consonant centre, like in *iäbbár:eebir*.

By looking beyond the triggers, one finds a contextual pattern. A long vowel centre is connected to the *á:i*-change in the stem vowel, and a short vowel centre is connected to the stem vowel change *á:e*. These alternations could thus be triggered by ˆWG instead of the special trigger ˆEWG[6]. A diphthong can be long, like in *kyevtis*, or it can be short, like in *lyeme*. The problem is that the orthography does not distinguish between short and long diphthongs. But this is marked in the dictionary ([8]), and this information could be marked on the stems to be used as context for the rules[7].

There are a handful of trisyllabic and contracted nouns within the described pattern which do not have any alternations at all, and they could be marked in

[6]This has been done from revision 125169 of the Inari Saami transducer

[7]The so-called dictionary orthography (used in the dictionary for writing the lemmas) differs from the ordinary orthography (used in all other writing) in distinguishing between short and long diphthongs and short and half-long vowels (shortness marked with apostrophe), and between short and half-long consonants, where half-long consonants are marked with a dot under the consonant. Cf. pairs such as *alme/a'lme* and *njuṇe/njune*.

the stem to avoid the alternations.

Some stem diphthongs are ambiguous: *iä* can be realised as *ee* like in *iäbbár:eebir* and as *ie* in *riäggás:riegis*. Also the diphthong *uá* has two realisations in nominative; it can be changed to *ye* as in *luámmán:lyeme* or to *uá* in *uállás:uáles*. For these words the stem in `lexc` has archiphonemes like *i5, u5*, so they will have other alternations, *ri5äggás, u5állás*. The letter *i* can act both as a vowel and as a consonant, as a part of the consonant centre. In the latter case it is underlyingly marked as *i4* (*päi4kk4i*). Altogether there are 30 different archiphonemes and marked vowels and consonants.

4 Conclusion

The Inari Saami morphophonology is not well documented. For an FST we have to model all words in the language, and not only prototypical example cases. We have done that, and the result is a model of the morphophonology, with 109 continuation lexica for nouns and 63 for verbs, including 17 different contextual triggers, 106 `twolc` rules with 274 contexts, and 30 consonant and vowel archiphonemes.

In hindsight, we see that many construction-specific triggers may be generalised across different patterns. But now, when we have more documentation in the test files and stems-files, we can look at which triggers we could have done without. Making rules from the context, and trying them out, gives us more insight into the Inari Saami morphophonology.

The resulting transducer is put into use as a generator for machine translation. Beyond that, it may also be used for other purposes, such as corpus analysis (a first test has already been conducted), e-dictionaries, spell checkers and pedagogical programs.

This we leave for the future.

Acknowledgments

Thanks to our collegue Miina Seurujärvi and Francis M. Tyers for participating in the FST work. Ilmari Mattus has kindly collected texts for the text corpus, and Ciprian Gerstenberger has processed them and put them online. The Finnish Saami Parliament has funded and coordinated the e-dictionary project. Our special thank goes to Hannu Kangasniemi for supporting the Aanaar Saami language technology project.

References

[1] Kimmo Koskenniemi. *Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production.* Publications of the Department of General Linguistics, University of Helsinki, No. 11, Helsinki, 1983.

[2] Kenneth R. Beesley and Lauri Karttunen. *Finite State Morphology.* CSLI publications in Computational Linguistics, USA, 2003.

[3] Trond Trosterud and Heli Uibo. Consonant gradation in Estonian and Sami: Two-level solution. In Antti Arppe et al., editor, *Inquiries into Words, Constraints and Contexts*, pages 136—150. CSLI Studies in Computational Linguistics online, 2005.

[4] Sjur N. Moshagen, Pekka Sammallahti, and Trond Trosterud. *Twol at work*, pages 94—105. CSLI, Stanford, California:, 2004.

[5] Matti Morottaja. *Anarâškielâ ravvuuh.* Kotimaisten kielten tutkimuskeskus, Helsinki, 2007.

[6] Marja-Liisa Olthuis. *Inarinsaamen kielen kielioppi.* Sämitigge, Inari, 2000.

[7] Erkki Itkonen, Raija Bartens, and Lea Laitinen. *Inarilappisches Wörterbuch*, volume 1–4 of *Lexica Societatis Fenno-Ugricae.* Société Finno-Ougrienne, Helsinki, 1988.

[8] Marja-Liisa Olthuis and Taarna Valtonen. *Säämi-suomâ-säämi sänikirje.* Sämitigge, Inari, 2000.

[9] Pekka Sammallahti and Matti Morottaja. *Säämi-suomâ sänikirje.* Girjegiisá, Utsjoki, 1993.

[10] Patrik Bye. *Grade alternation in Inari Saami and abstract declarative phonology*, pages 53—90. John Benjamins, Amsterdam, 2007.

Transcribe.mari-language.com: Automatic transcriptions and transliterations for Mari, Tatar, Russian, and more

Jeremy Bradley
Ludwig Maximilian University of Munich
Institute for Finno-Ugric and Uralic Studies
&
Koneen Säätiö
J.Bradley@lmu.de

December 31, 2015

Abstract

The aim of this paper is to introduce my efforts to create server-sided (i.e. platform independent web-based, from a user's perspective) automatic transcription and transliteration software for Uralic and non-Uralic languages. For four literary standards – Meadow Mari, Hill Mari, Russian, and Tatar – an operational interface can be found at `transcribe.mari-language.com` and the source code at `source.mari-language.com`. For other languages, software is under development. This paper details many of the fine aspects of writing systems used for (Meadow) Mari that I had to take into consideration when creating transcription mechanisms for that language.

1 Structure of this paper

Section 2 describes the circumstances that motivated the creation of the transcription/transliteration infrastructure presented in this paper. Section 3 describes how the software "normalizes" inputs: ads diacritic symbols users might not have on their

keyboard, etc. Sections 4– 6 introduce the transcriptions between the relevant writing systems for (Meadow) Mari that my software is capable of handling: Cyrillic (regardless of language) ⟷ ISO 9:1995, Meadow Mari Cyrillic (including historical, defunct orthographies) ⟷ UPA, UPA ⟷ IPA, respectively. (All other transcriptions/transliterations can be achieved by stringing these mechanisms together: for example, ISO 9:1995 can be converted into IPA by a transliteration into Cyrillic, a transcription from Cyrillic into UPA, and a transcription from UPA into IPA). I cannot give a comprehensive overview of all the transformation mechanisms within the limited scope of this paper, but can only provide a brief illustration of some of the more difficult aspects of creating software of this kind. Extensive documentation can be found on the site where this software is found, `transcribe.mari-language.com`.

Finally, Section 7 discusses the prospect of creating tools equivalent to those described in the previous sections for other languages. This has already been done for Hill Mari, Russian, and Tatar, but the limited nature of this paper does not allow me to describe the mechanisms implemented for those literary standards.

2 Why do we need web-based transcription software?

When dealing with languages of the Russian Federation, the choice of a writing system or transcription can be daunting or even politically charged. (Turkic) Tatar can serve as an anecdotal example of a language with a complex past (and present): literary Tatar used the Arabic script until 1927, then Latin-based orthographies until 1939, and since that time the Cyrillic alphabet [1, p.285]. Post-Soviet attempts to reintroduce a Latin-based orthography were rendered moot by a 2002 decision of the Russian constitutional court declaring that all state languages of the Russian Federation must be written in the Cyrillic alphabet [2, p.2].

The situation with regard to (Uralic) Mari is a bit less complex, but not greatly so. Mari literacy traces its roots back to the first Mari grammar, published in 1775 (an extensively annotated facsimile edition of which was published in 1956 [3]); from then until the present day Mari orthographies have predominantly used the Cyrillic alphabet. There are two literary norms of Mari that continue to be actively used, Meadow Mari and Hill Mari. Recent orthographic dictionaries demarcating the rules of the literary standard are available for both Meadow Mari [4] and Hill Mari [5]. However, great differences exist between the contemporary literary norms and historical orthographies used in numerous resources. Uralic sources traditionally use the so-called Finno-Ugric Transcription (or UPA – Uralic Phonetic Alphabet) presented in 1901 by Eemil Nestor Setälä [6], with a number of relatively recent high-impact publications (e.g. [7] [8] [9]) establishing what one might consider an unofficial standard

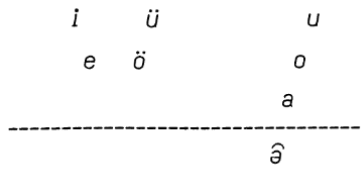

Figure 1: Meadow Mari Vowels [7, p.15]

for Latin transcription of Mari. However, competent transcription from Cyrillic into UPA (and vice versa) requires good knowledge of the idiosyncratic aspects of the Mari Cyrillic orthography.

Non-Uralic publications might ask contributors to use the ISO 9:1995 transliteration standard[1], or the International Phonetic Alphabet (IPA)[2]. Whereas an ISO 9 transliteration of literary (Cyrillic) Mari is straightforward and trivial for computer-literate scholars (albeit potentially time-consuming, as online applications for the ISO-9 transliteration of Cyrillic texts[3] cannot handle the additional characters found in Mari orthographies that are not part of the Russian alphabet: *ӓ*, *ҥ*,*ӧ*, *ӱ*, *ӹ*), using IPA for Mari is not. I am not aware of any publications other than my own [10] [11] that use IPA for Mari, and deriving IPA from UPA can be challenging due both to the fact that UPA is not as stringently standardized as IPA is and to a lack of information on the exact pronunciation of sounds in relevant sources. For example, Alho Alhoniemi's Finnish-language grammar of Mari [7], which thanks to its German translation [12] is still the most extensive and modern resource on Mari grammar at least marginally accessible to the international linguistic community, introduces the system of Meadow Mari vowels as seen in Figure 1.

The sound */ə̂/* is especially challenging here. Alhoniemi's graphic representation, which resembles the vowel trapezium, does not give detailed information concerning either the exact position of the vowel or rounding. According to Pekka Sammallahti, UPA */ə̂/* is a reduced mid central unrounded vowel [13, p.174] (*/ə/* in IPA), but even an inspection by ear casts that classification into doubt. My work group rather identified the sound as a close-mid back unrounded vowel (*/ɤ/* in IPA), and we marked it as such in our materials [14] [11].

In summary, there are numerous writing systems that scholars dealing with Mari

[1] `www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=3589`
[2] `https://www.internationalphoneticassociation.org/`
[3] e.g. `translit.cc`

might encounter and in which they might be expected to be able to produce texts. Transcriptions from some systems into others might be more or less straightforward from a technical standpoint (e.g. UPA ⟷ IPA), but require a very good understanding of the language (which general Uralicists or typologists dealing with Mari superficially might not have). Transliteration between Cyrillic and ISO 9:1995 is, technically speaking, absolutely trivial, but time-consuming for scholars not capable of writing their own transliteration scripts. Motivated by these circumstances, I have created a web-based interface that for four languages (or language standards) with which I am familiar – Meadow Mari, Hill Mari, Tatar, Russian – allows the transcription or transliteration of text from all relevant writing systems that I know of into (almost) all other writing systems, as accurately as the orthographies and transcription systems in question allow. An operational interface can be found at `transcribe.mari-language.com` and the PHP source code at `source.mari-language.com`. The relevant procedures can be found in the file *functions.php*; the user interfaces can be found in the files *transcription-general.php*, *transcription-specific.php*, and *transcription-universal.php*. Where relevant, sectiosn of this paper include a footnote containing the name(s) of the function(s) in *functions.php* carrying out the operations detailed in it.

By integrating Mari transcription mechanisms into our work group's electronic Mari-English Dictionary [11], which was compiled using contemporary Cyrillic orthography (with additional annotation compensating for defects in the orthography), it became usable in UPA and IPA, depending on a scholar's needs. Moreover, the dictionary's interface allows entries to be displayed using reverse sorting, i.e. sorted right-to-left, starting with the last letter of the word, then the penultimate letter, etc. This is especially useful due to the fact that the same vowel sound is indicated by different Cyrillic characters depending on its environment (UPA/IPA */a/* ⟷ Cyrillic *<а>*, *<я>*; UPA/IPA */e/* ⟷ Cyrillic *<э>*, *<е>*; UPA/IPA */u/* ⟷ Cyrillic *<у>*, *<ю>*) – a reverse-sorted list of lexemes is more useful in UPA or IPA than it is in the Cyrillic orthography we used when creating our dictionary.

For other languages of the Russian Federation, I have yet to develop fully functional interfaces of this type, but am hopeful that I will have the opportunity to do so in the future – although I would need input from competent scholars of the respective languages in order to implement equivalent software.

3 Orthographic Normalization

All software tools found on our website should be usable using an arbitrary Cyrillic (e.g. Russian) and Latin (e.g. English) keyboard layouts – i.e. keyboards layouts that only contain the 26 letters of the basic Latin alphabet (and punctuation marks,

numbers, etc.), and keyboard layouts that cover all letters used by the Russian Cyrillic alphabet, but not the additional Mari characters *ӓ*, *ҥ*,*ӧ*, *ӱ*, and *ӹ*. To facilitate this, the software includes a number of mechanisms allowing orthographic normalization, for both Cyrillic and Latin inputs. Users can access these by setting the same writing system as the input and the output in the user interface – "Cyrillic to Cyrillic", etc. These same normalization procedures are also carried out on inputs if other options are chosen – e.g. if users ask the software to transcribe Cyrillic to IPA, the input is subjected to the orthographic normalization procedures illustrated here.

3.1 Cyrillic

The strategies used by the software to normalize Cyrillic input are based on strategies used by Mari native speakers in colloquial contexts (e.g. in e-mails, on social network sites). To indicate a special Mari character, users either place a colon *:* after the letter from which it is derived (i.e. *а:* ⟶ *ӓ*, *н:* ⟶ *ҥ*, *о:* ⟶ *ӧ*, *у:* ⟶ *ӱ*, *ы:* ⟶ *ӹ*), or by capitalizing the letter from which the special character is derived inside a word (e.g. *шУм* ⟶ *шӱм /šüm/* 'heart')[4].

3.2 UPA

In Latin-based UPA inputs, users can place a colon *:* after a letter to create UPA-characters that are not part of the basic Latin alphabet, or can use a number of digraphs. In some cases, simple letters can be used to produce UPA symbols, as these simple letters (*y*, *q*, *h*) have no UPA value of their own. Table 1 gives an overview of normalization procedures.

If users wish to prevent two letters from being read as a digraph, they can place a vertical bar | between the two words: The input *sheme* produces the output *sheme*'black', while the input *s|heme* produces the output *sχeme* 'diagram' (a Russian loan word - the sound *χ* is not found in indigenous Mari vocabulary)[5].

4 Cyrillic ⟷ ISO 9:1995

ISO 9:1995[6] is a transliteration system: there is a deterministic 1:1 relationship between Cyrillic characters and Latin characters (e.g. Cyrillic *э* ⟷ ISO 9 *è*); the transliteration occurs completely independent of the pronunciation rules of the language(s)

[4] function cyrprep
[5] function latprep
[6] `www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=3589`

Input(s)	Output
a:, æ	ä
o:, ø	ö
u:	ü
y, õ	ə̂
y:, q	ə
z, zh	ž
s, sh	š
c, ch	č
n, ng	ŋ
h, x	χ

Table 1: Orthographic normalization of UPA inputs

in question. As such, an ISO 9:1995 transliteration can be realized by simply replacing Cyrillic characters with the corresponding Latin characters. Due to the simplicity and language-independence of the task, I have expanded the function responsible for transliteration between Cyrillic and ISO 9:1995 to also cover special characters found in other Uralic and non-Uralic languages of Russia (e.g. Udmurt *ӟ* ⟷ ISO 9 *z̈*)[7].

5 Meadow Mari Cyrillic Orthographies ⟷ UPA

Alho Alhoniemi's grammar of Mari gives a good overview of the relationship between the modern Meadow Mari Cyrillic Alphabet and UPA [7, p.28-29]; I. G. Ivanov's handbook on the phonetics of contemporary Mari [15] provides detailed accounts of the exact pronunciation of individual sounds. While the orthography is mostly straightforward and there is a 1:1 relationship between many consonant symbols and consonant sounds (e.g. Cyrillic <*ш*> ⟷ UPA /*š*/), there are a number of difficult aspects (where programming a transcription script is not trivial), and also some actual defects (where programming a fully automatic and accurate transcription script is not possible unless the user gives disambiguating information)[8].

[7] function cyr_to_iso
[8] function cyr_to_lat

	/C_/	/C'_/	/j_/	/#_/	/V_/	/Cj_/	/C'j_/
/a/	<Са>	<Ся>	<я>	<а>	<Va>	<Съя>	<Сья>
/u/	<Су>	<Сю>	<ю>	<у>	<Vу>	<Съю>	<Сью>
/e/	<Се>	<Се>	<е>	<э>	<Vэ>	<Съе>	<Сье>
/i/	<Си>	<Си>	<и>	<и>	<Vи>	-	-
/o/	<Со>	<Сьо>	<йо>	<о>	<Vо>	<Сйо>	<Сьйо>
/ö/	<Сö>	<Сьö>	<йö>	<ö>	<Vö>	<Сйö>	-
/ə̑/	<Сы>	<Сьы>	<йы>	<ы>	<Vы>	<Сйы>	-
/ü/	<Сÿ>	-	<йÿ>	<ÿ>	<Vÿ>	<Сйÿ>	-

Table 2: Vowel signs, palatalness, and the phoneme */j/*

5.1 Vowel signs, palatalness, */j/*

The most critical aspect is the usage of different vowel signs to indicate palatalness and the phoneme */j/*. The realization of vowel sounds in Mari orthography differs depending on their position in a word; the marking of palatalness (there is a phonological distinction */n/* */ń/* and */l/* */l'/* in Mari, and there are numerous other distinctions in Russian loan words) and the phoneme */j/* depends on the vowel sound, if any, following the consonant. Table 2 gives an overview of how the eight vowel sounds of Mari are realized in orthography, depending on whether they follow a non-palatal consonant */C/*, a palatal consonant */C'/* (either */ń/* or */l'/*), the sound */j/*, if they are in the initial position, etc. Note that some of these combinations are rarely encountered or only occur in compounds and/or Russian loan words.

The grey cells in Table 2 are especially problematic: before the vowels */e/* and */i/*, there is no orthographic distinction between palatal consonants and their non-palatal counterparts. Modern Mari orthography does not distinguish between */le/* and */l'e/*, for example, and homographs that are not homophones (i.e. words that are spelled, but not pronounced, in the same way) can be found. For example *<неле>*: */nele/* "difficult" */nel'e/* "(s)he swallowed". In these cases, users must manually indicate an orthographically unmarked palatalness with an apostrophe (i.e. *<л'е>* ⟶ */l'e/*) to get a correct transcription. We indicated palatalness by these means in our Mari-English dictionary [11].

5.2 Orthographically unmarked features

Whereas palatalness, discussed above, is sometimes marked in orthography and sometimes not, there are a number of processes and features that are systematically not

marked in the contemporary orthography (presented here without full historical explanations for the phenomena involved):

- The letter <ð>, while historically generally pronounced as UPA /δ/ / IPA /ð/ and today generally pronounced as /d/ (e.g. <киде̲м> /kide̲m/ “my hand, my arm”), is pronounced as /t/ in syllable-final position (e.g. <кид̲> /kit̲/ “hand, arm”) [7, pp.33-34].
- The letters <ð> and <г>, which have the prototypical values /d/ and /g/ (historically UPA /δ/ / IPA /ð/ and UPA /ɣ/ / IPA //), are pronounced as /t/ and /k/ respectively after voiceless obstruents. For example, the negative gerund in /-de/ /-te/ [7, pp.144-146] (orthographically always <-ðe>): <мол-> /tol-/ “to come” ⟶ <молде̲> /tolde̲/ “without coming”, but <поч-> /poč́/ “to open” ⟶ <почде̲> /poč́te̲/ “without opening” [7, pp.33-34]. This process occurs across orthographic word boundaries, e.g. the postposition /gə̑č́/ /kə̑č́/ “from” (orthographically always <гыч>): <ола г̲ыч> /ola g̲ə̑č́/ “from town”, but <мут г̲ыч> /mut k̲ə̑č́/ “from a word” [15, p.90].
- A number of consonant clusters are pronounced in manners that diverge from their orthographic realization, thanks to assimilation [15, pp.99-105]: <ж̲т> /š̲t/, <з̲т> /s̲t/, <ж̲ш> /š̲š/, <з̲ш> /s̲š/, <в̲к> /p̲k/, <г̲к> /k̲k/, <н̲к> /ŋ̲g/, <н̲г> /ŋ̲g/, <н̲ч> /ń̲č́/.
- Orthographically unmarked word stress tends to fall on the last full vowel of Mari words [7, p.17], where a full vowel is anything but the reduced vowel /ə̑/, and final unstressed /e/, /o/, and /ö/ [7, cf. pp.20-21; 39-40]. However, /e/, /o/, and /ö/ can occur as stressed full vowels in the final position, and there are examples where words are spelled the same, but are pronounced differently: <шерге>: /še•rge/ “expensive” /šerge•/ “comb”. That is to say, stress is a phonologically relevant feature that is not orthographically marked. It is usually, but not always, predictable; it is in cases where it is unpredictable that it might be phonologically relevant.
- Final unstressed /e/, /o/, and /ö/ are slightly reduced [15, pp.58-59], e.g. <йылме̲> /jə̑•lme̲ə̯̂/ “tongue; language”, <умо̲> /tu•mo̲ə̯̂/ “oak tree” <шӱдӧ̲> /šü•dö̲ə̯̂/ “hundred”
- More recent Russian loan words, and Russian names in particular, might be pronounced in accordance with Russian, rather than Mari, pronunciation rules.

With many of these features, it is questionable whether or not automatic transcription software should take them into consideration, even if it would be possible

for such a system to handle them. They would make back-transformation more difficult, and the orthography can in some cases have a disambiguating function. There are words that are pronounced the same due to the rules detailed above, but are not spelled the same, e.g. *<кид> /kit/* "hand, arm" *<кит> /kit/* "whale" (a Russian loan word). Thus an accurate transcription with respect to pronunciation rules is not lossless and might ultimately be considered unnecessary for many purposes: scholars acquainted with the rules of Mari pronunciation can derive the correct pronunciation from a transcription that retains some aspects of the orthography. It is left up to the user to decide whether or not the features described above are taken into consideration:

- If users activate the checkbox labelled "Orthographically unmarked features (assimilation, etc.)", the system will take the features discussed into consideration to the best degree possible.
- With respect to word stress, the system will assume that the stress falls on the last full vowel (see above) unless specified otherwise. Users can manually define the stress for a particular word by placing an asterisk * after the unpredictably stressed vowel.
- Square brackets *[]* can be used to indicate Russian words, names, and text segments as such. Any text enclosed in square brackets will be transcribed in accordance with the rules of Russian, rather than Mari, orthography (these rules are detailed in the documentation). For example, if the name *<Домодедово>* is placed in square brackets, it is transcribed as */Domod'edovo/* rather than */Domodedoβo/* – with a Russian palatalized consonant */d'/* and the letter *<д>* having its Russian value */v/* rather than Mari */β/*.

5.3 Early 20th century orthographies

Mari was subjected to an extensive orthographic reform in 1938 [16, p.291]. Numerous Mari-language newspaper texts from the 1920s and 1930s made available on the National Library of Finland's website [17] use the orthography that become obsolete with this reform, which differs significantly, but systematically from the contemporary orthography. Before 1938 the letter *<е>* was only used after palatal consonants and the sound */e/* was otherwise consistently marked by the letter *<э>*. Moreover, the earlier orthography consistently marked the phoneme */j/* with the letter *<й>*. Table 3 shows the different manner in which different sound combinations are indicated in the old and contemporary orthographies respectively, and illustrates that defects re-

			Example		
UPA	**1930s**	**Contemporary**	**UPA**	**1930s**	**Contemporary**
/ja/	<йа>	<я>	/Japonij/	<Йапоний>	<Японий> “Japan”
/C’a/	<Сьа>	<Ся>	/okt’abr’/	<октьабрь>	<октябрь> “October”
/je/	<йэ>	<е>	/mijen/	<мийэн>	<миен> “(s)he went”
/C’e/	<Сьэ>	<Се>	/ə̑l’e/	<ыльэ>	<ыле> “(s)he was”
/ju/	<йу>	<ю>	/jumo/	<йумо>	<юмо> “god”
/C’u/	<Сьу>	<Сю>	/pol’us/	<польус>	<полюс> “pole”
/Ce/	<Се>	<Се>	/den/	<дэн>	<ден> “and”

Table 3: Mari orthographies: pre-1938, and today

garding the marking of palatalness in modern orthography were not found in pre-1938 writing systems[9].

The software is capable of transcribing texts from the old orthography into the contemporary one. As the old orthography is less ambiguous, this is not difficult from a technical point of view. Because the old orthography is now defunct, the software does not offer transcriptions into it, despite its better handling of palatalness.

6 UPA ⟷ IPA

Once correspondences were established between UPA and IPA values, a transcription from UPA into IPA (or from Cyrillic into IPA via UPA) was more or less straightforward. One problem that arose here, however, is that UPA does not distinguish between palatal and palatalized consonants: UPA */ń/* corresponds to both IPA */ɲ/* and IPA */nʲ/*. As Mari has palatal rather than palatalized consonants, I configured the software, by default, to transcribe UPA */ń/* as IPA */ɲ/* and to transcribe */ń/* as */nʲ/* only within words or phrases marked as Russian by users by means of square brackets (see above). Thus *<сугынь>* “blessing” is transcribed into UPA as */sugə̑ń/* and then into IPA as */sugɤɲ/*, but Russian *<июнь>* “June”, if placed within brackets, is transcribed into UPA as */ijuń/* and then into IPA as */ijunʲ/*[10].

[9] function thirtiesprep

[10] function upa_to_ipa, function ipa_to_upa

7 Conclusions and prospects

For the time being, I have created language-specific diacritic helpers for a total of 102 languages of Eurasia, roughly half of which use the Cyrillic alphabet. Like the Mari-related mechanisms detailed in this paper, these can be found at `transcribe.mari-language.com`. The diacritic helpers allow users to access the specific special characters used in a language's alphabet using shortcuts. For example, the diacritic helper for Udmurt – which uses five characters not found in the Russian alphabet, *ӝ*, *ӟ*, *ӥ*, *ӧ*, and *ӵ*) – carries out the following transformations (on both lower-case and upper-case characters): *ж:* ⟶ *ӝ*, *з:* ⟶ *ӟ*, *и:* ⟶ *ӥ*, *о:* ⟶ *ӧ*, *ч:* ⟶ *ӵ*[11].

In order to create the kind of multilateral transcription and transliteration infrastructure for Udmurt and other Uralic and non-Uralic languages that I have created for Mari (Meadow and Hill), Tatar, and Russian (although in this paper it has only been possible to describe the transcription mechanisms I have implemented for Meadow Mari), I would need the kind of information on the writing systems used for these languages that I have explained in this paper: What pitfalls are there that must be avoided when transcribing, for example, Udmurt Cyrillic into UPA and IPA? My main purpose in writing this paper was to motivate my fellow Uralic scholars to provide me with information of this nature on the language(s) of their expertise to enable the creation of such much needed infrastructures.

References

[1] Árpád Berta. Tatar and bashkir. *The Turkic Languages*, 1998.

[2] Bernard Spolsky. *Language Policy – Key Topics in Sociolinguistics*. Cambridge University Press, 2004.

[3] Thomas A. Sebeok and Alo Raun. *The First Cheremis Grammar (1775)*. The Newberry Library, 1956.

[4] И. Г. Иванов et al. *Марий орфографий мутер*. МарНИИЯЛИ [`komikyv.ru/pdf/orfografi_muter.pdf`, accessed 2 November 2015], 2011.

[5] Л. П. Васикова. *Кырык марла орфографи лӹмдер*. Мары Элӹн периодика, 1994.

[11] These mechanisms are available for Mari as well, if one asks the infrastructure to transcribe from Cyrillic into Cyrillic.

[6] Eemil Nestor Setälä. Über transskription der finnisch-ugrischen sprachen [sic]. *Finnisch-ugrische Forschungen*, 1901.

[7] Alho Alhoniemi. *Marin kielioppi.* Suomalais-Ugrilainen Seura, 1985.

[8] Ödön Beke et al. *Mari nyelvjárási szótár I-IX.* Bibliotheca Ceremissica, 1997–2001.

[9] Alho Alhoniemi and Sirkka Saarinen. *Timofej Jevsevjevs Folklore-Sammlungen aus dem Tscheremisschen I-IV.* Suomalais-Ugrilainen Seura, 1983-1994.

[10] Jeremy Bradley. Mari converb constructions – interpretation and translation. *New Trends in Nordic and General Linguistics*, 2015.

[11] Timothy Riese, Jeremy Bradley, and Elina Guseva. *Mari-English Dictionary.* University of Vienna [`dict.mari-language.com`], 2014.

[12] Alho Alhoniemi. *Grammatik des Tscheremissischen (Mari).* Helmut Buske Verlag, 1993.

[13] Pekka Sammallahti. *The Saami Languages – An Introduction.* Davvi Girji OS, 1998.

[14] Timothy Riese, Jeremy Bradley, Emma Yakimova, and Galina Krylova. *Онгай марий йылме: A Comprehensive Introduction to the Mari Language (Release 2.1).* University of Vienna [`omj.mari-language.com`], 2012.

[15] И. Г. Иванов. *Кызытсе марий йылме – фонетика.* Марий книга савыктыш, 2000.

[16] И. Г. Иванов. *Марий литературт йылме историй.* Марий кугыжаныш университет, 2003.

[17] National Library of Finland. *Kansalliskirjasto Uralica.* National Library of Finland [`uralica.kansalliskirjasto.fi`, accessed 2 November 2015], 2013–.

Uralic and its Neighbors as a Test Case for a Lexical Flow Model of Language Contact

Johannes Dellert
Universität Tübingen
Seminar für Sprachwissenschaft
jdellert@sfs.uni-tuebingen.de

December 31, 2015

Abstract

This paper introduces a new method for inducing a language contact model from lexical data. Based on sets of etymologically related words which can be either automatically inferred or expert-annotated, the method analyses possible paths of borrowing in terms of lexical flow. The criterion of vanishing lexical flow gives rise to a conditional independence relation between languages, allowing a variant of the PC algorithm for causal inference to be applied.

The resulting partially directed network represents a parsimonious model of common ancestry and directional contact between the languages in the dataset. In an evaluation on a large lexical database comprising 1,016 concepts across 26 Uralic languages and 18 neighboring languages, the method is shown to detect and correctly infer the directionality of many instances of cross-family language contact which had a large impact on the basic lexicon.

1 Introduction

Recent computational methods for historical linguistics have the disadvantage of being imprecise due to abstraction over relevant details, but the advantage of weighing more evidence than a human brain can process in principled and reproducible ways. While methods for estimating phylogenies from cognacy judgments are already highly developed and in widespread use (e.g. [1, 2, 3]), the network models

used to account for language contact are still in their infancy [4]. Simple network methods such as Neighbor-Net [5] offer a visual summary of contradictory signals in the underlying character data in the form of reticulations, but do not give any hint about their interpretation. A reticulation in such a network may be the effect of anything from a dialect continuum to massive lexical borrowing.

The Uralic family is an ideal test case for evaluating automated methods because it is relatively small and quite well-understood. Previous computational work on Uralic by the BEDLAN group applies standard methods to infer trees [6] and networks [7], with interesting results concerning the reality of subgroupings above the level of primary branches. For these purposes, the group collected expert cognacy judgments for the realizations of around 300 concepts across 18 Uralic languages.

The method presented in this paper needs more than a few hundred concepts to yield good results, but has the advantage of inferring an explicit directional model of lexical influence between languages. Because expert cognacy judgments for a larger number of concepts are not readily available, I resort to automatically inferred sets of **correlates** (i.e. etymologically related words, not necessarily cognates) over a larger lexical database which covers Uralic and its neighbors.

Building only on correlate sets, the method uses ideas from causal inference to infer a partially directed network between attested languages, where each link represents either common inheritance or contact, and directed links can be taken to represent the dominant direction of borrowing. An initial evaluation on the Uralic dataset shows that the method correctly detects all the major cross-family contact events, and infers the right directionality for all but a few of them.

2 Conditional Independence and Causal Inference

Causal inference [8] is a relatively new subfield of statistics which attempts to infer causal relationships between variables from observational data alone. While the fact that correlation is not causation prevents us from inferring the direction of causality between an isolated pair of variables, the interaction between more than two variables often provides hints about the possible causal scenarios underlying the data.

The core building block of causal inference is a **conditional independence** relation between the variables involved. Intuitively, the conditional independence relation $(X \perp\!\!\!\perp Y \mid Z)$ expresses that any dependence between the variables X and Y can be explained by the influence of a third variable Z.

The inference techniques I will be using are inspired by the PC algorithm [9]. The first stage of the PC algorithm uses a sequence of conditional independence tests to reduce a complete graph to a **causal skeleton**, an undirected graph over the variables

where each link expresses an interaction which cannot be explained away by conditioning on other variables. Each removal of a link $X - Y$ relies on finding a **separating set**, i.e. a set of variables $\{Z_1, \ldots, Z_n\}$ such that $(X \perp\!\!\!\perp Y \mid Z_1, \ldots, Z_n)$.

In the second stage, the separating sets which were used to explain away each link are used to detect v-structures among triples of variables, i.e. causal patterns of the shape $X \rightarrow Z \leftarrow Y$. The presence of v-structures typically allows the PC algorithm to infer the directionality of causal influence along many links in the skeleton.

3 The Data

I am testing my inference procedure on a development version of NorthEuraLex [10], a lexical database of Northern Eurasia which aims to cover the realizations of 1,016 concepts across 100 languages of Northern Eurasia. The version I am using contains near-complete coverage of this concept list for 26 Uralic languages, and 18 languages which have historically been in close contact with Uralic languages.

On the Uralic side, the database includes six Finnic languages (Finnish, North Karelian, Livvi-Karelian, Veps, Standard Estonian, Livonian), six Saami languages (Southern Saami, Lule Saami, Northern Saami, Inari Saami, Skolt Saami, and Kildin Saami), the two written variants of both Mordvinian (Erzya and Moksha) and Mari (Meadow Mari and Hill Mari), three Permic languages (Komi-Zyrian, Komi-Permyak, Udmurt), Northern Khanty, Northern Mansi, Hungarian, and four Samoyed languages (Nothern Selkup, Tundra Nenets, Tundra Enets, Nganasan).

The contact languages consist of four Turkic languages (Chuvash, Tatar, Bashkir, Kazakh) and 13 Indo-European languages, including four Germanic (German, Danish, Swedish, Norwegian), two Baltic (Latvian and Lithuanian), and six Slavic languages (Russian, Polish, Czech, Slovak, Croatian, Bulgarian), as well as a single Romance language (Romanian). In addition, the Yeniseian language Ket was included as an important contact language in central Siberia.

3.1 Sound Correspondence Model and Phonetic String Distance

All the data was normalized to the ASJP format [11], a de-facto standard in distance-based approaches which reduces IPA to 41 equivalence classes. The encoding is designed to make long-distance comparison easier, but ignores some features (such as vowel length) that are highly relevant for Uralic. To give a few examples, Northern Saami *čalbmi* is represented as `[CalEbmi]`, and Hungarian *egyedül* as `[ECEdil]`.

For each segment in the ASJP strings, the information content was inferred from trigram models for each language and word class. This is necessary when operating on

dictionary forms, since a string-distance based method would otherwise overestimate the similarity e.g. between verbs which share an infinitive ending.

To enable the detection of cognates which have become dissimilar due to sound change, a model of segment correspondences was inferred separately for each language pair using a variant of the method described by List [12], which results in a segment distance matrix for each language pair. For instance, the distance matrix for Finnish and Hungarian makes it cheap to align a `[k]` to an `[h]`, and the matrix for Hungarian and Northern Saami assigns a low cost to aligning `[s]` and `[C]`. Using the language-specific segment distances and the information content as weights, normalized edit distances were computed for all pairs of realizations of the same concept.

3.2 Correlate Inference

The automated inference of correlate sets (under the name of cognacy detection) is an emerging subfield of computational historical linguistics [13, 12, 14]. My implementation of the LexStat toolchain [15], like the original, uses the UPGMA algorithm [16] to derive a hierarchical clustering of the phonetic strings for each concept based on their pairwise distances, and cuts the tree at a given threshold value to partition the strings into clusters of similar forms.

For any set of languages $L_1, \ldots, L_k$, I will write the number of correlates shared between all of them according to this partition as $c(L_1, \ldots, L_n)$. For a single language L, $c(L)$ then denotes the number of correlate sets covered. This number will later be used for normalization in order to compensate for the effect of the slightly uneven coverage of the concept list for the different languages.

4 Modeling Lexical Flow

The application of the PC algorithm to this dataset presupposes a useful definition of conditional independence between sets of languages. The idea of this paper is to use a **lexical flow model** to define such a conditional independence relation. Building on sets of correlates $cor(L_1, ..., L_k)$ shared by a the languages $L_1, \ldots, L_k$, the independence test can be based on a measure of conditional overlap, which I will call $I(L_1, L_2; Z)$ because of parallels with conditional mutual information:

$$I(L_1, L_2; Z) := \frac{|cor(L_1, L_2) \backslash \{c \mid \exists \{Z_1, \ldots, Z_k\} \subseteq Z \colon c \in cor(Z_1, \ldots, Z_k)\}|}{\min\{|cor(L_1)|, |cor(L_2)|\}} \tag{1}$$

Informally, $I(L_1, L_2; Z)$ quantifies the ratio of correlates between L_1 and L_2 which cannot be explained away by having been borrowed through a subset of the languages in Z. To use this measure of dependence as a conditional independence test, we simply check whether $I(L_1, L_2; Z) \leq \theta_{L_1,L_2}$ for a threshold θ_{L_1,L_2}, which could be derived from the number of false correlates between L_1 and L_2 which we expect due to automated correlate detection. In practice, I am setting $\theta_{L_1,L_2} := 0.02$ for all language pairs because the distribution of false correlates is difficult to estimate, and language-specific thresholds did not lead to better results in initial experiments on a smaller language set. On the NorthEuraLex data, this means that languages which share 20 correlates or less will be unconditionally independent, and every link the algorithm establishes will explain an overlap of at least 20 correlates.

Based on this conditional independence test, the first stage of the PC algorithm derives a causal skeleton which represents a scenario of contacts between pairs of input languages that is only as complex as necessary to explain the lexical overlaps. The model thus assumes that all similarities are primarily due to mutual influence, and never infers the existence of hidden common causes (i.e. proto-languages), although the links without any clear unidirectional signal can be interpreted in this way.

The PC algorithm is tractable because it tests separating set candidates in order of cardinality, and builds on the assumption that any separating set must be a subset of immediate neighbors of L_1 and L_2 in the current skeleton. For our model, we cannot make that assumption, because removal of a link between two languages should not rely on shared correlates with possibly unconnected neighbors. Instead, we need to explicitly model the lexical flow.

To explain away a correlate that is shared between two languages L_1 and L_2, it must have been possible for the lexeme in question to have travelled between the two languages on some other path. Therefore, any minimal separating set must form a union of acyclic paths between L_1 and L_2. My implementation uses a depth-first search of the current graph to get all such paths which contain four nodes or less, and generates all combinations of these paths which lead to separating set candidates of a given cardinality. Longer paths would need to be considered in theory, but did not lead to different results on my data, at a much higher computational cost.

5 Deciding Directionality

In the second stage of the standard PC algorithm, directionality inference on the causal skeleton is performed by asking whether the central variable in each pattern of the form $X - Z - Y$ was part of the separating set that was used for explaining away the link $X - Y$. The idea is that if Z was not necessary to explain away $X - Y$, this

excludes all causal patterns except $X \rightarrow Z \leftarrow Y$. Since there will often be many separating sets of the same size, the result of this decision procedure can be highly dependent on the order in which separating set candidates are tried out. In practice, this means that many possible orders have to be tested, often giving rise to conflicting evidence which needs to be reconciled. Moreover, this type of inference relies on the very strong assumption that every scenario in which X has an influence on Y and Y on Z, this would become visible as a dependence between X and Z. While this assumption may be unproblematic for continuous statistical variables, it is certainly not true for our notion of independence, since it is easily conceivable that if a language L_1 borrows from a language L_2 which in turn borrows from a language L_3, none of the lexical material from L_3 will appear in L_1.

Still, the essential idea behind this reasoning can also be applied to our case. For each triple of languages (L_1, L_2, L_3) and a given causal scenario, we can measure the difference between the expected number of correlates shared between all three languages, and the observed number of such correlates. More precisely, if the number of observed correlates $c(L_1, L_2, L_3)$ is significantly lower than the number we expect under any causal assumption which includes $L_1 \leftarrow L_2$, this gives us evidence in favor of the arrow $L_1 \rightarrow L_2$. So what is the expected number of shared correlates between all three languages under the assumption $L_1 \leftarrow L_2$? Assuming independent instances of language contact, both scenarios $L_1 \leftarrow L_2 \leftarrow L_3$ and $L_1 \leftarrow L_2 \rightarrow L_3$ allow us to multiply the ratios $r(L_1, L_2) := c(L_1, L_2)/\min\{c(L_1), c(L_2)\}$ and $r(L_2, L_3) := c(L_2, L_3)/\min\{c(L_2), c(L_3)\}$ to arrive at the percentage of $c(L_1, L_3)$ that we expect to also be shared with L_2.

In a triangle $L_1 - L_2 - L_3$, the amount of the information we can derive about the directionality of $L_1 - L_2$ in this way becomes higher the more correlate overlap there is between L_2 and L_3. This gives rise to a definition of the **counterevidence score** $sc(L_1 \rightarrow L_2)$ for the arrow $sc(L_1 \rightarrow L_2)$ based on a weighted sum over all triples:

$$sc(L_1 \rightarrow L_2) := \sum_{L_3} c(L_2, L_3)^2 \cdot \frac{c(L_1, L_2, L_3)}{r(L_1, L_2) \cdot r(L_2, L_3) \cdot \min\{c(L_1), c(L_3)\}} \quad (2)$$

Based on these scores, directionality decisions for each language pair $L_1 - L_2$ can be made by comparing the strength of counterevidence for $sc(L_1 \rightarrow L_2)$ and $sc(L_2 \rightarrow L_1)$. For the experiment, my implementation assumes that the evidence favors one direction if the ratio of counterevidence scores is lower than 0.9. As in the standard interpretation of causal graphs returned by the PC algorithm, counterevidence score ratios near 1.0 can be interpreted as being a consequence of either bidirectional influence (mutual borrowing) or a hidden common cause (ancestral relationship). Algorithm 1 gives an overview of the entire resulting inference procedure in pseudocode.

Algorithm 1 infer_network($L_1, \ldots, L_n$)

$G := (\{L_1, \ldots, L_n\}, \{\{L_i, L_j\} \mid 1 \leq i \neq j \leq n\})$, the complete graph
$s := 0$
while $s < n - 2$ **do**
  for $\{L_i, L_j\} \in G$ by increasing strength of remaining flow **do**
    for each combination $P_1, ..., P_k$ of paths from L_i to L_j of length ≤ 4 **do**
      if $|S| = s$ for $S := \bigcup\{P_1, \ldots, P_k\}$ **then**
        if ratio of $c(L_i, L_j)$ not explainable by flow across S is < 0.02 **then**
          remove $\{L_i, L_j\}$ from G
        end if
      end if
    end for
  end for
  $s := s + 1$
end while
for $\{L_i, L_j\} \in G$ **do**
  if $sc(L_i \rightarrow L_j)/sc(L_j \rightarrow L_i) < 0.9$ **then**
    add arrow $L_i \rightarrow L_j$ to network
  end if
end for
return network consisting of G and arrows

6 Results

Using my Java implementation, applying the method to the NorthEuraLex correlate sets takes less than five minutes on a single core with 2.2 GHz. Figure 1 shows the resulting network. For the visualization, languages (symbolized by their ISO 639-3 codes) are placed roughly at their geographical positions. Contact arrows are colored green, and links for which directionality evidence did not exceed the threshold are in black. The thickness of each edge symbolizes the amount of unexplained flow, i.e. the amount of lexical flow which the model assumes must have gone through the link in question.

Note that Indo-European (red nodes) and Uralic (blue nodes) form two clusters of black edges which are only connected by contact edges, i.e. the model correctly separates these two language families, and can explain all lexical similarity between them by contact alone. The same is not true for the separation of Turkic (purple nodes) and Uralic, though. The complex interaction between Chuvash (chv) and the

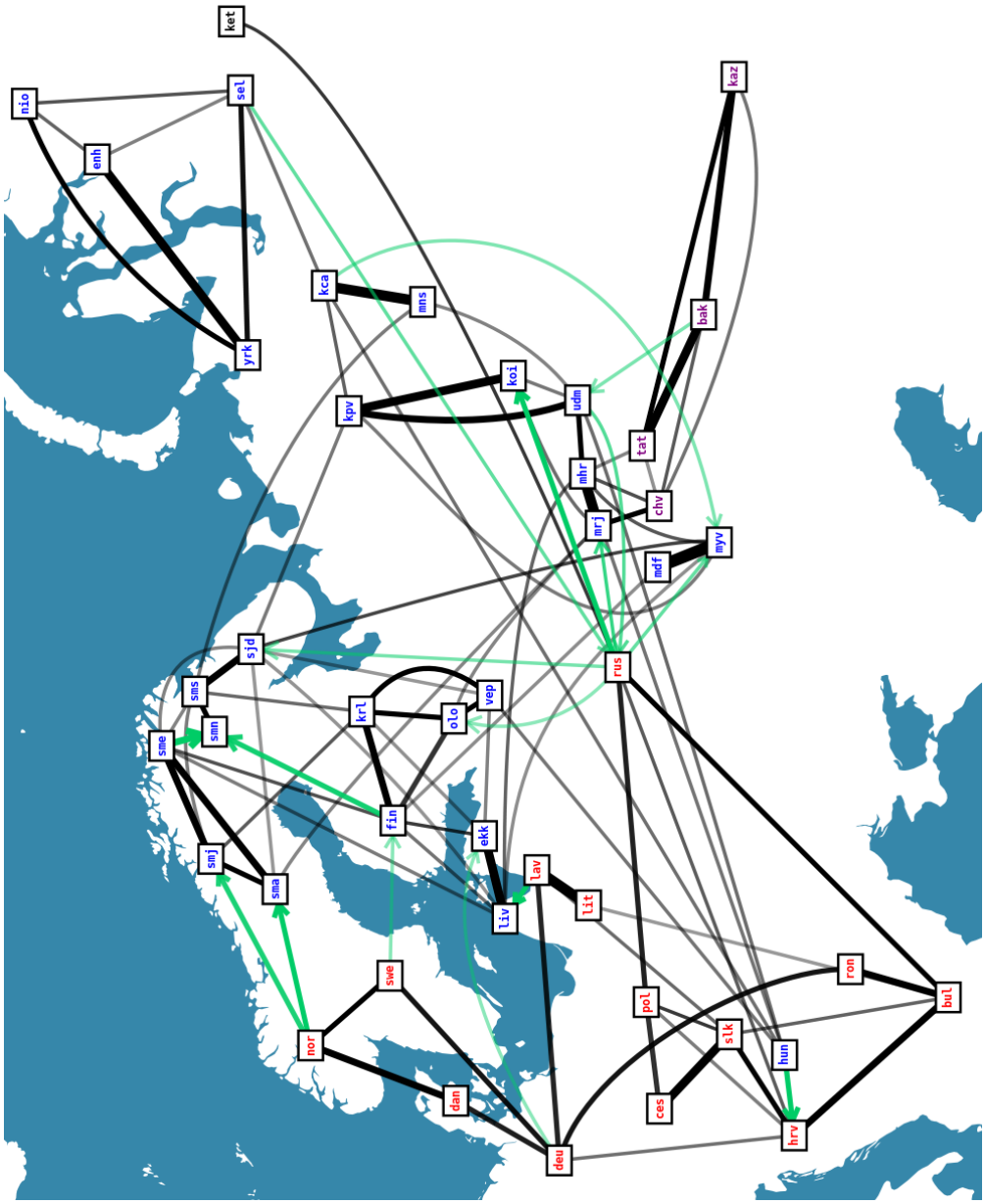

Figure 1: The lexical flow network derived from the data.

two variants of Mari prevents the method from deriving a directionality. One problem may be that Chuvash actually exerted most of its considerable influence on an earlier historical stage of Mari, and not separately on Hill Mari (mrj) and Meadow Mari (mhr), as the model was forced to infer. Otherwise, all the contacts with Turkic languages inferred by the model are correct, though of course far from exhaustive [17].

The internal structure of the language families becomes visible rather nicely in the different intensities of inferred lexical flow among members of the same branch and across branches. For instance, the connection between Latvian (lav) and Lithuanian (lit) is much stronger than the one between Latvian and German (deu). Also, the dialect chain structure of Saami [18] and Finnic [19] becomes quite clearly visible, as geographical neighbors share more lexical material than do more distant pairs of languages from these branches.

Considering the contacts around the Baltic sea, the model correctly detects strong influence of North Germanic [20] (represented by Norwegian) on Lule Saami (smj) and Southern Saami (sma). Interestingly, the remaining lexical overlap between Northern Saami (sme) and Norwegian is not enough to infer direct contact, since all the shared correlates may have entered Northern Saami through either Lule or Southern Saami. The strong influence of Swedish (swe) on Finnish (fin) is detected just as well as the influence of German on Estonian (ekk) [21]. Since this arrow can be interpreted to show the influence of the Teutonic Order, the link between German and Latvian could have displayed the same direction, but the flow representing the ancestral relationship of the two languages is stronger than the layer of German loans in Latvian. The contact between Livonian (liv) and Latvian is seen as monodirectional, which is justifiable for the lexicon because there are many Baltic loanwords in Finnic, and an additional stratum of later Latvian loans into Livonian [22]. This link is further strengthened by material from German in Livonian, all of which can be explained as either going through Estonian or Latvian. Finally, the lexical overlap between Russian (rus) and Livvi-Karelian (olo) as well as Kildin Saami (sjd) is correctly recognized as being due to heavy Russian influence on the other two languages [23].

Russian is also correctly detected as exerting considerable influence on many of the Uralic minority languages [23]. In all cases except Udmurt (udm) and Selkup (sel), Russian is correctly recognized as the donor language. Interestingly, any lexical material shared between Russian and the Northern Samoyed languages (yrk, enh, nio) is also shared with Selkup, causing the model to assume that all Russian material in Northern Samoyed was transmitted via Selkup. This is unexpected, because Russian influence on Tundra Nenets (yrk) was actually much stronger than on Selkup [23], but the inferred pattern may be true for the more basic lexicon covered by the database.

The erroneous black edge between Ket (ket) and Russian illustrates that the method runs into problems when faced with an isolate. A possible way towards resolving this

issue would be to check whether the shared correlates belong to the most stable basic vocabulary or to later strata, which would indicate contact as opposed to a genealogical relationship.

7 Conclusion and Outlook

We have seen that causal inference built on a conditional independence relation defined by vanishing lexical flow is a powerful tool for inferring network models of language contact. This new type of network has the advantage of also expressing hypotheses about the dominant direction of lexical borrowing.

The evaluation on Uralic and its contact languages has shown that the major cross-family contact events in the history of the current Uralic languages are correctly detected. Moreover, the method was always right when it inferred the existence of a link, although it could not detect the directionality for all instances of language contact, especially when an isolate was involved. Altogether, the method promises to be a worthwhile tool for providing initial hypotheses about language relationships in less well-researched linguistic areas.

The major problem of the current version is that it does not model the existence of proto-languages, and will therefore always model all instances of contact as occurring between observed languages. In reality, many of the detected contacts will actually have occurred between proto-languages. In future work, the lexical flow model will therefore be combined with ancestral state reconstruction [24] to provide a hypothesis about the correlate sets present at proto-languages, which should make it possible to also infer the existence and directionality of contacts between proto-languages, if they explain the data more parsimoniously than the assumption that only the observed languages influenced each other.

Acknowledgments

This research has been supported by the ERC Advanced Grant 324246 EVOLAEMP, which is gratefully acknowledged. My thanks go to Alina Ladygina for contributing the data for Tundra Enets, Hill Mari, and Komi-Permyak. Furthermore, thanks are due to Alla Münch, Ilja Grigorjew, Thora Daneyko, Natalie Clarius, and Roland Mühlenbernd for their help in collecting the data for the Turkic languages and Ket. Finally, I would like to thank Pavel Sofroniev for implementing the Sanavirta visualization component which was used to create the map.

References

[1] Clare Janaki Holden. Bantu language trees reflect the spread of farming across sub-Saharan Africa: a maximum-parsimony analysis. *Proceedings of the Royal Society of London B: Biological Sciences*, 269(1493):793–799, 2002.

[2] Michael Dunn, Stephen C Levinson, Eva Lindström, Ger Reesink, and Angela Terrill. Structural phylogeny in historical linguistics: methodological explorations applied in Island Melanesia. *Language*, 84(4):710–759, 2008.

[3] Claire Bowern and Quentin Atkinson. Computational phylogenetics and the internal structure of Pama-Nyungan. *Language*, 88(4):817–845, 2012.

[4] Johann-Mattis List, Shijulal Nelson-Sathi, Hans Geisler, and William Martin. Networks of lexical borrowing and lateral gene transfer in language and genome evolution. *Bioessays*, 36(2):141–150, 2014.

[5] David Bryant and Vincent Moulton. Neighbor-Net: An Agglomerative Method for the Construction of Phylogenetic Networks. *Molecular Biology and Evolution*, 21(2):255–265, 2004.

[6] Kaj Syrjänen, Terhi Honkola, Kalle Korhonen, Jyri Lehtinen, Outi Vesakoski, and Niklas Wahlberg. Shedding more light on language classification using basic vocabularies and phylogenetic methods: a case study of Uralic. *Diachronica*, 30(3):323–352, 2013.

[7] Jyri Lehtinen, Terhi Honkola, Kalle Korhonen, Kaj Syrjänen, Niklas Wahlberg, and Outi Vesakoski. Behind Family Trees - Secondary Connections in Uralic Language Networks. *Language Dynamics and Change*, 4(2):189–221, 2014.

[8] Judea Pearl. *Causality.* Cambridge University Press, 2009.

[9] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search.* MIT Press, 2nd edition, 2000.

[10] Johannes Dellert. Compiling the Uralic Dataset for NorthEuraLex, a Lexicostatistical Database of Northern Eurasia. *Septentrio Conference Series*, 0(2):34–44, 2015.

[11] Cecil H. Brown, Eric W. Holman, Søren Wichmann, and Viveka Velupillai. Automated classification of the world's languages: A description of the method and preliminary results. *STUF – Language Typology and Universals*, 61(4):285–308, 2008.

[12] Johann-Mattis List. LexStat: Automatic Detection of Cognates in Multilingual Wordlists. In *Proceedings of the EACL 2012 Joint Workshop of LINGVIS & UNCLH*, pages 117–125, Avignon, France, April 2012. Association for Computational Linguistics.

[13] Bradley Hauer and Grzegorz Kondrak. Clustering Semantically Equivalent Words into Cognate Sets in Multilingual Lists. In *Fifth International Joint Conference on Natural Language Processing, IJCNLP 2011, Chiang Mai, Thailand, November 8-13, 2011*, pages 865–873, 2011.

[14] Taraka Rama. Automatic cognate identification with gap-weighted string subsequences. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, May 31 – June 5, 2015 Denver, Colorado, USA*, pages 1227–1231, 2015.

[15] Johann-Mattis List. *Sequence comparison in historical linguistics*. Düsseldorf University Press, Düsseldorf, 2014.

[16] Robert R. Sokal and Charles D. Michener. A statistical method for evaluating systematic relationships. *University of Kansas Science Bulletin*, 38:1409–1438, 1958.

[17] András Róna-Tas. Turkic influence on the Uralic languages. In Sinor [25], pages 742–780.

[18] Pekka Sammallahti. Saamic. In Abondolo [26], pages 43–95.

[19] Tiit-Rein Viitso. Fennic. In Abondolo [26], pages 96–114.

[20] Ante Aikio. On Germanic-Saami contacts and Saami prehistory. *Journal de la Société Finno-Ougrienne*, 91:9–55, 2006.

[21] Sándor Rot. Germanic influences on the Uralic languages. In Sinor [25], pages 682–705.

[22] Seppo Suhonen. Die baltischen Lehnwörter der finnisch-ugrischen Sprachen. In Sinor [25], pages 596–615.

[23] Gyula Décsy. Slawischer Einfluss auf die uralischen Sprachen. In Sinor [25], pages 616–637.

[24] Gerhard Jäger and Johann-Mattis List. Investigating the potential of ancestral state reconstruction algorithms in historical linguistics. Workshop "Capturing Phylogenetic Algorithms in Linguistics", Lorentz Center, Leiden, 2015.

[25] Denis Sinor, editor. *The Uralic Languages. Description, History and Foreign Influences.* Handbuch der Orientalistik 8. Brill, Leiden, 1988.

[26] Daniel M. Abondolo, editor. *The Uralic Languages.* Language Family Descriptions Series. Routledge, 1998.

Demonstration of Minority Translate, a tool for making small Wikipedias bigger

Kristian Kankainen
MTÜ Keeleleek
kristian@keeleleek.ee

December 31, 2015

Abstract

Minority Translate is a tool that streamlines the process of creating, editing and saving new articles in any language edition of Wikipedia, also the new language editions starting out in the Incubator. The tool has good offline functionality and can be used for working with thousands of articles at a time. The tool adds metadata to translated articles. Statistics and socio-linguistic parameters are collected. Easing the development and growth of Wikipedia as a language resource in itself for the lesser resourced language is seen as the main contribution, but many fruitful connections can and should be made for incorporating other language technology to the tool.

1 Introduction

Minority Translate [1] is a stand-alone tool designed to ease and streamline the process of creating, editing and saving new articles in any language edition of Wikipedia, the free encyclopedia. A Wikipedia in a lesser resourced language is seen as a socio-linguistical prestige booster for the language community, but this works only when the language's encyclopedia is big enough. Size can readily be increased by translating content from other language editions. The main thrust of the tool comes from simple automatization.

The tool presented here is believed to offer a two-folded contribution for language technology and lesser resourced languages. Firstly, Wikipedia is a rich resource in

itself. As a collection of texts it is a corpus and since each version of the articles' texts are saved separately, the corpus has potential for longitudinal/diachronical analysis.

Considering Wikipedia as a format, then much of the effort put into utilizing it from other language editions can be carried over to the lesser resourced languages "for free". Though differences exist between different language editions, some aspects can be seen as universal. Article titles together with their inter-language-links can directly be used as a simple multilingual dictionary. More complex semantic dictionary structures have been created, such as the BabelNet [2]. The diachronical aspect of the texts could be used for example as feedback in language planning or education.

Minority Translate contributes to this Wikipedia universe. Not only by explicitly growing the encyclopedia but also by collecting socio-linguistic meta information about the process. If the user accepts, the tool collects meta data about the translation process. Parameters such as which source language(s) was used for translating an article is saved to a central repository. The collected statistics [3] is available under a free license (CC BY-SA). The user can also mark the created text as being an exact translation. This could help in choosing aligning algorithms if the texts are to be parallelized or compared.

The second point of contribution is the mutual gain when language technology gets incorporated into the tool. On the one side, the tool can offer a place to implement and put in use developed language technology or to test language technology being developed. This on the other hand could increase the speed of content generation which increase the growth of the encyclopedia as a useful resource to language technology and also the growth of available text as a corpus for linguistic analysis. This can be leveraged for example to training further language models used in language technology applications.

Minority Translate has plugins for a few dictionary APIs. Spell checking is possible using Hunspell and an experimental machine translation plugin for Apertium exists.

The inclusion of language technology can also raise the user's awareness of available support for his/her language which can further boost the socio-linguistic status of the language.

2 Helping small Wikipedias grow

Hale [4] has found that multilingual Wikipedia users are much more active in editing and creating articles than their single-edition (monolingual) counterparts and that smaller-sized editions with fewer users have a higher percentage of multilingual users than larger-sized editions. Hale also points out that other studies comparing content

across Wikipedia's language editions have found a "surprisingly small amount of content overlap between languages of Wikipedia".

The latter point shows that there is a substantial gap that the tool can be used to fill and the former that there could exist potential users. Since the tool doesn't constrain but instead rather relaxes the translation relation between the original text(s) and the created article, these users need not be professional translators. Any degree of content transfer is accepted, which is relevant from the point of view of the language community.

The Finno-Ugric Wikipedia Cooperation [5] states that there are 30 Finno-Ugric language editions of Wikipedia. More than half of them (17) are still in the so called Incubator. The Wikipedia Incubator is the place where new language editions can start and build up a community. There are some "rules" for becoming a "real" Wikipedia, such as having an active and large enough community. The encyclopedia's user interface need also to be fully localized. The major difference is that a language edition in the Incubator lacks its own subdomain and thus shares the namespace with the other languages in the Incubator.

The majority of the Finno-Ugric language editions in the Incubator contain less than 300 articles and only two of them have more than a thousand articles. While of the 13 language editions that have managed to become their own Wikipedias, three of them contain more than 100 000 articles, one contain around 10 000 articles and the rest contain less than 6000 articles. Minority Translate supports translation between any Wikipedia, also the editions in the Incubator. Carrying over Hale's aforementioned points, closely related languages and/or cultures could gain impetus from each other.

Minority Translate's first test users have been from the Võru language community in Estonia. The usage statistics intermittently collected during the development of the tool shows that for 151 articles with Võru as target language, one sixth of them were translated using more than one source language. At two occasions no language was used, which shows that the tool can be used also for editing and not only translation. The statistics show that up to five source languages was used at a time. Most common was to use only Estonian as source language but second most common was using three: Estonian, English and Finnish. The rest of the collected statistics show singular article creations in other target languages – Swedish, Hungarian, Mordvin-Erzya and Veps. There has been shown interest in adopting the tool by the Karelian Wikipedia community and currently a work-group is being put together for the Votic language edition.

There exists some problematic issues in the field of minority languages' Wikipedias. Minority Translate sees these as inherent in the open philosophy that everyone is free to contribute to the encyclopedia. This is an enormous responsibility for a commu-

nity and can collide with the language community's concerns of language cultivation. Often it is the case that language enthusiasts are not native speakers and are not connected to the community by other means than pure enthusiasm.

Enabling machine translation and automatization in the content generation process amplifies these dangers that the Wikipedia is filled with less grammatically correct text, which in turn affects the credibility of the encyclopedia. Minority Translate doesn't specifically try to prevent any such misuse of the tool but requires the user to be logged in. All contributed edits are linked with the account and there exists mechanisms in the Wikipedia platform for dealing with different kind of misuse and also cleaning up by the other contributors. The problematic issues are thus left to be dealt with at the source and by the community of the free encyclopedia.

3 Minority Translate

Several attempts have been made to create content translation tools for Wikipedia [6]. These tools divide into three genres: a) they are built around a specific machine translation service, b) they are oriented towards the bilingual capacity of a specific language community or c) they try to integrate with the Wikipedia infrastructure.

The tools built around proprietary machine translation services are inherently limited to the service and its supported language pairs. Some tools are limited to English as the only source language. The tools conceived in specific language communities are also limited in terms of which language pair is supported.

Only the last group de-centralizes the notion of language and trivializes supporting new language editions of the free encyclopedia. This last group is represented by two tools, ContentTranslation and the tool presented here. ContentTranslation is built in to Wikipedia as a beta feature for logged-in users and may be regarded as an official Wikimedia project. It was made available during the same time as Minority Translate. The ContentTranslation project is open for suggestions of more resources for languages to be included.

There are three key aspects where the two tools differs: support of the Wikipedia Incubator, being web-based and having single language orientation. Unfortunately ContentTranslation is not available for language editions in the Incubator. Supporting the Incubator is crucial for starting language editions if their size is to be enlarged by translation. Hopefully this difference is simply a matter of time.

Being web-based makes ContentTranslation dependent on constant internet connection, which might not always be available nor affordable. Minority Translate downloads all articles being worked on and the translated articles can later be uploaded in batch.

Both tools work-flows focus around a list of articles to translate. ContentTranslation has an automatically generated list of suggestions which focus on ”good quality” articles. Instead Minority Translate has many lists and it is favoured that the community makes their own list or adapts existing lists. These article lists can be a great way to guide and set strategic goals for the language community and adapt the task of translation to their particular community needs or interests.

ContentTranslation is single language oriented and opens the source article in only one language. Minority Translate makes it easy for the user to open any number of existing source articles in any language the user knows at the same time. One sixth of all articles translated with Minority Translate has used more than one source language.

It should here be stated, that upon the availability of ContentTranslation in the Võru Wikipedia, the users have started using it instead of Minority Translate and finds it “elegant” and “more intuitive”.

4 Acknowledgments

MTÜ Keeleleek acted as the fiscal sponsor for the tool. Keeleleek is an Estonian NGO propagating free software principles in Estonian language technology. The project was lead by Ivo Kruusamägi from Wikimedia Eesti. The Software was developed by Andrjus Frantskavitsius. The creation of the tool was funded by two Project & Event Grants by the Wikimedia Foundation in the spring of 2014 and in the autumn of 2015.

References

[1] Minority Translate. `http://translate.keeleleek.ee/wiki/Esileht`.

[2] Roberto Navigli and Simone Paolo Ponzetto. BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. *Artificial Intelligence*, 193:217–250, 2012.

[3] Minority Translate Statistics. `http://mwtranslate2.keeleleek.ee/`.

[4] Scott A. Hale. Multilinguals and Wikipedia Editing. 2013.

[5] English/Fenno-Ugric cooperation – Wikimedia Eesti. `https://ee.wikimedia.org/wiki/English/Fenno-Ugric_cooperation`.

[6] Machine translation - Meta. `https://meta.wikimedia.org/wiki/Machine_translation#Attempts`.

Parsing Estonian: Tools and Resources

Kadri Muischnek
University of Tartu
kadri.muischnek@ut.ee

Kaili Müürisep
University of Tartu
kaili.muurisep@ut.ee

Tiina Puolakainen
University of Tartu
tiina.puolakainen@ut.ee

Krista Liin
University of Tartu
krista.liin@ut.ee

January 3, 2016

Abstract

This article gives an overview of the state of the art of tools and resources for syntactic analysis of Estonian. A morphosyntactic disambiguator, surface-syntactic analyser and dependency parser are all based on the Constraint Grammar formalism. Also, the paper describes some experiments conducted with the statistical parser. As for language resources, a 400,000-word manually annotated dependency treebank has been created. Our tools have also been tested by large-scale corpus annotation.

1 Introduction

This paper describes a set of tools and resources for parsing Estonian texts starting from morphological analysis and disambiguation to dependency parsing and syntax-based applications. In 1995, the first version of morphological analyser of Estonian ESTMORF was created and already couple of years later it was able to assign adequate morphological descriptions to 99% tokens in text [1]. In the same year, Fred Karlsson together with his colleagues published a monograph on Constraint Grammar [2], a framework for disambiguating and parsing non-restricted text that has been successfully used not only for analysing the Indo-European languages but also e.g. for analysing Finnish.

That spurred the work on Estonian Constraint Grammar (EstCG). Its earlier versions used a locally developed parsing engine, but its last version uses VISL CG-3 format and software [3]. EstCG parser consists of separate sets of grammar rules for determining clause boundaries, morphological disambiguation, surface syntactic analysis and determining dependency relations. In addition to rule-sets, the system also includes several valency lexicons and a special module for identifying particle verbs [4].

The rest of the paper is organized as follows. Sections 2 and 3 provide an overview of morphological disambiguation and clause boundary detection module, sections 4 and 5 describe the grammar of surface and dependency syntax, section 6 reports the experimental results of applying MaltParser to Estonian. Section 7 gives an overview of graphical user interface for combining different modules together. In section 8, we describe Estonian Dependency Treebank. In section 9, we conclude the paper with describing some applications of our syntactic tools and discussing some ideas for future work.

2 Morphosyntactic disambiguator

EstCG parser takes morphologically analysed text as input, i.e. each word-form has all the possible morphological analysis attached to it. Morphological ambiguity rate of Estonian text is ca 50%. For example word-form *või* can be noun *või* ('butter') in nominative or genitive case-form, negative present tense form of verb *võima* ('may') in all three persons in singular and plural or conjunction *või* ('or').

Constraint Grammar rules for morphological disambiguation delete the readings that are inappropriate regarding the context. Preliminary clause boundaries are also set at the same stage. If it is not possible to disambiguate basing on the contextual information, all possible readings are retained.

The disambiguating grammar consists of more than 3400 handwritten rules, almost a quarter of them address certain word-forms. For example a very frequent word-form *on* is ambiguous between the readings of simple present 3rd person singular and plural of the verb *olema* ('to be'). The other rules can again cover broader ambiguity classes.

A difficult case for disambiguation is the choice between readings of nominative, genitive, partitive or short illative (additive) case forms of a noun. This type of ambiguity tends to be more characteristic of frequent and common words, eg. nouns *ema* ('mother') and *isa* ('father') are ambiguous between nominative, genitive and partitive readings. The word-form *metsa* 'forest' is an example of typical homonymous form of singular genitive, partitive and short illative cases. Its parallel form of illative case,

in this example *metsasse* ('into the forest'), is actually not used in Estonian.

The other frequent sources of errors and ambiguities are participles (they are always four-way ambiguous: negative indicative past tense, past participle, adjectival use of past participle and noun as a nominalisation of an adjective), and also ambiguous readings of adposition, adverb and noun of some word forms.

For example, *peale* can be an autonomous adverb (most general meaning 'onto') or a particle as a part of a particle verb, e.g. *peale sattuma* 'stumble on/across'; it can be also a postposition governing a noun in genitive case (meaning 'in addition to') or elative case (meaning 'starting from') or preposition governing a noun in genitive case or partitive case (meaning 'after'); after all, *peale* can be also a noun *pea* ('head') in a singular allative case. As a consequence of this multi-way ambiguity of the word-form *peale*, the Estonian phrase *asetama selle peale* can have 3 different meanings: (1) 'to place onto this' with *peale* as a postposition; (2) 'to place this onto' with *peale* as a particle; (3) 'to place this onto the head'.

Tests made with a 26,700 word test corpus showed results of 97.1% recall and 90.2% precision. In other words, the output contained 2.9% of errors (word-forms that did not contain the correct reading among all survived readings in a cohort) and 9.8% of retained readings were not correct (superfluous). The initial morphologically analysed text contained 51.8% of superfluous readings and 0.6% of word-forms did not have a correct reading in the cohort (recall 99.4% and precision 48.2%). This happens most often with unknown words, mostly proper noun, but also in other cases, for example word-form *väikesed* ('small') is given only an adjective reading but in some sentences it is functioning clearly as a noun.

A common source of errors are elliptical sentences as for example a title *Suhtlemise puudus* ('Lack of communication'), there the word-form *puudus* ('lack') is considered to be a verb being an only possibility for that in the sentence, but in this case should be a noun as the sentence contains only one noun phrase.

One of the hardest tasks is disambiguation of noun forms with homonymous nominative, genitive and partitive or genitive, partitive and additive case forms. The following sentence (1) has two appropriate readings depending what role the noun *rõõm* ('joy') is playing in the sentence – an object of the main verb and consequently has to be considered being in partitive case or a modifier of a noun *koostegemine* ('cooperation') and then accordingly in genitive case:

(1) a. *Külades* *tuntakse* *rõõmu* *koostegemisest.*
village[INE.PL] know[IMPS.PL] joy[GEN] cooperation[ELA] (INE=inessive)
'The cooperation of joy is known in the villages.' (ELA=elative)

b. *Külades* *tuntakse* *rõõmu* *koostegemisest.*
village[INE.PL] feel[IMPS.PL] joy[PAR] cooperation[ELA] (PAR=partitive)
'There is a feeling of joy of cooperation in the villages.'

3 Clause boundary detector

The clause boundary annotation is a simple way to constrain context of morphosyntactic rules, also, the performance of statistical parser improved if the model had information about clause boundaries. Currently, EstCG has ca 80 hand-crafted rules for detecting clause boundaries. The beginning of each clause is annotated by a special label.

The rules mainly consider conjunctions, punctuation marks, finite verbs, relative adverbs and pronouns. Although these are simple cues for assuming a clause boundary, often it is not obvious, how to distinguish clause-initial position from coordinating or modifying usage within a clause, as a morphologically analysed (but not yet disambiguated) text contains plenty of ambiguities for different interpretations (a classical but not single example is past participles that can function as a predicate of a clause or just an adjectival modifier of a noun). Also special clause boundary tags are introduced for embedded clauses, where, for example, a subject and a predicate of main clause may be separated by a relative clause and therefore would be not related to each other without special care.

4 Surface-oriented syntactic analyser

The syntactic or, more precisely, the surface-syntactic module of the EstCG adds a label of syntactic function to every word-form in the text. According to the EstCG annotation scheme, the members of the verbal chain can be finite or infinite main verbs (FMV, IMV), and finite or infinite auxiliaries (FCV, ICV). Also, we distinguish particles as parts of particle verb (VPart), and verb negators (NEG). The arguments of the verb are labelled as subject (SUBJ), object (OBJ), predicative (PRD) or adverbial (ADVL); the adjuncts also get the adverbial label. The attributes of a nominal are tagged according to their part-of-speech (AN for adjectives, NN for nouns, KN for adpositions, DN for adverbs and INFN - for infinitives). We distinguish the nouns governed by an adposition with a special label (<P or P>) and also nouns governed by a quantifier (<Q or Q>). There is a special symbol for indicating whether the word form is a pre- or postmodifier (<NN or NN> for example). Also, we label direct addresses (VOC), conjunctions (J) and interjections (I).

The annotation created by the analyser is very shallow: the clause boundaries are set and the syntactic functions of the word-forms in every clause are labelled, but no inter-clausal relations are identified.

Also, the head verbs are not connected with their arguments. For example, if a clause contains an infinitive subclause and both verbs have an object, there is no way to tell from the annotation which object complements which verb. Also, the objects are not connected with their head verbs and if a clause contains an infinitive subclause and both verbs having an object, there is no way to tell from the annotation which object complements which verb. There is no direct connection between an attribute and its head, but pre- and postmodifying attributes are distinguished.

The adverbials form a large and heterogeneous class, also sentence and phrase adverbials are not distinguished. So both word-forms *väga* ('very') and *kiiresti* ('quickly') get the label ADVL in the sentence *Ta jooksis väga kiiresti* ('S/he ran very quickly').

Deeper syntactic analysis is the goal of the next grammar module, a module for building dependency trees.

During the surface syntactic analysis, first all possible labels are added depending on the part-of-speech tag and grammatical categories. Then the syntactic labels that do not conform with other labels or morphological information present in the same clause are deleted one by one. For example, a noun in partitive case form gets the label of the direct object during the initial mapping phase, but it also gets several other syntactic labels. The object label is deleted, if the finite verb in that clause is an intransitive one or is a verb that under certain circumstances takes only a total object[1] (i.e. an object in genitive or nominative case) or if the same clause contains a noun with non-ambiguous object reading and the word-form under consideration is not in a coordinating relation with that.

The module for surface syntactic analysis comprises ca. 1300 rules. Experiments on a manually annotated 9500-token corpus showed that the recall of the whole syntactic analysis (including morphological disambiguation) was 92.9% and precision 69.3%; the error rate was 7.1%. It means that 7.1% of tokens don't get the correct label and 30–31% of the added labels are either superfluous or erroneous.

The majority of errors occur in annotating objects, subjects and predicatives as they can be coded using the same morphological cases. A noun in nominative case form can be a subject, an object or a predicative. A noun in genitive case form can be an object (only in singular) or a genitive attribute. A noun in partitive case form can be a subject, object, predicative, a modifier of a quantifier. Also, the nouns in

[1]Grammatical aspect in Estonian has not developed into a consistent grammatical category, but it emerges in the object case alternation. One can read about the complicated system of Estonian object case alternation in [5, pages 96–97].

nominative, genitive or partitive case can act as adverbials, appositions, belong to the adposition phrase or perform some less observed roles in the sentence.

A substantial amount of non-solved ambiguity in the output is caused by the indiscernibility of adverbials and adverbial attributes. The problem is similar to pp-attachment, e.g. in the sentence *Seal tuleb mees metsast* ('There comes a man from forest') the word-form *metsast* ('from forest') is ambiguous between adverbial and attributive readings.

5 Dependency parser and particle verb detector

Recently, the EstCG parser has been enhanced with dependency rules and this stage is still under development. However, the analysis provided by CG dependency parser helped to develop the first version of Estonian Dependency Treebank, consisting of 400,000 words [6], which in turn gave an opportunity to experiment with statistical parsing methods, namely training and evaluating MaltParser [7] for analysing Estonian texts.

The grammar of dependencies consists of ca 600 rules. The EstCG parser achieves an unlabeled attachment score (UAS) of 77.2%.

We added a special module of rules in order to recognize particle verbs i.e. multiword expressions consisting of a verb and an adverbial particle, also called phrasal verbs in more general terms. The module for identification of Estonian particle verbs consists of a grammar of approximately 500 rules and a thorough lexicon for 70 particles and corresponding lists of verbs. As our results indicate, our lexicon- and rule-based approach can be regarded as successful. More than 95% of particle verbs receive correct analysis at the shallow syntactic level and 95–100% of particle verbs get correct dependency relations (i.e. the particles get combined with correct verbs), what makes it possible to use annotated data for practical linguistic purposes.

In the following example (see Figure 1) there are two different correct translations of the sentence depending on the choice of taking the *üle* ('over') as a preposition (2a) or as a part of a particle verb *üle mängima* ('to outplay') (2b):

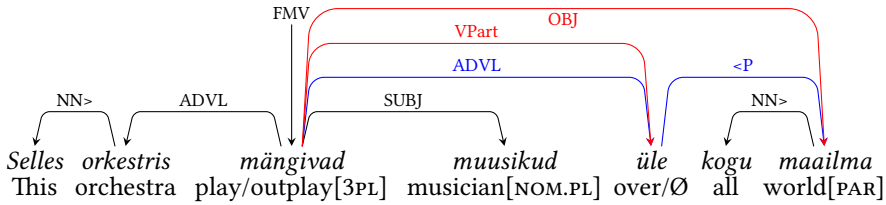

Figure 1: Two alternative analyses of the sample sentence.

(2) a. 'The musicians around the world are playing in this orchestra.'
b. 'The musicians are outplaying all the world in this orchestra.'

6 Statistical parser

For our first experiments with statistical analysis we have selected MaltParser [7] since it has been successfully employed for a wide range of languages, including morphologically rich languages with relatively small treebanks (for example, Latvian and Lithuanian). In addition, MaltParser includes the MaltOptimizer system [8] which helps the end user to select the appropriate parameters and parsing algorithm without having expert knowledge on underlying methods.

First, we transformed the texts from the CG format to the CoNNL-X format. As the regular set of POS tags consists of 15 tags, there is also an option to employ 22 fine-grained POS tags. Most of morphological description has been retained except valency information (e.g. intransitivity of verbs). The syntactic labels remain same as in the EstCG annotation (27 labels), except that the main verb of the main clause (or the head of the verbless clause) gets the label ROOT.

Only the part of the treebank that was double-checked at that point of time (191,000 tokens, 13,310 sentences) was used for statistical parsing. Half of the corpus consists of newspaper texts, while the other half contains fiction and scientific texts. All the sentences have been manually morphologically disambiguated. Every 5th sentence was moved to the testing part of corpora, so the training set consisted of 153,471 tokens. We used MaltOptimizer to find most appropriate training model and parameters. The tool suggested to use Covington-Non-Projective algorithm and a specific feature model.

The preliminary results gave labeled attachment score (LAS, the label and relation link are both correct) 83.6% on 37,959 tokens. This result includes the analysis of punctuation marks (which is a trivial task) and non-sentential constructions like passages in foreign languages, chemical formulas or bibliographical references in scientific texts annotated by label NONE.

After excluding punctuation marks and non-sentential constructions from the analysis, the LAS decreased to 80.3% (31,434 tokens). Also, we observed the unlabeled attachment score (UAS) of 83.4% and the label accuracy (LA) of 88.6%.

We have conducted several experiments on running Maltparser along with EstCG parser: using syntactic information provided by EstCG parser as input for Maltparser or applying special fixing rules to the output of Maltparser. These improved overall performance by 1% [6].

7 Language pipeline

In order to make language technology easier to use for people who are not at home in the command line programs, there is also a graphical web interface for executing annotation workflows - Keeleliin[2] (Language Pipeline). In this interface, it is possible to combine different modules, such as morphological disambiguation or dependency annotation (e.g. picking either Constraint grammar or MaltParser) into reusable workflows that are then executed in the server (see Figure 2). It is also possible to share prepared workflows with other users, so that users with little knowledge about the underlying structure can also use Keeleliin to annotate their texts with different syntactic workflows with no need to install anything beforehand.

At the moment Keeleliin is still in development, so the majority of modules will not be inserted until 2016, as the respective web services are made available. The current version is already open for testing to academic users.

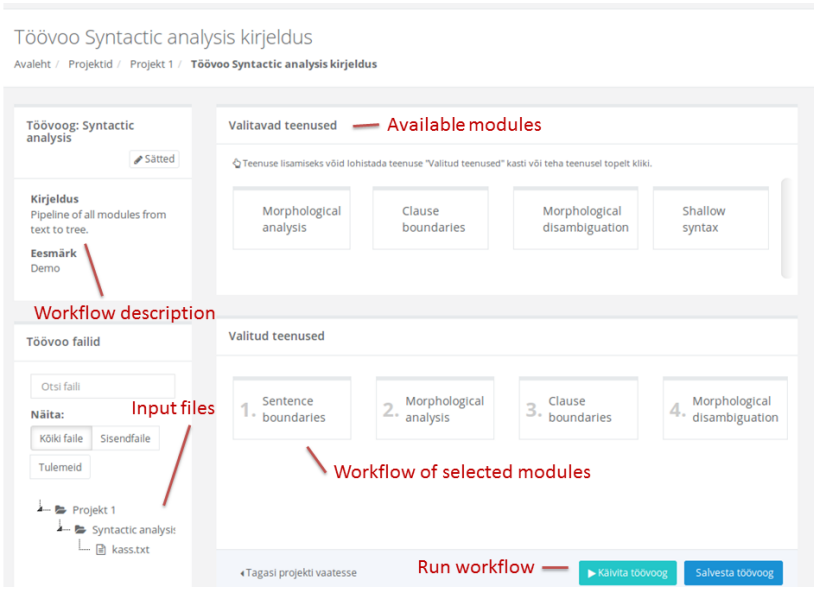

Figure 2: Creating a workflow in the language pipe web service. View of available modules is restricted to those that accept the output format of already selected modules.

[2] `http://keeleliin.keeleressursid.ee`.

8 Corpora and treebanks

The initial versions of the EstCG parser were developed basing on the linguistic knowledge as presented in a descriptive grammar of Estonian [9] and a small experimental test and development corpus (12,000 words). In order to improve the coverage of the rule-based CG parser and to experiment with a machine learning based parser, creating a larger manually annotated corpus was essential.

We succeeded to get funding for creating an Estonian Dependency Treebank and completed its first version by the end of 2014 [10]. The treebank contains approximately 400,000 tokens and is annotated for part of speech, morphological description, syntactic functions and dependency relations.[3]

Figure 3 depicts an Estonian sentence *Hommikul püüdis kass kinni kena paksu hiire* ('In the morning, the cat caught a nice fat mouse'). For every word in the sentence there is a separate row for its analysis. It begins with a lemma, followed by an inflectional ending, POS tag and morphological description.[4] The syntactic function labels begin with @ and tags indicating dependency relations with #.

```
"<s>"
"<Hommikul>"
     "hommik" Ll S com sg ad cap @ADVL #1->2                morning
"<püüdis>"
    "püüd" Lis V main indic impf ps3 sg ps af @FMV #2->0    caught
"<kass>"
    "kass" L0 S com sg nom @SUBJ #3->2                      cat
"<kinni>"
    "kinni" L0 D @Vpart #4->2                               verbal particle
"<kena>"
    "kena" L0 A pos sg gen @AN> #5->7                       nice
"<paksu>"
    "paks" L0 A pos sg gen @AN> #6->7                       fat
"<hiire>"
    "hiir" L0 S com sg gen @OBJ #7->2                       mouse
"<.>"
    "." Z Fst CLB #8->7
"</s>"
```

Figure 3: Sample sentence "In the morning, the cat caught a nice fat mouse."

In order to join in an international effort and to make available the Estonian Dependency Treebank with a cross-linguistically consistent treebank annotation for many languages we have started with conversion of the afforementioned treebank to the Universal Dependencies [11] annotation scheme.[5]

Perhaps there is no better method to test a program for linguistic analysis than

[3] It is freely available from `https://github.com/EstSyntax/EDT`.

[4] explained in detail in `http://www.cl.ut.ee/korpused/morfliides/seletus.php?lang=en`.

[5] `https://github.com/EstSyntax/EstUD`.

large-scale corpus annotation; at least we decided to test our tools this way.

There exists a relatively big corpus of contemporary Estonian.[6] A subcorpus of the afforementioned big corpus (Balanced Corpus, 15 million tokens) was parsed using the CG surface-syntax rules. Resulting language resource is available in two ways: one can query the corpus using corpus query interface at Keeleveeb[7] or one can obtain the full parsed corpus at request.

9 Conclusions and future work

The plans for the near future include experiments for combining rule-based CG parser and MaltParser and also experimenting with other statistical parsers, e.g. Mate [12] or LingPars [13].

We have already started converting the Estonian Dependency Treebank to Universal Dependencies annotation scheme.

Building a morphosyntactic and syntactic analyser or parser can be an interesting task per se and building large syntactically annotated corpora promotes both language technology and linguistic research. But of course our aim is also to foster using Estonian Constraint Grammar in applications.

Among those one could mention language learning programs Oahpa! and Vasta! developed at Giellatekno [14, 15] – programs using linguistic tools for generating new tasks for language learner and testing the student's answer, enabling more flexibility for the generated tasks and the possible answers and more deliberate and precise feedback to the student accordingly to particular linguistic issues relevant for the student's answer. Estonian Oahpa! and Vasta! are currently under development [16]. Another system where we are planning to employ Estonian Constraint Grammar is rule-based machine translation platform Apertium [17].

One can test our demo version of the syntactic parser at `https://korpused.keeleressursid.ee/syntaks` or install it as an open-source software.[8]

Acknowledgements

This work was supported by Estonian Ministry of Education and Research (grant IUT20-56 "Eesti keele arvutimudelid /Computational models for Estonian") and National Programme for Estonian Language Technology.

[6] `http://www.cl.ut.ee/korpused/segakorpus/`.

[7] `http://www.keeleveeb.ee`.

[8] `https://github.com/EstSyntax/EstCG`

References

[1] Heiki Jaan Kaalep. An Estonian Morphological Analyser and the Impact of a Corpus on Its Development. *Computers and the Humanities*, 31(2):115–133, 1997.

[2] Fred Karlsson, Atro Voutilainen, Juha Heikkilä, and Arto Anttila. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Mouton de Gruyter, Berlin, 1995.

[3] Eckhard Bick and Tino Didriksen. CG3 Beyond Classical Constraint Grammar. In *Proceedings of the 20th Nordic Conference of Computational Linguistics NODALIDA 2015*, pages 31–40, 2015.

[4] Kadri Muischnek, Kaili Müürisep, and Tiina Puolakainen. Estonian particle verbs and their syntactic analysis. In Z. Vetulani and H. Uszkoreit, editors, *Human Language Technologies as a Challenge for Computer Science and Linguistics: 6Th Language & Technology Conference Proceedings*, pages 338–342. Adam Mickiewicz University, 2013.

[5] Mati Erelt, editor. *Estonian Language*, volume 1 of *Linguistica Uralica Supplementary series*. 2003.

[6] Kadri Muischnek, Kaili Müürisep, and Tiina Puolakainen. Dependency Parsing of Estonian: Statistical and Rule-based Approaches. In Andrius Utka, Gintare Grigonyte, Jurgita Kapociute-Dzikiene, and Jurgita Vaicenoniene, editors, *Baltic HLT*, volume 268 of *Frontiers in Artificial Intelligence and Applications*, pages 111–118. IOS Press, 2014.

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

[8] Miguel Ballesteros and Joakim Nivre. MaltOptimizer: An Optimization Tool for MaltParser. In Walter Daelemans, Mirella Lapata, and Lluís Màrquez, editors, *EACL*, pages 58–62. The Association for Computer Linguistics, 2012.

[9] M. Erelt, R. Kasik, H. Metslang, H. Rajandi, K. Ross, H. Saari, K. Tael, and S. Vare. *Eesti keele grammatika II. Süntaks*. Eesti TA Keele ja Kirjanduse instituut, 1993.

[10] Kadri Muischnek, Kaili Müürisep, Tiina Puolakainen, Eleri Aedmaa, Riin Kirt, and Dage Särg. Estonian Dependency Treebank and its annotation scheme. In

Verena Henrich et al., editors, *Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13)*, pages 285–291. University of Tübingen, 2014.

[11] Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 92–97, 2013.

[12] Bernd Bohnet. Top Accuracy and Fast Dependency Parsing is not a Contradiction. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING*, pages 89–97. Tsinghua University Press, 2010.

[13] Eckhard Bick. LingPars, a Linguistically Inspired, Language-Independent Machine Learner for Dependency Treebanks. In Lluís Màrquez and Dan Klein, editors, *CoNLL*, pages 171–175. ACL, 2006.

[14] Lene Antonsen, Saara Huhmarniemi, and Trond Trosterud. Interactive pedagogical programs based on constraint grammar. In *Proceedings of the 17th Nordic Conference of Computational Linguistics*, volume 4 of *NEALT Proceedings Series*, pages 10–17, 2009.

[15] Lene Antonsen, Saara Huhmarniemi, and Trond Trosterud. Constraint Grammar in Dialogue Systems. In *Proceedings of the NODALIDA 2009 workshop Constraint Grammar and robust parsing*, volume 8 of *NEALT Proceedings Series*, pages 13–21, 2009.

[16] Heli Uibo, Jaak Pruulmann-Vengerfeldt, Jack Rueter, and Sulev Iva. Oahpa! Õpi! Opiq! Developing free online programs for learning Estonian and Võro. In *Proceedings of the 4th workshop on NLP for Computer Assisted Language Learning at NODALIDA 2015*, volume 26 of *NEALT Proceedings Series*, pages 51–64. Linköping University Electronic Press, 2015.

[17] Mikel L. Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, 2011.

Rule-Based and Statistical Morph Segments in English-to-Finnish SMT

Tommi A Pirinen
tommi.pirinen@computing.dcu.ie
ADAPT Centre—School of Computing, DCU

Antonio Toral
atoral@computing.dcu.ie
ADAPT Centre—School of Computing, DCU

Raphael Rubino
rrubino@prompsit.com
Prompsit Language Engineering S.L.

January 3, 2016

Abstract

Morphological segmentation is recognised as a potential solution in statistical machine translation (SMT) to deal with data sparsity posed by morphologically complex languages like all Uralic languages. Two approaches have been used in the literature, rule-based and statistical, but always in isolation. In addition, previous work has failed to bring significant improvement and conclusive analyses of the effects of segmentation. In this paper we use both rule-based and unsupervised approaches to segmentation jointly and aim to find out where they excel and where they fail. Our case study is on English-to-Finnish using the datasets provided at the WMT 2015 shared task. We present a comprehensive evaluation of SMT systems built with different segmenters including: intrinsic evaluation, MT automatic metrics, MT human evaluation and MT linguistic evaluation. In terms of automatic metrics, the best system is the one that combines both rule-based and unsupervised segmentations, outperforming an unsegmented system by 1.08 BLEU and 3.64 TER points. Human evaluation shows that the outputs produced by an SMT system with rule-based segmentations are preferred over those of the system that uses unsupervised segmentations.

1 Introduction

Morphologically complex languages are well known to cause problems for contemporary *statistical machine translation* (SMT) systems. *Morphological segmentation*, in which words are divided into sub-word units prior to training, has been a popular method to deal with morphologically complex languages in SMT. In this regard, [1] presents a comprehensive overview of the topic. However, despite a lot of effort put into the use of morphological segmentation in SMT, automatic evaluation for morphologically rich languages, to the extent of the richness of Uralic languages rather than most Indo-European, have yielded modest improvements to downright negative results [2].

In this paper we aim to lay out a pathway on this problem by extensively studying various segmentation schemes and the errors they introduce and avoid. We systematically evaluate and compare segmentations produced by the two most widely used approaches to morphological segmentation: rule-based and statistical. Our aim is then to find out where each of these approaches excels and where they fail, and whether their joint use can be beneficial. To have a systematic and thorough evaluation, we use four evaluation schemes for the segmentations: intrinsic test of segmentation quality against a gold standard, the automatic and human evaluation of segmented MT systems, automatic evaluation of linguistic features and translation model features.

2 Morphological Segmentation

Morphological segmentation is a well-established technique in SMT and there is a large amount of related work to consider: [3, p. 324] provides an extensive reading list on the topic. It is important to note that the term *morphological segmentation* is often used for a number of different techniques, which we believe are not comparable, or relevant to Uralic SMT. For example, segmenting Chinese non-space separated sentences or Vietnamese syllables into words is not considered in this article (we associate this to *tokenisation*), nor changing word-forms into structure consisting of abstract dictionary word and suffix identifiers (i.e., *morphological analysis*). The segmentation we consider involves solely finding segmentation points within a word token. There are two approaches to segmentation that we study in this article: unsupervised statistical and rule-based. The state-of-the-art of statistical segmentation has been determined in *MorphoChallenge* shared tasks[4]. In rule-based morphology, researchers generally concentrate on the higher level linguistic morphological analysis rather than on plain segmentation, and thus there is no comprehensive evaluation of the state of the art in the *segmentation* task. For Uralic languages it makes sense to

follow the prevalent *Finite State Morphology* [5]. These are the frameworks we use in this paper.

Much of the prior work in Finno-Ugric languages, mainly for Finnish and Estonian, is based on using unsupervised morphological segmentation only [2, 6, 7, 8, 9]. One of the new emphases presented in this article is the comparison and combination of rule-based morphologies to unsupervised segmentation. [7] make use of rule-based segmentation in determining their baseline but they carry on to use only the better-performing unsupervised segmentation in their actual experiments.

The majority of prior work that shows more optimistic scores concerns language families whose morphological complexity is considerably simpler than that of Uralic languages, e.g. Slavic [10] and Germanic [11]. German is mainly pre-processed for compound simplification; Finnish in comparison is productive both for compounding and for regular inflection. The closest language that has been extensively studied in previous work is Turkish [12, 13]. In addition, Basque [14] is comparable to Finnish in terms of morph distributions after segmentation.

Our approach to morphological segmentations is done in pre-processing and post-processing steps that perform addition and deletion of segmentation points operating on segmentation markers. We compare two approaches to morphological segmentation: rule-based and unsupervised. The methods are implemented using the following software: HFST [15][1] for rule-based and Morfessor [16] for unsupervised segmentation. For both approaches we create two different segmentation models: for rule-based we select segmentation points to match annotations for word-segmentation (referred to in the rest of the article with the code-name `hfst-comp`) and morph-segmentation (`hfst-morph`). For unsupervised versions we use Morfessor 2.0 Baseline (`morfessor`) and Morfessor Flatcat (`flatcat`) [17]. Our experiments include morphological segmentation methods used separately as well as in system combination.

For rule-based morphological segmentation we developed a segmenter on top of omorfi [18],[2], an open-source implementation of weighted finite-state morphology. Omorfi's morphological segmenter has a number of segments annotated: `MB` for morph boundaries, `DB` for derivation boundaries, `WB` and `wB` for word boundaries and `STUB` for other stemmer-type boundaries. We use these to produce two segmented versions: one where all `WBs` and `wBs` are turned into segmentation points and one where `WBs`, `wBs` and `MBs` are.[3] These are called *compound* and *morph* segmentations, respectively. Rule-based morphological segmentation is ambiguous and we use the

[1] `http://hfst.sf.net`

[2] `http://github.com/flammie/omorfi/`

[3] The two remaining types of segmentation points (`DB` and `STUB`) are discarded as they are not relevant for our task.

Segmenter	text
None	kuntaliitoksen selvittämisessä
hfst-comp	'kunta→ ←liitoksen selvittämisessä
hfst-morph	kunta→ ←liitokse→ ←n selvittämise→ ←ssä
Flatcat	kun→ ←tali→ ←itoksen selvittämis→ ←essä
Morfessor	kun→ ←ta→ ←liito→ ←ksen selvittä→ ←misessä
Gloss	municipality+annexation.GEN examination.INE
Translation	examination regarding municipal annexation

Table 1: Different segmentation methods.

1-best result. For word-forms not recognised by the morphological analyser, no segmentation points are produced.

Unsupervised morphological segmentation is based on statistically likely segmentation points found from an unannotated training corpus when trying to iteratively optimise a given function. In the case of `morfessor`, the optimisation function is based on minimum description length (MDL), so the aim of the algorithm is to minimise the vocabulary size of the output, i.e. to find the segmentation with the lowest number of different morphs. `Flatcat` extends this by using hidden markov models (HMM) and context to create classes for the morphs: stems, suffixes, prefixes and non-morphemes. Thus, for example, if there is a morph identified as a common suffix, it should be more unlikely for it to be split off from the beginning of a word, even if doing so would result in a lower set of distinct morphs as per MDL. For each word we select the 1-best segmentation.

Examples of the different segmenters are shown in Table 1. The semantics of the gloss can be most easily traced to match the `hfst-morph` version.

3 Experimental Setup

3.1 MT Tools and Datasets

Our experimental setup matches the one used in [19].The training, development and test data set used in our experiments are obtained from the WMT 2015 shared task.[4] In this shared task, participants train and apply MT systems on pre-defined corpora for training, development and testing, the domain of the latter being news. Our translation models (TMs) are trained on the Europarl v8 Finnish–English parallel corpus and the language models (LMs) benefit from the additional shuffled news monolingual corpus (*News Crawl: articles from 2014*). All typical pre-processing steps are performed. All the scripts used to pre-process the data are available with the MOSES distribution [20]. Finally, we generate segmented training sets for both parallel and

[4] `http://www.statmt.org/wmt15/translation-task.html`

monolingual corpora following the segmentation methods described in Section 2. The segmented SMT systems output segmented Finnish text, thus a post-processing step (morph-joining) is performed to obtain the final translation.

We assess empirically the performance of two LM training methods: concatenation of parallel and monolingual corpora, or linear interpolation of two individual LMs based on the minimisation of the perplexity obtained on the development set. We observe that segmented LMs reach better results with the concatenation method, while the word-based LM benefits from the interpolation approach. We also experiment enriching the phrase-based SMT pipeline with additional components such as multiple reordering models (joint use of word-, phrase-based [21] and hierarchical [22] reordering models), Operation Sequence Model (OSM) [23] and neural language models [24] such as the Bilingual Neural LM (BiNLM) [25]. Again, we empirically evaluate adding these models to our SMT systems based on the development set. We observe an improvement of the results with the three reordering models for segmented and non-segmented systems, while OSM and BiNLM yield improvements to the word-based system only.

3.2 Segmentation

For our rule-based segmentation we used the segmentation automaton `omorfi.segment.hfst` from omorfi version 20150326 by simply rewriting the word boundary and morph boundary markers into arrows and any other boundaries into zero-length strings as described in Section 2. For unsupervised segmentation we used `morfessor` 2.0.2-alpha and `Flatcat` 1.0.4 trained on the Finnish side of the europarl-v8 corpus, using default settings except for the fact that we remove the segmentation points of non-morphemes in `Flatcat`.

Accordingly, we build four SMT systems using the aforementioned segmentations: `hfst-morph`, `hfst-comp`, `morfessor` and `Flatcat`. In addition, we explore the joint use of more than one segmentation by means of system combination with `MEMT` [26].[5] We try three different combinations:

a) `combo-unsup`, where we attempt to build the most competitive system using only unsupervised methods. This combines `morfessor`, `Flatcat` and the baseline SMT system (unsegmented). b) `combo-rb`, where the attempt is on building the most competitive system using rule-based methods. This combines `hfst-morph`, `hfst-comp` and, again, the baseline. c) `combo-all`, where the aim is to build the most competitive system using both unsupervised and rule-based methods. This com-

[5]We use default settings except for the radius (5, default is 7), following empirical results obtained on the devset.

bines all the five systems: `morfessor`, `Flatcat`, `hfst-morph`, `hfst-comp` and the unsegmented baseline system.

3.3 Evaluation

For intrinsic evaluation of segmentation accuracy we used morphochallenge 05 [27] gold test data and `evaluation.perl`,[6] since it most closely resembles the segmentation setup of our SMT setting (i.e. no annotations or deep analysis).

Automatic evaluation of MT outputs was performed using the following evaluations scripts: `mteval13a.pl` for BLEU [28], `tercom-7.25.jar` for TER [29][7] and `meteor-1.5.jar` for METEOR [30].[8]

For human evaluation we used the Appraise toolkit [31].[9] The evaluation was conducted by three Finnish native speakers with a background in Computational Linguistics. This evaluation is inspired by the human evaluation conducted as part of the translation task in WMT; the evaluators are given a set of outputs coming from different systems and they are asked to rank them according to their quality (ties are allowed).

Finally, for the linguistic analysis, we used the morphological fluency classifications of [7], basing on those, we developed a new automatic evaluation script using omorfi analyses.

4 Evaluation

4.1 Segmentation Evaluation

In order to evaluate how the quality of segmentation –as defined by the gold standard written by a linguist– affects the final MT results, we evaluated our segmentation methods on the gold standard provided at the morphochallenge 2005 shared task on morphological segmentation. The results are shown in Table 2.

As expected, the results of the linguistic analyser `hfst-morph` matches the linguistic gold standard quite well, with unsupervised methods performing considerably worse. The linguistic analyser `hfst-comp` of course does not obtain high recall in segmenting all boundaries as it only aims to select a very specific subset of those (i.e. compound boundaries or stem-stem boundaries in unsupervised terms).

[6] `http://research.ics.aalto.fi/events/morphochallenge2005/data/evaluation.perl`
[7] `https://www.cs.umd.edu/~snover/tercom/tercom-0.7.25.tgz`
[8] `https://www.cs.cmu.edu/~alavie/METEOR/download/meteor-1.5.tar.gz`
[9] `http://github.com/cfedermann/Appraise`, commit `9b643ae55647...`

System	F-Measure	Precision	Recall
Flatcat	54.04 %	76.04 %	41.91 %
hfst-comp	43.82 %	97.63 %	28.25 %
hfst-morph	86.32 %	92.39 %	81.00 %
Morfessor	53.89 %	71.01 %	43.42 %

Table 2: Results of the intrinsic evaluation of the four segmentation methods

	Dev. set			Test set		
System	BLEU	TER	METEOR	BLEU	TER	METEOR
Baseline	0.1577	0.7479	0.3069	0.1402	0.7609	0.2997
Flatcat	0.1481	0.7699	0.3060	0.1387	0.7712	0.3001
hfst-comp	0.1541	0.7415	0.3019	‡0.1471	0.7405	0.2977
hfst-morph	0.1575	0.7381	0.3050	‡0.1451	0.7476	0.2986
Morfessor	0.1434	0.7868	0.2987	0.1343	0.7882	0.2942
combo-unsup	0.1595	0.7267	0.3031	0.1408	0.7367	0.2937
combo-rb	0.1569	0.7179	0.3002	‡0.1459	**0.7214**	0.2959
combo-all	**‡0.1638**	**0.7160**	**0.3074**	**‡0.1510**	0.7245	**0.3011**

Table 3: Automatic evaluation of MT systems built with different segmentation methods. The baseline is unsegmented. Statistical significance tests (paired boostrap resampling) run on BLEU (‡ $p = 0.01$).

4.2 MT Automatic Evaluation

We evaluate MT systems built on the training data segmented using each of the four segmentation methods with the three aforementioned state-of-the-art automatic metrics: BLEU, TER and METEOR (see Table 3).

We observe that systematically the system combination of all segmentation models performs the best, with the exception of TER on the test set, where the combination of rule-based and baseline methods results in the best score. Furthermore we note that the rule-based combination beats the unsupervised combination on the test set, but on the dev. set the unsupervised combination is slightly better (except for TER). Contrasting this to single system scores, which are worse across the board, we can conclude that each individual system contributes different parts to the output produced by the system combinations.

4.3 MT Human Evaluation

We performed human evaluation of the translations with 3 native speakers ranking the sentences. We produced the final rankings from the human evaluation judgements using the TrueSkill method adapted to MT evaluation [32] with its implementation in WMT-Trueskill,[10] following its usage at WMT15.[11] Namely, we run 1,000 iterations of

[10] `https://github.com/keisks/wmt-trueskill`

[11] `https://github.com/mjpost/wmt15`

rankings followed by clustering ($p = 0.95$). Results are shown in Table 4.

#	Score	Range	System
1	0.529	1-2	combo-all
2	0.414	1-2	combo-rb
3	-0.943	3-3	combo-unsup

Table 4: The results of human evaluation by three native speakers with background in computational linguistics as measured by TrueSkill.

The results show that human annotators, in general, prefer either the combination of all systems (`combo-all`) or the rule-based combination (`combo-rb`) over the purely unsupervised combination (`combo-unsup`). More specifically, `combo-all` is the best performing system (0.529), closely followed by `combo-rb` (0.414) with `combo-unsup` clearly performing worst (-0.943). In terms of significance (column range), at $p = 0.95$, `combo-all` and `combo-rb` are in the same cluster (range 1-2), thus meaning neither of the two is significantly better than the other, while `combo-unsup` is in a different cluster (range 3-3), meaning its performance is significantly worse, compared to the other two systems.

The inter-annotator agreement as shown by Fleiss' $\kappa = 0.26$ suggests that there is a mild tendency of agreement between the annotators. This is in the same range as agreement at the WMT 2014 shared task [33].

4.4 MT Linguistic Evaluation

In order to evaluate the fluency of the translations, [7] suggest using morphological analysis to determine translation issues over a set of linguistic criteria. We measure the recall of the following constructions in the MT output as compared to the reference translation: a) *Noun marking* (NM), for nouns with case different than nominative. b) *Possession* (POSS) for any word with possessive suffix. c) *Noun-adjective agreement* (NAA) for sequences of adjective-noun, where case is shared. d) *Subject-verb agreement* (SVA) for sequences of noun-verb, where number is shared. e) *Transitive object* (TP) for sequences of verb-noun, where case is accusative or partitive. f) *Postposition* (PP) for sequences of adposition-noun, where case is genitive. Of these tests, NM and POSS pertain to single tokens and NAA and SVA sequences of two tokens, whereas TP and PP scan the whole context and are thus less reliable.

There is no clear tendency for any single system to be the best in morpho-syntactic fluency as measured by these tests, e.g., it seems that combo and rule-based systems will recover NM and PP better but unsupervised matches the most POSS forms. An additional error analysis should reveal the effects of missing forms.

The translation models (phrase and reordering tables) present different charac-

System	NM	TP	POSS	NAA	SVA	PP
Frequency	10.03	1.48	1.40	0.92	0.76	0.14
Baseline	71.94	**39.24**	54.83	31.62	45.22	21.97
Unsup	72.38	37.24	**60.36**	33.80	**46.78**	21.94
Rule-based	72.95	36.32	56.65	32.42	45.13	**29.49**
Combo	**73.34**	36.78	56.06	**34.37**	43.87	24.26

Table 5: Linguistic fluency of translated sentences compared to the reference translation. The metric is F_1 of the analysed MT output compared to the analysed reference. Frequency is the number of occurrences of the construction (as automatically detected) in the reference translation per sentence.

	Baseline	Flatcat	hfst-comp	hfst-morph	Morfessor
#Phrase-pairs (M)	84.6	86.2	86.8	82.6	85.5
Fertility	0.786	1.029	0.856	1.151	1.047
Lexical ambiguity	43.3	29.3	36.7	24.0	28.7

Table 6: Statistics extracted from the trained SMT models, the first row indicates the number of phrase-pairs (millions), the second row contains the word-level fertility measured (English→Finnish) and the third row indicates the average number of target words aligned with each source word calculated at the corpus level.

teristics whether the training data was segmented or not, but also according to the different segmentation methods. For instance, depending on the segmenter, the number of extracted and scored phrase-pairs in the phrase table differs, as shown in the first row of Table 6. These results show that segmenting the data leads to a larger amount of phrase-pairs extracted, which is related to the differences in alignment points found by MGiza. Only hfst-morph leads to a lower amount of extracted phrase-pairs. The performance of each segmentation method according to Table 2 is apparently inversely correlated with the number of phrase-pairs: the highest the *f-score*, the lower the amount of phrase-pairs.

An interesting phenomenon is observed on the word-level fertility from English to Finnish (how many Finnish words are generated by one English word), as shown in the second row of Table 6. These scores indicate to which extent the segmentation leads to ambiguous alignments. These results are supported by the lexical ambiguity scores shown in the third row of the same Table 6. The lexical ambiguity scores are obtained by averaging the number of target words aligned with a source word with a non-null probability at the corpus level, the lower the score the better. We can see that the fertility scores are inversely correlated with the lexical ambiguity. These notable differences between our SMT systems lead to variable translations from the same source sentences depending on the SMT system used. To illustrate these differences, we show some translation examples in Table 7. As shown in the examples, the morph-based translation methods can come up with a correct compound or

Source	The water should be conducted to a fixed drain or rain water network, and not just into a container.
Baseline	Vesi pitäisi johtaa kiinteään viemäriin tai että sadevesi verkkoon, eikä vain astian.
hfst-morph	Vesi pitäisi hoitaa kiinteään viemäriin tai **sadevesiverkostoon**, eikä vain astiaan.
Reference	Vesi pitäisi johtaa kiinteään viemäriin tai sadevesiverkostoon, eikä vain astiaan.
Source	the news is reported by BBC, who refers to governmental sources.
Baseline	uutinen on raportoinut BBC, joka viittaa valtion lähteistä.
hfst-comp	uutinen on raportoinut BBC, joka viittaa **hallituslähteisiin**.
Reference	asiasta kertoo BBC hallituslähteisiin viitaten.

Table 7: Examples of translations where words in bold are generated at decoding time without being observed in the training data.

morphological combination, not found in training data, e.g., the term *sadevesiverkostoon* (sewage network) rather than the un-idiomatic and grammatically questionable *sadevesi verkkoon* (network of rainwater). In the second example the generated compound *hallituslähteisiin* matches the idiomatic compound for 'governmental sources' whereas the baseline results in the less idiomatic *valtion lähteistä* 'sources from the state' and gets the case wrong.

5 Conclusions and Future Work

This paper has explored the joint use of different segmentations methods in SMT for the English-to-Finnish language direction. We have shown that both rule-based and unsupervised morphological segmentation methods are useful as they are complementary. While morphological segmentation approaches in isolation do not result in substantial increments of performance according to automatic MT metrics, using different segmentations jointly does lead to notable increments of performance (+1.08 BLEU and -3.64 TER compared to an unsegmented system).

For future work it might be interesting to see if some of the more advanced morphological processing methods. For example abstraction of morphemes and morph prediction method used by [7] has been shown to improve English-Finnish translation. Likewise, using n-best lists and re-ranking with morphs—e.g. in style of [34, 8]—could improve the final system even more.

Regarding the automatic system that uses a morphological analyser to check for linguistic similarity, for future research it would be interesting to couple this with a syntactic parsing in order to better recognise long-span features such as verb argument structures.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement PIAP-GA-2012-324414 (Abu-MaTran).

We thank Inari Listenmaa, Hege Roivainen and the student organisiation of linguistics in University of Helsinki for human evaluation.

References

[1] Ann Clifton. *Unsupervised morphological segmentation for statistical machine translation.* PhD thesis, Applied Science: School of Computing Science, 2010.

[2] Sami Virpioja, Jaakko J. Väyrynen, Mathias Creutz, and Markus Sadeniemi. Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner. In *Proceedings of the Machine Translation Summit XI*, pages 491–498, Copenhagen, Denmark, September 2007.

[3] Philipp Koehn. *Statistical machine translation.* Cambridge University Press, 2009.

[4] Mikko Kurimo, Mathias Creutz, and Ville Turunen. Overview of morpho challenge in clef 2007. In *Working Notes of the CLEF 2007 Workshop*, pages 19–21, 2007.

[5] Kenneth R Beesley and Lauri Karttunen. *Finite state morphology.* Center for the Study of Language and Inf, 2003.

[6] Mark Fishel and Harri Kirik. Linguistically motivated unsupervised segmentation for machine translation. In *LREC*, 2010.

[7] Ann Clifton and Anoop Sarkar. Combining morpheme-based machine translation with post-processing morpheme prediction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 32–42, 2011.

[8] Minh-Thang Luong, Preslav Nakov, and Min-Yen Kan. A hybrid morpheme-word representation for machine translation of morphologically rich languages. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 148–157, 2010.

[9] Adrià De Gispert, Sami Virpioja, Mikko Kurimo, and William Byrne. Minimum Bayes risk combination of translation hypotheses from alternative morphological decompositions. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 73–76, 2009.

[10] Maja Popovic and Hermann Ney. Towards the use of word stems and suffixes for statistical machine translation. In *Proceedings of LREC*, 2004.

[11] Fabienne Cap, Alexander Fraser, Marion Weller, and Aoife Cahill. How to produce unseen teddy bears: Improved morphological processing of compounds in smt. In *Proceedings of EACL 2014*, 2014.

[12] Kemal Oflazer and Ilknur Durgar El-Kahlout. Exploring different representational units in English-to-Turkish statsistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 25–32, 2007.

[13] Coşkun Mermer. Unsupervised search for the optimal segmentation for statistical machine translation. In *Proceedings of the ACL 2010 Student Research Workshop*, ACLstudent '10, pages 31–36, 2010.

[14] Relevance of different segmentation options on Spanish-Basque SMT, author=de Ilarraza, Arantza Dıaz and Labaka, Gorka and Sarasola, Kepa, booktitle=Proceedings of the EAMT, year=2009.

[15] Krister Lindén, Erik Axelson, Sam Hardwick, Tommi A Pirinen, and Miikka Silfverberg. Hfst—framework for compiling and applying morphologies. *Systems and Frameworks for Computational Morphology*, pages 67–85, 2011.

[16] Sami Virpioja, Peter Smit, Stig-Arne Grönroos, Mikko Kurimo, et al. Morfessor 2.0: Python implementation and extensions for morfessor baseline. 2013.

[17] Stig-Arne Grönroos, Sami Virpioja, Peter Smit, and Mikko Kurimo. Morfessor FlatCat: An HMM-based method for unsupervised and semi-supervised learning of morphology. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*, pages 1177–1185, 2014.

[18] Tommi A Pirinen. Omorfi–free and open source morphological lexical database for Finnish. In *Nordic Conference of Computational Linguistics NODALIDA 2015*, page 313, 2015.

[19] Raphael Rubino, Tommi Pirinen, Miquel Esplà-Gomis, Nikola Ljubešić, Sergio Ortiz Rojas, Vassilis Papavassiliou, Prokopis Prokopidis, and Antonio Toral. Abu-MaTran at WMT 2015 translation task: Morphological segmentation and web crawling. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 184–191, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[20] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180, 2007.

[21] Philipp Koehn, Amittai Axelrod, Alexandra Birch, Chris Callison-Burch, Miles Osborne, David Talbot, and Michael White. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *IWSLT*, pages 68–75, 2005.

[22] Michel Galley and Christopher D Manning. A simple and effective hierarchical phrase reordering model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856, 2008.

[23] Nadir Durrani, Helmut Schmid, and Alexander Fraser. A joint sequence translation model with integrated reordering. In *Proceedings of ACL/HLT*, pages 1045–1054, 2011.

[24] Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. Decoding with large-scale neural language models improves translation. In *EMNLP*, pages 1387–1392, 2013.

[25] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1370–1380, 2014.

[26] Kenneth Heafield and Alon Lavie. Combining machine translation output with open source: The Carnegie Mellon multi-engine machine translation scheme. *The Prague Bulletin of Mathematical Linguistics*, 93:27–36, 2010.

[27] Mikko Kurimo, Mathias Creutz, Matti Varjokallio, Ebru Arisoy, and Murat Saraçlar. Unsupervised segmentation of words into morphemes–challenge 2005: An introduction and evaluation report. In *Proceedings of the PASCAL Challenge Workshop on Unsupervised segmentation of words into morphemes*, 2006.

[28] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318, 2002.

[29] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for machine translation in the Americas*, pages 223–231, 2006.

[30] Satanjeev Banerjee and Alon Lavie. METEOR: an automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[31] Christian Federmann. Appraise: An open-source toolkit for manual evaluation of machine translation output. *The Prague Bulletin of Mathematical Linguistics*, 98:25–35, September 2012.

[32] Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. Efficient elicitation of annotations for human evaluation of machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 1–11, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.

[33] Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA, June 2014.

[34] Christopher Dyer, Smaranda Muresan, and Philip Resnik. Generalizing word lattice translation. Technical report, DTIC Document, 2008.

Exploring Natural Language Processing Methods for Finno-Ugric Langages

Thierry Poibeau
CNRS and Ecole normale supérieure and Université Sorbonne Nouvelle
1 rue Maurice Arnoux, 92120 Montrouge, France
thierry.poibeau@ens.fr

Svetlana Toldova
National Research University "Higher School of Economics"
School of Linguistics
toldova@yandex.ru

November 15, 2015

Abstract

This paper presents some preliminary experiments concerning the automatic processing of Finno-Ugric languages with computers. We present symbolic methods as well as machine learning ones. Given the lack of corpora for some languages, we think finite state transducers may sometimes be the best approach, even if machine learning techniques are nowadays supposed to outperform symbolic methods. We also consider some machine learning approaches that could be valuably applied in this context, more specifically lightly supervised techniques involving reduced sample of annotated data with larger amonts of non annotated data. Lastly we present the LAKME project that will explore new techniques for parsing morphology rich languages.

1 Introduction

The Finno-Ugric language family includes more than 30 languages which are for a large part endangered [1]. Most of these languages are spoken by a declining number of speakers and there is thus a growing interest in "documenting" these languages.

This includes the preservation, normalization and annotation of corpora, as well as the production of reference tools (lexicon, grammars) that can be re-used in various applications[1].

In this paper we present some joint work between the Lattice laboratory at the Ecole normale supérieure in Paris and the National Research University of Moscow Higher School of Economics, to develop resources and techniques for Finno-Ugric languages. We explore symbolic methods (esp. finite state transducers) as well as machine learning ones (unsupervised as well as supervised methods). We think there is a need to adapt methods to the problem since, given the language under consideration, texts can be available or not, and the same applies for dictionaries or annotated data. Lightly supervised methods (i.e. methods requiring a small sample of annotated data as well as larger amounts of non annotated data) are also considered since they seem especially relevant in our case.

The structure of the paper is as follows. We first consider briefly the corpora available in Moscow. We then detail some experiments we have done with finite state transducers and with the morphological analyzer Morfessor. In the last section we detail the LAKME project, which aims at developing parsing techniques for morphology rich languages. We conclude with a few consideration on evaluation and some perspectives.

2 Available Corpora

The University of Moscow as well as the National Research University "Higher School of Economics" conduct regular field work campaigns concerning Finno-Ugric languages spoken in Russia. The data collected (mostly audio data that is then transcribed and analyzed) concern Mari, Komi, Udmurt, Khanty, Erzya and Moksha, among others.

Once transcribed, this data is available as raw text (sometimes with some annotation using the SIL format, see `http://www.sil.org`) but automatic tools would be very useful to assist linguists in this process. Our goal is thus to enrich this data with linguistic annotation so as to make it more visible and more specifically easier to use for researchers interested in a specific linguistic phenomenon.

[1]From this point of view, we share the same goal as several other projects. See, among others, the FinUgRevita project, described at http://www.ieas-szeged.hu/finugrevita/

3 Lexical Analysis through Finite state transducers

Finite state transducers (FST) are widely available and different implementations exist for the easy and quick development of efficient natural language processing systems. One example of such a toolbox for NLP applications is the Unitex platform developed at the University of Marne-la-Vallée in France[2]. This toolbox includes resources for various languages including Finnish: resources for Finnish have been developed at the University of Caen[3]. Unitex is provided with a LGPL license, which means the software is open source and can be used in various contexts without restriction (academic as well as industrial contexts).

It is well known that finite state transducers are especially efficient for word processing as well as for the recognition of local syntactic patterns. Thanks to FST It possible to describe the lexicon of inflected forms of a language based on a list of stems and declension paradigms in a very compact way. Unitex allows such an implementation. Once compiled the system produces a formal lexical analysis of all kinds of linguistic units (words as well as compounds and idioms), along with relevant information attached to the lexical forms (see figure 1).

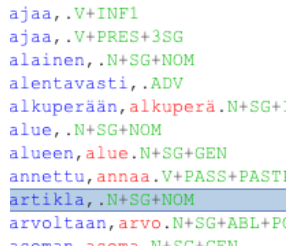

Figure 1: An example of lexical analysis with Unitex (lexical forms in blue, lemma in red, linguistic features in green

Beyond lexical analysis, a typical application is the automatic recognition of local sequences of texts. A typical example is named entity recognition, which includes the recognition of person names, location names as well as dates and more generally any semantic pieces of information relevant for a given application. We have presented in

[2] http://www-igm.univ-mlv.fr/~unitex/
[3] http://www.unicaen.fr/ufr/homme/linguistique/ressources/finnois/

2003 the implementation of such a system for a douzain languages, including Finnish[4] [3]. The idea is now to address less visible Finno-Ugric languages.

FST are interesting in that they make i possible to describe a grammar through a collection of readable graphs. The description is generally compact since the formalism is recursive: a graph can include different subgraphs, as shown on figure 2, where here grey box refers to a subgraph that is called dynamically.

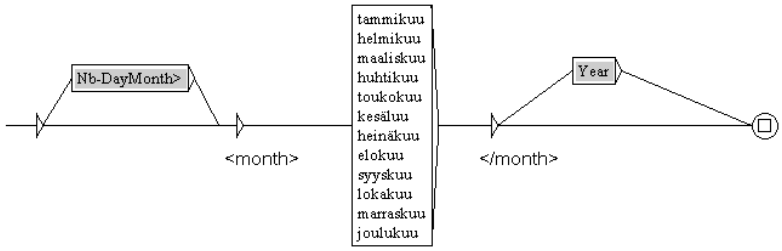

Figure 2: A Unitex graph (here an extract of a grammar recognizing dates in Finnish)

The drawback of FST is the time required to write a hgh coverage grammar as well as the maintenance of such a collection of graph, when its size expands. Machine learning are known to generally give better results for different tasks nowadays but it should also be noted that FST still provides fast and efficient implementations for a number of local linguistic phenomena, when few data is available for training. We thus think that FST remains interesting for endangered languages.

4 Automatic morphological segmentation using Morfessor

We have investigated the automatic segmentation of moksha words with the Morfessor 2.0 software (http://www.cis.hut.fi/projects/morpho/ and https://github.com/aalto-speech/morfessor) [4, 5]. Morfessor uses raw text data and machine learning methods to find words segmentation in natural language. Morfessor can use unsupervised or semi-supervised methods, we tested both.

[4]The implementation was then made with Intex [2], that is no longer maintained. A transfer to another FST toolbox like Unitex would ne quite straighforward.

Our training corpus is composed of an extract of the mokshen pravda and an extract of the wikidumps of the moksha wikipedia (`https://mdf.wikipedia.org/`). We have counted the frequencies of each word written in cyrillic alphabet of those sets of texts, leading to a word list of 120759 types (for 1352317 tokens) .

For our first experiments, we used a test file is composed of only 16 sentences (183 words) from a wiki entry. For the semi-supervised approach we provided to morfessor-train an annotation file with manually segmented words coming from a dictionnary. The main drawback of this data set is that it is composed of non flexionnal forms but we are currently preparing a reference corpus of inflected forms so as to get more relevant results.

Here is for example a sentence in Moksha from our corpus:

> Бабань ям (илякс Бабань озкс) - мокшень тундань озкссь, коза пуромкшнесть аньцек ават ди коза сявондевсть шабатневок.

And here the automatic analysis proposed by Morfessor:

> Баба нь
> я м
> (иля кс
> Баба нь
> озкс
>)
> -
> мокш ень
> тунда нь
> озкс сь ,
> коза
> пуромкшне сть
> ань цек
> ава т
> ди
> коза
> сявондев сть
> шаба тневок .

It is clear that these results are not optimal. We are confident that by providing more information to the system (so as to be able to guide the system with a semi-supervised approach – this is possible since Morfessor is provided with a purely unsupervised as well as with a lightly supervised mode) we will get more accurate results. These experiments are currently on going.

5 Machine learning approaches for Finno-Ugric languages: an overview of the LAKME project

LAKME is a project dedicated to the automatic production of linguistically annotated corpora. Textual corpora are nowadays largely available, including for ancient as well as for under-resourced languages. However, from a linguistic point of view, these corpora are nothing if they are not enriched with linguistic information, allowing the researcher to go beyond purely "surface" patterns. At the same time, machine learning techniques and natural language processing (NLP) have made much progress, so that it is now possible to accurately analyse texts (at least at the morphosyntactic and syntactic level). Most research so far has been done on English (and other Indo-European languages) but much more still needs to be done on other languages (esp. morphology rich languages). This project aims at developing new machine learning methods for text annotation. Targeted languages are Hebrew, French (esp. Medieval French) and Uralic languages.

For Uralic languages, the project will involve researcher from the Lattice laboratory who will develop machine learning methods while the National Research University Higher School of Economics from Moscow will provide the data and some aid for the analysis.

5.1 Previous experience with POS tagging

Directly relevant for this project is the SEM tagger module recently developed by Isabelle Tellier and colleagues for the morphosyntactic analysis of contemporary French[5] [6]. This tool is based on a machine learning technique called CRF (Conditional Random Fields, [7]) and has obtained the best performance for French. A CRF is able to predict a sequence of labels from a sequence of tokens taking the context into consideration. The notion of context refers here to previous tokens as well as associated labels, making this kind of device more powerful than traditional Markov models.

On a reference corpus of French (the French Treebank [8]), SEM Tagger obtained the best results for part-of-speech (POS) tagging, outperforming all other analyzers for French (compare TreeTagger, a standard tool [9], that got 96.4 F-measure with SEM that obtained 97.7 F-measure) [6]. The improvement may seem modest but is in fact crucial since the quality of POS tagging has a direct influence on the quality of parsing (full syntactic analysis): one word wrongly analyzed may have consequence over the whole sentence. It is a well-known fact that improving POS is crucial and

[5] see http://www.lattice.cnrs.fr/sites/itellier/SEM.html

especially difficult when the baseline is already quite high (hence going from 96.4 to 97.7 is both difficult and crucial).

We will apply this tool to Finno-Ugric languages. One strategy could be to apply the tool directly to word segmented using Morfessor, in order to label data at the morphological level. This part of the work is currently under development.

5.2 Towards Parsing: the Case of Morphology-Rich Languages

The previous section has presented a recent experiment concerning part-of-speech tagging. The next stage is of course to automatically provide a full syntactic analysis, what is called "parsing". Parsing crucially relies on an accurate POS tagging of the corpus to be analyzed.

Most of the developments in parsing have been done on English, for obvious reasons (importance of English as a communication language, existing evaluation campaigns, funding opportunities, etc.). The availability of reference datasets has also played a key role since it is crucial to be able to compare performances across different systems and/or different approaches. However, an approach that is relevant for English may not be as efficient when applied to more diverse languages. The direct transfer of algorithms that are efficient for English to other languages has often led to unsatisfactory results, since language properties differ: at best, a simple adaptation from English leads to representation problems (e.g. when the model adopted for the English PennTreebank has been applied to Arabic, which is a free word order language), at worse it leads to annotation errors since the system makes wrong assumptions.

During the course of the project, we will explore techniques like PCFG-LA, a technique built on top of probabilistic context free grammar (PCFG) [10] LA refers to "Latent Annotations": instead of just considering general linguistic categories (like noun phrases), the system is able to decompose these general categories into homogeneous sets of objects with a similar behavior (e.g. there are different kinds of noun phrases with very different behaviors: distinguishing these different behaviors has a crucial benefit for parsing). This technique is known to better represent the data and thus obtains statistically meaningful improvements over the traditional version of the model (PCFG-LA obtained among the best results on different parsing tasks, for different languages).

Apart the use of PCFG, it has been shown (see [11]) that it is mandatory to take into account language specific features. For example, in the case of Uralic languages, it is crucial to provide a fine-grained morphological analysis, capable of decomposing complex word beginnings and word endings, among other things. English or French are rather analytic, in that most of the relational information between words

is supported by word position and specific relational words, esp. prepositions. This is not the case of most languages (like Hebrew, but also Arabic, Uralic languages or Japanese, to cite a few) and then, in this context, establishing a proper treatment of word morphology is both a complex and crucial task. This is why these languages are of prominent importance since English is highly unrepresentative from this point of view (English having a remarkably low morphologic complexity). Focusing the analysis on morphology complex languages will bring new challenges to the field and guarantee that the developed models are more adapted to language diversity.

A related topic concerns the treatment of unknown words. This is crucial for any parsing system but is even more important in the case of morphology-rich languages since most of the time the context does not give as many cues as in the case of analytic languages for word categorization.

6 Evaluation

Evaluation of natural language processing tools is an open research domain since evaluation must take into account the task, the domain and the context of development. We are nonetheless working on the development of gold standards for the different languages and tasks we are exploring, so that performances can be accurately measured. For most applications (for example part-of-speech tagging or named entity recognition) we think that relevant measures already exist (most of time, precision, recall and F-measure are relevant) and should also be used for Finno-Ugric languages whenever possible.

Using existing measures and open domain evaluation datasets allows one to compare results on a same task and sometimes across domains and/or languages. However, some tasks are clearly more difficult for morphology rich languages than for other languages with a low morphology complexity (as English). To address this issue, it could be interesting to be able to balance evaluation results with morphology complexity.

7 Conclusion

In this paper we have presented different experiments for the automatic analysis of Finno-Ugric languages. We have also given some details about future plans, more specifically through the description of the LAKME project. We are now working on practical experiments so as to get more detailed results soon on some Finno-Ugric languages from Russia (we are for example currently experimenting the automatic

morphological segmentation of Moshka with Morfessor). We are especially open to collaboration since one of the objectives is to provide results for most languages, without duplicating similar work developed elsewhere.

8 Acknowledgements

We want to thank PSL (Paris Sciences et Lettres) for supporting this research through the LAKME project.

References

[1] Daniel Abondolo, editor. *The Uralic Languages*. Routledge, London, 1998.

[2] Max Silberztein. INTEX: a Finite State Transducer toolbox. *Theoretical Computer Science*, 231(1):33–46, 1998.

[3] Thierry Poibeau. The multilingual named entity recognition framework. In *Proc. of the European Conference of the Association for Computational Linguistics (EACL 2003)*, pages 155–158, Budapest, 2003. Association for Computational Linguistics.

[4] Mathias Creutz and Krista Lagus. Unsupervised discovery of morphemes. In *Proceedings of the Workshop on Morphological and Phonological Learning of ACL-02*, pages 21–30, Philadelphia, Pennsylvania, 2002.

[5] Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. *Morfessor 2.0: Python Implementation and Extensions for Morfessor Baseline*. Aalto University publication series Science + Technology, 25/2013, ISBN 978-952-60-5501-5, Helsinki, 2013.

[6] Mathieu Constant and Isabelle tellier. Evaluating the impact of external lexical resources unto a crf-based multiword segmenter and part-of-speech tagger. In *Proc. of the Language Resource and Evaluation Conference (LREC)*, Istambul, 2012.

[7] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[8] Anne Abeillé, Lionel Clément, and François Toussenel. Building a treebank for French. In *Treebanks*, Kluwer, Dordrecht, 2003.

[9] Helmut Schmid. Part-of-speech tagging with neural networks. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, Kyoto, 1994. ICCL.

[10] Yoav Goldberg and Michael Elhadad. Joint hebrew segmentation and parsing using a PCFGLA lattice parser. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, pages 704–709, 2011.

[11] Yoav Goldberg and Michael Elhadad. Word Segmentation, Unknown-word Resolution, and Morphological Agreement in a Hebrew Parsing System. *Computational Linguistics*, 9:121–160, 2013.

Automatic Speech Recognition for Northern Sámi with comparison to other Uralic Languages

Peter Smit[1]
peter.smit@aalto.fi

Juho Leinonen[1]
juho.leinonen@aalto.fi

Kristiina Jokinen[2]
kristiina.jokinen@helsinki.fi

Mikko Kurimo[1]
mikko.kurimo@aalto.fi

[1]Department of Signal Processing and Acoustics, Aalto University, Finland
[2]Institute of Behavioural Sciences, University of Helsinki, Finland

December 30, 2015

Abstract

Speech technology applications for major languages are becoming widely available, but for many other languages there is no commercial interest in developing speech technology. As the lack of technology and applications will threaten the existence of these languages, it is important to study how to create speech recognizers with minimal effort and low resources.

As a test case, we have developed a Large Vocabulary Continuous Speech Recognizer for Northern Sámi, an Finno-Ugric language that has little resources for speech technology available. Using only limited audio data, 2.5 hours, and the Northern Sámi Wikipedia for the language model we achieved 7.6% Letter Error Rate (LER). With a language model based on a higher quality language corpus we achieved 4.2% LER. To put this in perspective we also trained systems in other, better-resourced, Finno-Ugric languages (Finnish and Estonian) with the same amount of data and compared those to state-of-the-art systems in those languages.

1 Introduction

The field of speech recognition is maturing, as companies start to actively use and sell products that utilize Large Vocabulary Continuous Speech Recognition (LVSCR). Especially the creators of operating systems for mobile devices incorporate methods into their products to operate devices using voice.

These commercial applications however, are only focusing on small fraction of the languages in the world. Other languages do not have the required data and expertise readily available, and are therefore left out from these systems as it would not be commercially viable to create these applications. Especially minority languages and languages from developing countries receive only minor academic and commercial interest for the development of LVCSR systems. [1]

One for these under-resourced languages is Northern Sámi, the largest of the nine Sámi languages with approximately 25,000 speakers. It belongs to the Uralic language family. [2]

Like other languages in the Finno-Ugric branch of the Uralic language family, e.g. Finnish and Estonian, it is a highly morphological language that uses independent suffixes extensively. This poses challenges for speech technology applications as the number of inflections, derivations and compoundings cause the size of the vocabulary to be enormous, especially compared to the languages in the Indo-European family [3]. A large vocabulary especially causes problems in the estimation of language models, which can not produce any words beyond those seen in the the training data.

Northern Sámi is an under-resourced language, as there are little corpora of spoken and written language available, and financial resources to collect these data are limited. Even though there is active linguistic research on Northern Sámi, there are limitations to the expert resources available for speech recognition, such as pronunciation dictionaries.

To combat the challenges of building an LVCSR system for an under-resourced language we have employed several techniques. First we used 'found data' for building the acoustic and language models. For the acoustic model we bootstrapped from a better resourced related language (Finnish). For the language model we increased the coverage of the model by employing sub-word units (morphs) instead of words. Similar techniques have been used in [1, 4] but here we wanted to evaluate their applicability to uralic languages, in particular. This work is an extension of [5].

Because there are no state-of-the-art LVCSR references for Northern Sámi, we simulated the potential of larger resources by studying also two better resourced Uralic languages. First we produced systems for Finnish and Estonian using the corresponding data as we had for Northern Sámi. Then we compared these systems to similar systems that we produced using larger data and, finally, to the state-of-the art systems for these languages. These results helped us to estimate the gains for collecting more Northern Sámi data.

1.1 WikiTalk and DigiSami

Another motivation to build a recognizer for Northern Sámi is to utilize it as part of a spoken dialogue system in the WikiTalk application [6]. This is one part of the DigiSami project which is a research project at University of Helsinki aiming to support content generation of less resourced languages with the help of language technology. Currently, the main dangers to Sámi language are the disappearance of the traditional lifestyle and work of Sámi culture, and emigration of Sámi people away from their old living areas. However, there are also studies and discussion on using new technologies to revitalize languages [7]. In [8], revitalization for the Northern Sámi language is described using spoken language data collection in interactive setting for the WikiTalk application. In WikiTalk, the idea is to have users (children or adults) find out more about subjects that interest them by discussing with the humanoid robot Nao. They can ask for more information on the subject and then Nao will read them the related Wikipedia article [9]. Described in this paper is the first step to building this end-to-end system.

2 ASR for under-resourced languages

The majority of the state-of-the art methods in Large Vocabulary Speech Recognition require large amounts of data and expertise.

Firstly, a great number of high quality spoken utterances have to be collected and correctly transcribed. For a Speaker-Independent (SI) system, i.e. a system that can recognize anyone who speaks the target language, utterances from many different persons are needed. For a Speaker-Dependent

(SD) system, i.e. a system that can only recognize the voice of the person who provided the training data, only a few hours of transcribed speech are required.

The second required dataset is a large corpus of written text, preferably in the same style and domain as what should be recognized by the system. Te corpus is used to train the language model and it should contain all common words in their expected contexts.

Lastly, a speech recognizer needs a pronunciation dictionary; i.e. a list of all possible words with all their possible phonetic transcriptions. The phonemes also need to be grouped according to different phonetic properties, so that their probability distributions can be shared in the training of the acoustic model.

For under-resourced languages, as the name suggests, none of the above data is readily available, but alternative solutions have to be developed. An easy alternative to a large corpus of transcribed audio data is to collect audio books. Although the quality of the speech varies, projects such as Librivox have freely available audio books in many languages which can be used for this purpose. Using a temporary acoustic model and simple text processing techniques these audio books can be automatically segmented into sentence-long utterances that are suitable for training a minimal speaker-dependent model.

Language data is also freely available on-line, and e.g. Web-scraping can give a rudimentary dataset for training a language model [10]. Also sites like Wikipedia have often big collections of easily available text. However, the quality and usability of such data varies, and many of the sources that can be 'found' on-line suffer from the problem that their style and topic are non-standard and do not necessarily match written nor spoken language conventions. Moreover, on-line texts often contain foreign language segments, symbols or abbreviations which decrease their usability for building language models.

One of the main resource consuming tasks is the preparation of a pronunciation dictionary, which normally requires extensive manual work and linguistic knowledge. One solution to build the pronunciation dictionary quickly is to model the graphemes (letters) of the words directly, instead of using the actual phone they represent [11]. In languages such as English this does not, of course, give very good results since graphemes can have very different realizations. Consider for example the words 'tough' and 'dough' that resemble each other in writing, but are pronounced in a completely different way. In the Uralic languages studied here however, a grapheme-to-sound system works reasonably well since, in general, every grapheme is realized as a single distinct sound.

Lastly, the phonetic grouping or 'phoneme question set' is a small dataset that requires linquistic expertise. Although there are algorithms available that can replace this set altogether [12], it is often undesirable as it makes the system less effective. It is also possible to modify the phoneme set of a closely related language, and such small modifications to approximate the target language do not necessarily require so much expert effort.

Even though the above simplified solutions can replace all the expensive data needs, they will inevitably limit the performance of the speech recognizer. Adding more and domain related training data as well as developer expertise will naturally improve the system performance significantly. However, the low-resource systems can already serve some basic language technology needs. The largest limiting factor for these systems is that a real SI system requires training data from more than a hundred speakers.

3 Acoustic modeling

The Acoustic Modeling part of the speech recognizer was done with a standard Hidden Markov model with Gaussian mixture models as emission distribution (HMM-GMM). Mel Frequency Cepstral Coefficients were used as input features. [13]

The audio data is prepared by splitting the audio files (originally chapter length or similar) into sentence utterances. This is done by doing Baum-Welch forced alignment with a temporary speech recognition model. The temporary speech recognition model was created by taking a well trained Finnish model and mapping the Finnish phonemes to the one of the target language. In later iterations the best speech recognition model of the language was used to do the forced alignment again, resulting in a perfect split of training utterances.

The HMM-GMM model is trained using multiple iterations of Baum-Welch maximum likelihood estimation. To manage the model complexity Gaussians were shared between different HMM-states using decision tree clustering. The modeling unit of the acoustic model is a tri-state tri-phone, which means that all the phonemes with a different preceding and succeeding phoneme are modeled as separate units, as are the beginning, middle and end of each tri-phone.

In Section 7 the number of Gaussians for different models are reported.

4 Language modeling

A language model is an important part of any speech recognition system. Even though theoretically a good acoustic model with a lexicon could be enough to recognize words, a model that takes the word context is essential. For languages which have many homophones, i.e. words with the same pronunciation but different meaning, it is also essential to have a language model, so as to pick the right word meaning given a pronunciation in the context.

A language model predicts words based on their sentence context. For synthetic languages like Finnish and all the Uralic languages, the main issue with word-based language modeling is that a huge lexicon is needed in order to decrease the out-of-vocabulary (OOV) rate to a manageable percentage. Since the OOV-rate is the minimum WER possible, an OOV-rate much less than 10% is necessary. For an English speech recognizer, a vocabulary size of 20 000 word may provide an OOV-rate of 2.4-2.7%, while with a vocabulary of 40 000 words, an OOV-rate less than one percent is achieved [14]. In contrast, a Finnish recognizer needs a 410 000 word vocabulary to have an OOV-rate of 4.0-7.3% [15].

An interesting alternative for a word-based language model is to use a sub-word language model. A sub-word model builds words out of a smaller set of word fragments. The word fragments are particularly effective in agglutinative languages or languages with a lot of compound words. When the words are built from smaller units, also the OOV words can be modelled by using the probabilities of sub-word unit combinations learned from the training corpus. If the word fragments are chosen appropriately, the OOV-rate can become close to zero, even for smaller language data corpora.

4.1 Morfessor

Morfessor is a machine learning tool that uses a statistical model to split words into smaller fragments, which can be used for language modeling [16]. This resembles closely the splitting of words into their smallest informational units, morphemes.

Morfessor has three components; the model, the cost function, and the training and decoding algorithms. The model contains the lexicon, i.e. the properties of the morphs, the written form of the morph itself and its frequency, as well as the grammar, which contains information of how the morphs can be combined into words. The Morfessor cost function is derived from a MAP estimation with the goal of finding the optimal parameters $\boldsymbol{\theta}$ given the observed training data $\boldsymbol{D}_W$:

$$\boldsymbol{\theta}_{MAP} = \arg\max_{\theta} P(\boldsymbol{\theta}|\boldsymbol{D}_W) = \arg\max_{\theta} P(\boldsymbol{\theta})P(\boldsymbol{D}_W|\boldsymbol{\theta}). \quad (1)$$

The cost function to be minimized is the negative logarithm of the product $P(\boldsymbol{\theta})P(\boldsymbol{D}_W|\boldsymbol{\theta})$

$$L(\boldsymbol{\theta}, \boldsymbol{D}_W) = -\log P(\boldsymbol{\theta}) - \log P(\boldsymbol{D}_W|\boldsymbol{\theta}). \quad (2)$$

The purpose of this is to generate a small set of morphs that represents the words in the training corpus compactly. If only letters were used as morphs the set of would be small but representing the corpus with individual letters would be cumbersome. In contrast using whole words as morphs would result in a large set of morphs so the optimal solution is somewhere in between. However, individual letters are added to the morph set so even previously unseen words can always be segmented.

A greedy search algorithm is used to find the optimal segmentation of morphs for the training data. When the best model is found, it is used to segment the language model training corpus with the Viterbi algorithm. This result can be used to generate n-gram models with morphs as LM units.

4.2 n-gram modeling

n-gram models predict the output of the next word or sub-word given the $n-1$ previous words or sub-words. They are normally created by counting all occurrences of the word and sub-word sequences. To prevent the model from being too big and too much tailored to the training data (overfitting), pruning is applied. Also, some of the probability mass is reserved for unseen contexts, for example with the Kneser-Ney smoothing technique[17].

When n-gram models are build for words, the order the model, i.e the value of n, is typically between three and five. If the order is high, the models get too big, and they do not contain enough necessary information. With the sub-word models, however, the contexts can be much deeper, as there are less types in the vocabulary and the context counts are more sparse. Also intuitively, to cover the same context, the order of a sub-word model must be higher than the order or the word model. Standard tools for n-gram modelling have problems with correctly smoothing and growing high-order n-gram language models. VariKN [18] is a specific algorithm and tool to solve this problem and it was used in this paper for building high-order sub-word n-gram models.

5 Experiment setup

The experiments were carried out using our open source speech recognition toolkit called AaltoASR[1] [13][19]. It uses context-dependent tri-phones with diagonal Gaussian mixture models (GMM) as emission distributions and the speech features itself are Mel-Frequency Cepstral Coefficients (MFCCs).

[1]Open source, available from `https://github.com/aalto-speech/AaltoASR`

Both words and sub-word units were used for language modeling. The sub-word unit models were created with Morfessor 2.0[2], an implementation of the Morfessor Baseline algorithm[20].

Variable length n-grams used for language modeling were generated by both SRILM[3] [21] and VariKN[4] [18, 22]. The decoder of AaltoASR is a time-synchronous one-pass token passing decoder where the beam search is complemented by a language model look-ahead [23].

6 Northern Sámi ASR evaluation

The audio data used for the Northern Sámi recognizer came from the UIT-SME-TTS corpus[5]. There are data for two speakers, one male and one female. The male audio data was 4.7 hours and the female data 3.3 hours. Separate data is needed for development and evaluation, and we used 75% for training. This makes 3.5 and 2.5 hours of training data for the male and female voice, respectively.

The initial recognition model was created by using a Finnish model. With this model, the audio data was split into sentences and trained with the procedure described in Section 3. This resulted in two speaker dependent systems, one for the male and one for the female speaker (resp. SM1 and SF1). These models are Speaker-Dependent models as there is data only from the two speakers available.

For language model, we evaluated both word and morph n-gram models. In addition to the training sentences, we also used the Northern Sámi Wikipedia dump (Train+Wiki).

The results for basic recognition are shown in Table 1. Besides the standard Word Error Rate (WER), also the Letter Error Rate (LER) is reported. LER is common for speech recognition experiments on languages which are morphological complex such as Northern Sámi, Finnish and Estonian.

		Speaker SF1			Speaker SM1		
Unit	**Toolkit**	**5-gram**	**7-gram**	**9-gram**	**5-gram**	**7-gram**	**9-gram**
words	SRILM	52.9 / 12.7	52.9 / 12.7	52.9 / 12.7	48.6 / 11.1	48.7 / 11.1	48.7 / 11.1
morphs	SRILM	40.0 / 9.0	39.9 / 9.3	39.1 / 9.1	37.6 / 8.5	36.8 / 8.4	37.3 / 8.5
morphs	VariKN	38.4 / 8.6	38.5 / 8.7	37.6 / 8.7	35.4 / 8.1	33.7 / 7.6	34.1 / 7.9

Table 1: ASR recognition results for the Northern Sámi SD recognizers. Word Error Rate / Letter Error Rate reported.

We first observe that the SM1 recognizer is slightly better than the SF1 recognizer. However, this is most likely caused by the fact that there was more data available for the training of the acoustic model.

As expected, the morph based language models have much lower error rates than the word-based models. Looking at Table 2, we notice that the OOV-rate for word based models is rather high which causes the big difference in performance to sub-word models.

For word-based models there is no effect on using higher order n-grams. This can be seen in Figure 1 which shows WER for different n-gram models with the SM1 system. In this comparsion we used the Big Northern Sámi language model which is trained from approximately 12 million word tokens of data from 'Den samiske textbanken'. There is no change in performance after the 3rd order n-grams

[2]Open source, available from `http://www.cis.hut.fi/projects/morpho/`

[3]Open source, available from `http://www.speech.sri.com/projects/srilm/`

[4]Open source, available from `https://github.com/vsiivola/variKN`

[5]Provided by the University of Tromsø

	Word	Morph
Female	22%	0%
Male	20%	0%

Table 2: Out-of-vocubulary percentages for the Female and Male testsets.

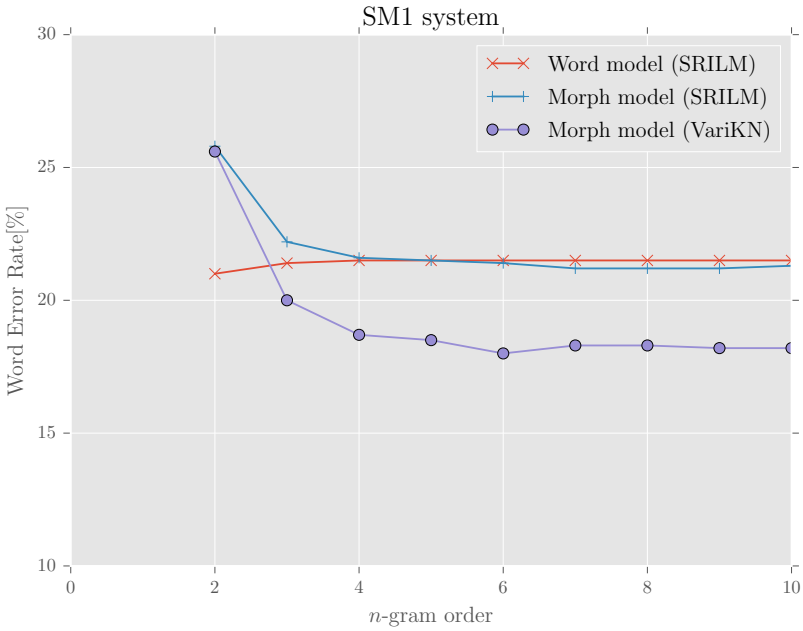

Figure 1: Word error rates for the SM1 system with the Big language model.

for the the word-based model, whereas for the VariKN morph-based models there are clear effects when using higher order models.

The best Word Error Rate on the Big language model for the SM1 system is 18.2%, the best Letter Error Rate 4.2%.

7 Comparison of low-resource systems for multiple languages

To compare the results of the Northern Sámi recognizer with recognizers in different languages we first train Speaker Dependent models for both Finnish and Estonian audio books. The available audio datasets are described in Table 3 and the available text corpora in Table 4.

Even though all datasets are audio books, there are a number of differences. EF1, EM1 and FM1 were encoded with the mp3-codec, while for the FF1, SM1 and SF1 audio books original high quality uncompressed audio files were available. The speaking style was generally the same, with a prosody typical to story telling. An exception to this was the FF1 book, an audio book created for blind persons, which has been read in a very monotone voice with little prosodic variation. This makes the book also understandable when played at higher speeds.

[6]Provided by YLE. Can be listened on `http://areena.yle.fi/1-1301621`

	Language	Gender	Title	Amount
EF1	Estonian	Female	Nils Holgerssoni imeline teekond läbi Rootsi	16 hours
EM1	Estonian	Male	Würst Gabriel ehk Pirita kloostri wiimsed päewad	6 hours
FF1	Finnish	Female	Syntymättömien sukupolvien Eurooppa	12 hours
FM1	Finnish	Male	Seitsemän veljestä[6]	13 hours
SF1	Northern Sámi	Female	UIT-SME-TTSF	3.3 hours
SM1	Northern Sámi	Male	UIT-SME-TTSM	4.6 hours

Table 3: Audio data for the trained speaker dependent systems.

Language	Source	#sentences	#word tokens	#word types
Estonian	Wikipedia	895k	10M	778k
Estonian	newspaper+web+broadcast [24]	19M	229M	3.8M
Finnish	Wikipedia	2.2M	22M	1.5M
Finnish	Kielipankki	13M	143M	4.1M
Northern Sámi	Wikipedia	10k	88k	20k
Northern Sámi	Den samiske tekstbanken	990k	12M	475k

Table 4: Language modeling data for the trained speaker dependent systems.

The experiments in Section 6 confirmed the hypothesis that morph-based n-gram models trained with the VariKN toolkit give the best performance, hence only this combination will be used.

To compare the systems for different languages fairly, we artificially reduce the amount of audio and text data to match that of our smallest system. We only use 2.5 hours of audio data and a random 10.000 sentences of the Wikipedia data set for each language. The systems are trained with a 10-gram VariKN sub-word language model. The statistics in Table 5 show that the datasets have equal number of sentences, but not equal number of word types or tokens. This is most likely due to the Northern Sámi Wikipedia having more stub articles that contain short sentences with similar words.

The Train+Wiki language models are trained from the combination of the recognizer's training sentences and the small Wikipedia dataset as described in Table 5. The Big language models are trained from the higher quality text sources, which are described in Table 4.

The results of the comparable systems with the Train+Wiki dataset are shown in Table 6. The word error rates are close to each other, confirming that the systems are comparable. One exception is the FF1 system, which performs much better. This better result is most likely a combination of the speaking style, which had little variation, and a better match between the text of the language model and the test data.

We also tested the models with the same amount of acoustic data and their respective Big language models. The improvements are significant with the best improvement being 64% relative improvement in WER for the FF1 system. This indicates the importance of the availability of high quality language model data for the performance of a Uralic speech recognition system. The amount of data

Language	#sentences	#word tokens	#word types
Estonian	10k	108k	41k
Finnish	10k	103k	43k
Northern Sámi	10k	88k	20k

Table 5: Reduced subsets of wikipedia data for use in the Train+Wiki language model.

		Train+Wiki		Big	
Language	**Voice**	**WER**	**LER**	**WER**	**LER**
Estonian	EF1	39.6	15.8	25.0	11.4
Estonian	EM1	39.2	13.3	25.5	9.6
Finnish	FF1	25.2	4.1	8.9	2.1
Finnish	FM1	35.8	7.7	24.9	5.6
Northern Sámi	SF1	37.5	8.5	23.7	5.5
Northern Sámi	SM1	39.5	9.4	20.9	4.9

Table 6: Word Error Rates for using 2.5 hours of training data and either the Train+Wiki or Big language models. All language models were 10-gram VariKN sub-word models.

				2.5 hours		All data	
Language	**Voice**	**#hours**	**#Gaussians**	**WER**	**LER**	**WER**	**LER**
Estonian	EF1	8	31.5k	25.0	11.4	18.8	8.3
Estonian	EM1	4.5	12.6k	25.5	9.6	23.2	8.4
Finnish	FF1	9	26k	8.9	2.1	8.1	1.9
Finnish	FM1	10	28k	24.9	5.6	19.8	3.7
Northern Sámi	SF1	2.5	7.7k	23.7	5.5	23.7	5.5
Northern Sámi	SM1	3.5	9.6k	20.9	4.9	18.1	4.2

Table 7: Speech recognizer results for the full audio books with the Big language model.

however is less important, as the Big language model for Northern Sámi gives a similar improvement as the Big language models for the other systems, even though the amount of data in the Big language model for Northern Sámi is lower than the amount of data in the Train+Wiki systems for Estonian and Finnish.

To see the effect of using more acoustic data, we also trained all systems on their full acoustic datasets and evaluated them with the Big language model. While the 2.5 hour data systems were all modeled with appr. 7,500 Gaussians, the bigger models have proportionally more Gaussians.

The results are shown in Table 7. There are a couple of surprising results. For the FF1 system, there is a small improvement on the already very good result. On the other hand, the SM1 system already improves with 13% relative WER with only an hour of added data. In general, there is a clear pattern that more acoustic data improves the model, except if the data has so little variation that an optimum is already reached earlier.

7.1 State-of-the-art recognizers

The experiments in the previous sections show that results on Finnish and Estonian systems are comparable with Northern Sámi systems if the same amount of data is provided. This allows us to look to the state-of-the-art recognition systems for Finnish and Estonian systems and project how well a Northern Sámi system would perform if the same amount of data would be collected.

Table 8 shows the reported error rates for different systems. The most important difference with the systems discussed in the previous sections is that these are Speaker Independent recognizers, which are tested with different speakers than those present in the training data. Also the quality and type of speech are different.

Of these state-of-the-art results, the results on the Finnish Speecon set and the Finnish telephone

Language	Description	WER	LER	Source
Estonian	Broadcast conversations	17.9%		[25]
Estonian	Oral presentations	26.3%		[25]
Finnish	Speecon testset		2.9%	[26]
Estonian	Telephone speech	33.1%	11.9%	[13]
Finnish	Telephone speech	21.6%	6.8%	[13]

Table 8: State of the art results for Finnish and Estonian Speaker Independent ASR.

speech are most impressive. Even though there is much more speaker variability, the result on the Speecon testset is close to the result of the FF1 SD recognizer. This is done using speaker adaptive training and discriminative training techniques.

The telephone speech results are focused on lower quality speech data. Again the results seem better for the SD systems in the previous section, but the variability in speakers, the speech quality and the language content of the utterances are much more complex.

Given that the Speaker Dependent systems all performed with similar accuracy, we expect that tasks of similar difficulty would perform as well for Northern Sámi as they would for Finnish or Estonian, given that the data would be available.

8 Conclusions

Using a number of techniques, most notably sub-word language models and grapheme-to-sound acoustic modeling, we have overcome challenges caused by a small amount of data available for developing speech recognizer systems for under-resourced languages. We have demonstrated the feasibility of this approach by training Speaker Dependent speech recognizer systems for the Northern Sámi language, an under-resourced Finno-Ugric language, and achieved a letter error rate of only 4.2%.

In order to put the result in perspective and validate the techniques, we also trained systems for Finnish and Estonian using artificially limited datasets. These experiments show that the Northern Sámi recognizer gives comparable results to the Finnish and Estonian recognizers and can effectively use similar techniques such as sub-word language models.

In future work we plan to use cross-lingual techniques to build Speaker Independent systems for Northern Sámi, even though acoustic datasets with enough different speakers might not be available, or only available without transcriptions.

All scripts used in this paper are published as open-source under the Modified BSD license[7].

9 Acknowledgements

We thank the University of Tromsø for the access to their Northern Sámi datasets. This research has been supported by the Academy of Finland under the Finnish Centre of Excellence Program 2012–2017 (grant n°251170, COIN) , and through the project Fenno-Ugric Digital Citizens (grant n°270082). We acknowledge the computational resources provided by the Aalto Science-IT project.

[7]Available from `https://github.com/phsmit/iwclul2016-scripts`

References

[1] Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz. Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100, 2014.

[2] M. Paul Lewis, Gary F. Simons, and Charles D. (eds.) Fennig. Ethnologue: Languages of the world, eighteenth edition. Online version: http://www.ethnologue.com., 2015.

[3] Fred Karlsson. *Suomen kielen äänne- ja muotorakenne*. WSOY, Helsinki, 1982.

[4] Viet-Bac Le and L. Besacier. Automatic speech recognition for under-resourced languages: Application to Vietnamese language. *Audio, Speech, and Language Processing, IEEE Transactions on*, 17(8):1471–1482, Nov 2009.

[5] Juho Leinonen. Automatic speech recognition for human-robot interaction using an under-resourced language. Master's thesis, Aalto University School of Electrical Engineering, Espoo, 2015.

[6] Graham Wilcock and Kristiina Jokinen. Wikitalk human-robot interactions. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction (ICMI)*, pages 73–74. ACM, 2013.

[7] Patrick Eisenlohr. Language revitalization and new technologies: Cultures of electronic mediation and the refiguring of communities. *Annual Review of Anthropology*, pages 21–45, 2004.

[8] Kristiina Jokinen. Open-domain interaction and online content in the Sami language. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2014)*, 2014.

[9] Kristiina Jokinen and Graham Wilcock. Multimodal open-domain conversations with the Nao robot. In *Natural Interaction with Robots, Knowbots and Smartphones*, pages 213–224. Springer, 2014.

[10] Kevin P Scannell. The crúbadán project: Corpus building for under-resourced languages. In *Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop*, volume 4, pages 5–15, 2007.

[11] Stephan Kanthak and Hermann Ney. Multilingual acoustic modeling using graphemes. In *INTERSPEECH*, pages 1145–1148, 2003.

[12] Sakhia Darjaa, Milos Cernak, Marián Trnka, Milan Rusko, and Róbert Sabo. Effective triphone mapping for acoustic modeling in speech recognition. In *INTERSPEECH*, pages 1717–1720, 2011.

[13] Teemu Hirsimäki, Janne Pylkkönen, and Mikko Kurimo. Importance of high-order n-gram models in morph-based speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 17(4):724–732, 2009.

[14] Phil C Woodland, CJ Leggetter, JJ Odell, V Valtchev, and SJ Young. The 1994 HTK large vocabulary speech recognition system. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 73–76. IEEE, 1995.

[15] Teemu Hirsimäki, Mathias Creutz, Vesa Siivola, Mikko Kurimo, Sami Virpioja, and Janne Pylkkönen. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech & Language*, 20(4):515–541, 2006.

[16] Mathias Creutz and Krista Lagus. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(1):3, 2007.

[17] Stanley F Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318. Association for Computational Linguistics, 1996.

[18] Vesa Siivola, Teemu Hirsimaki, and Sami Virpioja. On growing and pruning Kneser–Ney smoothed-gram models. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(5):1617–1624, 2007.

[19] Janne Pylkkönen. An efficient one-pass decoder for Finnish large vocabulary continuous speech recognition. In *Proceedings of The 2nd Baltic Conference on Human Language Technologies*, pages 167–172, 2005.

[20] Sami Virpioja, Peter Smit, Stig-Arne Grönroos, Mikko Kurimo, et al. Morfessor 2.0: Python implementation and extensions for Morfessor baseline. Technical report, 2013.

[21] Andreas Stolcke et al. Srilm-an extensible language modeling toolkit. In *INTERSPEECH*, 2002.

[22] Vesa Siivola, Mathias Creutz, and Mikko Kurimo. Morfessor and variKN machine learning tools for speech and language technology. In *INTERSPEECH*, pages 1549–1552, 2007.

[23] Stefan Ortmanns, Andreas Eiden, Hermann Ney, and Norbert Coenen. Look-ahead techniques for fast beam search. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 3, pages 1783–1786. IEEE, 1997.

[24] Mikko Kurimo, Seppo Enarvi, Ottokar Tilk, Matti Varjokallio, André Mansikkaniemi, and Tanel Alumäe. Modeling under-resourced languages for speech recognition. *Language Resources and Evaluation*, in review.

[25] Tanel Alumäe. Recent improvements in Estonian LVCSR. In *Spoken Language Technologies for Under-Resourced Languages*, 2014.

[26] Janne Pylkkonen and Mikko Kurimo. Analysis of extended Baum-Welch and constrained optimization for discriminative training of hmms. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(9):2409–2419, Nov 2012.

Intermediate representation in rule-based machine translation for the Uralic languages

Francis M. Tyers,
HSL-fakultehta
UiT Norgga árktalaš universitehta
francis.tyers@uit.no

Tommi A. Pirinen
ADAPT Centre
School of Computing,
Dublin City University
tommi.pirinen@computing.dcu.ie

Abstract

This paper presents some of the major obstacles and challenges in creating machine translation systems between Uralic languages where the intermediate representation is based on morphology and syntax. The Uralic languages are very alike in many ways: similar case inventories, word order and non-finite clause forms. However current rule-based grammatical resources take many different approaches to encoding this information. These approaches are sometimes based on legacy or traditional grammatical description, important for making the tools comfortable for linguists, but sometimes based on arbitrary and incompatible decisions. This paper presents an overview of some of the issues in working with existing tools and representations and provides some guidelines and suggestions to facilitate future work.

1 Introduction

Creating *rule-based machine translation* (RBMT) systems is a process where one creates a mapping between units of source language and target language. The units can be different depending on the approach to the problem,

i.e., on scale of translating word-forms to word-forms to translating via an intermediate abstract universal language, or an *interlingua*. In this article we study the approach of using just morphological analysis with the Uralic languages. The problem of such a system is that, even when morphologies of the closely related Uralic languages are expected to match, there are often engineering issues that make the work more tedious and cumbersome than necessary. Minimising the amount of simple engineering work is vital for making rule-based machine attractive to linguists and programmers alike.

The rest of the article is structured as follows: first we describe the backgrounds of the problem in 2, then we introduce the resources we are going to use in 3, we suggest some common best practices in 6, in 7 we briefly describe universal parts-of-speech and morphological features, and finally in 8 we provide some short concluding remarks.

2 Background

RBMT is a popular way of developing high-quality machine translations between related languages [1]. The building of an RBMT system rapidly for related languages is possible, as has been done with, e.g. Dutch and Afrikaans [2]. A wide-coverage machine translation requires wide-coverage lexical resources for the languages. Developing an analyser to a stage where it is usable by multiple applications, including RBMT, can take years, so it is often a good idea to use readily available resources instead of rewriting a new analyser from the scratch. However, the majority of existing analysers are made with language-dependent annotation systems, which unnecessarily complicate the description of machine translation. It should be clear, that if two related languages use the same morphological and syntactic structures to describe a phenomenon, a rule mapping between the two should be entirely trivial. This is not the case when taking most off-the-shelf analysers for contemporary Uralic morphologies. Table 1 shows an example of the morphological annotation of five Uralic languages for a simple five-word sentence.

James	*ja*	*Mary*	
+N+Prop+Sem/Mal+Sg+Nom	+CC	+N+Prop+Sem/Fem+Sg+Nom	
leaba	*gárdimis*	.	
+V+IV+Ind+Prs+Du3	+N+Sg+Loc	+CLB	
Джеймс		*марто*	*Марит*
+N+Prop+Sem/Mal+Sg+Nom+Indef		марто+Po+COM	+N+Prop+Sem/Fem+Pl+Nom+Indef
садпиресэтъ		.	
+N+SP+Ine+Indef+Der/Pr+V+Ind+Prs+ScPl3		+CLB	
James	*ja*	*Mary*	*ovat*
N Prop Nom Sg	Part	N Prop Nom Sg	V Prs Act Pl3
puutarhassa	.		
N Ine Sg	Punct		
James	*ja*	*Mary*	*on*
+H+sg+nom	+J	+H+sg+nom	+V+indic+pres+ps3+pl+ps+af
aias	.		
+S+sg+in	.		
James	*és*	*Mary*	a
/NOUN	/CONJ	/NOUN	/ART
kértben	*vannak*	.	
/ADJ<CAS<INE»	/VERB<PLUR>	/PUNCT	

Table 1: Translations of the sentence 'James and Mary are in the garden.' in several Uralic languages (North Sámi, Erzya, Finnish, Estonian, Hungarian) with the tag strings used in their morphological analysers. There are examples of real morphosyntactic differences (compare the third-person dual in North Sámi with the third-person plural in other languages) and arbitrary tag differences (compare the tag that the word for *and* receives in the different languages).

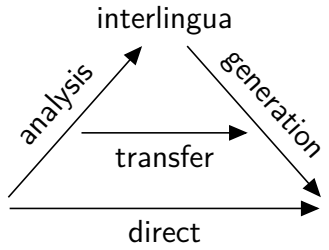

Figure 1: The Vauquois triangle which illustrates the amount of transfer needed for different levels of intermediate representation.

2.1 Intermediate representations

In machine translation, an intermediate representation is an abstraction away from the surface forms of the language. Figure 1 shows the Vauquois triangle, a common illustration of different levels of intermediate representation.

At the bottom of the triangle, there is no intermediate representation and translation is performed on a word-for-word basis. At the apex of the triangle is interlingual translation, where the source language is first mapped to a language-independent semantic representation, and this representation is then used to generate the target language.

In the middle is (morpho-)syntactic transfer. Here the source language is analysed to a language-dependent intermediate representation (usually based on a combination of syntactic structure and morphosyntactic features) and then transfer rules are applied to convert the source language intermediate representation to one compatible with the target-language generation component.

3 Resources

In this paper we make use of five sets of linguistic data for five different Uralic languages: Finnish, North Sámi, Erzya, Estonian and Hungarian. We take the North Sámi and Erzya data from the Giellatekno language technology repository.[1] The North Sámi data has primarily been developed by the Divvun and Giellatekno groups at UiT Norgga árktalaš universitehta and the Erzya data has been developed by Jack Rueter at Helsingin yliopisto [3]. For the Estonian data, we use the *plamk* analyser[2] written by Jaak Pruulmann-Vengerfeldt, for Finnish, *omorfi* [4][3] and for Hungarian, hunmorph [5].[4]

[1] http://giellatekno.uit.no

[2] https://github.com/jjpp/plamk

[3] https://github.com/flammie/omorfi

[4] http://mokk.bme.hu/resources/hunmorph/

4 Strategies

There a different ways to fix systematic mismatches. We evaluate the followings:

4.1 Relabelling

An obvious approach to getting around the problem of divergent tagsets is to simply perform relabelling. This is where you replace the canonical tags in one language with their equivalents in the other language, or with a common equivalent in both languages.

$$+CC \rightarrow <cnjcoo> \leftarrow +J+Coord$$

However, this solution has its disadvantages. Even though +J and +CC both are used for conjuctions, the *plamk* tag is also used with subordinating and other conjunctions, while the Giellatekno tag excludes those. Relabelling +J+Coord to +CC and any other +J to +CS might work on the analyser, but will not work in a disambiguation rule saying "select the noun reading if the word to the right is tagged +J", here we need to relabel +J to (+CS or +CC). In the opposite direction, +CS would need to be relabelled to (+J but not +Coord). The distinction between these may be irrelevant for the translation process (in all cases, *ja* in North Sámi will be translated to *ja* in Estonian), but for the intervening grammatical tools, it may be vital to make (or not) the distinction.

4.2 Interlingua

Another potential solution is to use a semantic interlingua (see description in section 2.1). This is the approach adopted by the machine translation system based on Grammatical Framework [6].[5] In this framework there is no direct transfer of morphological features.

[5] http://grammaticalframework.org

5 Specific linguistic issues

There are a number of linguistic issues in RBMT. We cover the following in detail:

5.1 Copula

There are two main copula constructions in the Uralic languages, the first functions more or less like in the Germanic languages. The copula is a normal verb that agrees with the subject. The second copula construction works like in the Turkic languages. In languages with the Turkic-style copula, it does not typically surface in the third-person singular present tense. In our examples, North Sámi, Finnish and Estonian are of the Germanic type, while Hungarian and Erzya are of the Turkic type.

	'She is a student.'	'She was a student.'
North Sámi	Son lea studeanta.	Son lei studeanta.
Erzya	Сон студент.	Сон студентель.
Finnish	Hän on opiskelija.	Hän oli opiskelija.
Estonian	Ta on üliõpilane.	Ta oli üliõpilane.
Hungarian	Ő hallgató.	Ő hallgató volt.

In North Sámi, Finnish and Estonian, the treatment of *lea, on* is similar. It is a verb which inflects and agrees like other verbs.

There are divergences when we look at the Erzya and Hungarian examples. Although they have the same structure, zero copula in the present tense and surfaced copula in the past tense. The morphological analyser for Erzya treats the copula as a derivation:

студент+N+Sg+Nom+Indef+Der/Pr+V+Ind+Prs+ScSg3

Where in Hungarian it is simply omitted in the present (if it surfaced it would be *van*), and in the past it is considered a verb form.

5.2 Non-finite verb forms

Non-finite verb forms are infinitives and participles on the on hand and derivations on the another. There are a different number of them between languages and their tasks vary from being syntactic arguments of constructions to derived words, and a wide range of analyses are used to accommodate that. There are some differences in the table 2

Language	Sentence	Non-finite tag
	'I see the man who is running'	
North Sámi	Oidnen dievddu viehkame	Actio+Ess
Erzya	Неян цёрань, конась чийни.	Der/ЫI+ActPrcShort+A
Finnish	Näen miehen juoksemassa.	InfMA+Ine
Estonian	Näen meest, kes jookseb.	—
Hungarian	Látom a futó embert.	/VERB[IMPERF_PART]/ADJ
	'While running I saw the man'	
North Sámi	Oidnen dievddu viegadettiinan.	Ger+Px1Sg
Erzya	Неян чийниця цёрань.	Der/ЫIця+ActDemPrc+A
Finnish	Näin miehen juostessani.	InfE+Ine+PxSg1
Estonian	Jooksmise ajal nägin ma meest.	Der/mine+Gen
Hungarian	Futás közben láttam az embert.	/VERB[GERUND]/NOUN
	'I see the running man.'	
North Sámi	Oainnán viehkki dievddu.	PrsPrc
Erzya	Чийнемась седень кецявты.	Der/ОмА+Nom
Finnish	Näen juoksevan miehen.	PrsPrc
Estonian	Näen jooksvat meest.	Der/v+A+Nom
Hungarian	Látom a futó embert.	/VERB[IMPERF_PART]/ADJ
	'Running is fun.'	
North Sámi	Viehkan lea suohtas.	Actio+Nom
Erzya	Мелезэнь тукшны чийнемась.	Der/ОмА+Nom
Finnish	Juokseminen on kivaa.	Der/minen+Nom
Estonian	Jooksmine on lahe.	Der/mine+Nom
Hungarian	A futás jó dolog.	/VERB[GERUND]/NOUN
	'I like running.'	
North Sámi	Liikon viehkat.	Inf
Erzya	Чийнемстэ неия цёрань.	Inf+Ela
Finnish	Pidän juoksemisesta.	Der/minen+Ela
Estonian	Mulle meeldib joosta.	Inf
Hungarian	Szeretem futni.	/VERB<INF>

Table 2: Examples of the use and tagging of non-finite verb forms in the languages in our sample. It is not to be expected that the tags are completely equivalent, but for example, given the similarity in structure, should there be a difference in annotation between Finnish PrsPrc and Estonian Der/v+A?

5.3 Derivation, compounding and lexicalisation

A classical problem in computational morphologies lies in question of lexicalisation and productivity of certain processes; is a morphologically created word-form a new word or a form of a, possibly distant root. Morphologies take widely different and opposing approaches to this ranging from lexicalise-everything to collect-everything. See examples below:

	'to drink'	'a drink'	'drinker'	'brewery'
North Sámi	juhkat	juhkamuš	—	vuolla·buvttadeaddji
Erzya	симемс	симема-пель	симиця	пиянь завод
Finnish	juoda	juo-ma	juo|ja	olut·tehdas
Estonian	jooma	joo|gi	joo|	õlle·tehas
Hungarian	iszik	ital	iv|ó	sör·főzde

The symbols '·', '-' and '|' stand for compounding, inflection and derivation, respectively.

5.4 Pronouns and determiners

The distinction between pronoun and determiner is not widely made in traditional grammars of most Uralic languages. Words which may be considered both pronouns and determiners are lumped into a single morphosyntactic class (usually pronoun). Consider the following examples involving the word 'this'

	'I see this house.'	'I see this.'
North Sámi	Oainnán dán viesu.	Oainnán dán.
Erzya	Неян те$_{det}$ кудонть.	Неян тень$_{pron}$.
Finnish	Mä näen tämän$_{pron}$ talon.	Mä näen tämän$_{pron}$.
Estonian	Ma näen selle$_{pron}$ maja.	Ma näen selle$_{pron}$.
Hungarian	Nézem ezt$_{det/noun}$ a$_{art}$ házat.	Nézem azt$_{det/noun}$

In traditional grammars of North Sámi, Finnish and Estonian both the pronominal and the modifier analyses of 'this' are classified as pronouns. In Hungarian and Erzya, a distinction is made, with Hungarian making a pronoun/determiner distinction and Erzya making a distinction between quantifier (determiner) and nominalised quantifier.

If we consider a standard definition of *pronoun* to be 'that which stands in place (pro-) of a noun phrase (-noun)' then we can see that in the above, only the tools for Erzya follow this. The other languages leave the distinction to tools later in the pipeline.

	North Sámi	Erzya	Finnish	Estonian	Hungarian
and	ja+CC	марто+Po+COM	ja Part	ja+J	és /CONJ
very	hui+Adv	пек+Adv+AdA	tosi Part	väga+Adv	nagyon /ADV
under	vuolde+Po	алов+Po+Lat	alle Part	alla+K	alatt /POSTP
now	dál+Adv	ней+Adv+Temp	nyt Part	praegu+Adv	most /ADV
hello	bures+Interj	шумбрачи+Interj+Formulaic	moi Part	tere+I	szia /UTT-INT

Table 3: Some examples of non-inflecting words with divergent morphological and syntactic annotation. In terms of morphology, the transfer of these tags may be a simple one-to-one substitution. However the syntactic environments may vary substantially.

5.5 Non-inflecting words

All languages in the Uralic family have a wide variety of non-inflecting word forms. Depending on the grammatical tradition followed by the language resource these may be simply lumped into a single class, or they may have extensive syntactic or semantic subcategorisation. Table 3 gives a number of examples of non-inflecting words and the equivalent morphological analyses they receive in each of the languages we are studying. To a machine translation practitioner, these distinctions are largely superfluous, *ja* in North Sámi will be translated as *ja* in Finnish and *ja* in Estonian. However, the distinctions may be vital for the intervening disambiguation tools, and as such need to be taken into account.

6 Guidelines

6.1 Separation of lexicon and morphotactics

One of the main components of any rule-based system for morphologically-complex languages is a lexicon consisting of stems and inflectional/derivation categories. In some cases, such as for Finnish, these are partly provided by a state institution, such as a language board. In other cases they are the product of many years of work.

Although categorising stems for inclusion in a morphological lexicon (many contain over 100,000 entries) can take a substantial amount of work, even if done semi-automatically, implementing the morphotactics (that is, the rules covering inflection, derivation and compounding) may take substantially less time.

6.2 Maximise parallelism

In line with the Universal Dependencies project (see 7), we propose the adoption of a principle of maximum parallelism. In short "things that are the same should be tagged the same". We do not propose that this should mean that all distinctions should be made in all languages. For example, those Uralic languages without object conjugation should not be required to adopt the agreement tags of those that have it. But it should be possible to come up with principled and consistent guidelines for closed categories.

7 Universal dependencies

Universal dependencies is a large multi-language project [7] aiming at common tagset for part-of-speech, morphosyntactic features and dependency relations. We do not propose adopting the exact tagset of the universal dependency project. Most projects working on Uralic languages have been ongoing for many years and the tools that they create are used for more than just machine translation. What we find more important is to adopt, or make available tools based on a consistent theoretical background and consistent morphosyntactic description. This could form the basis of a kind of *universal* morphosyntactic interlingua for the Uralic languages. These tools do not have to replace the current tools, and may be automatically generated from them, but they must be consistent. A systematic mapping needs to be considered while developing. The national Uralic languages have specifications for universal dependencies [8, 9, 10]. But these specifications differ in unnecessary ways. For example, consider the annotation of 'that house' in the two treebanks for Finnish: Turku Dependency Treebank (TDT) and FinnTreeBank (FTB); and Hungarian:

	this		house
Finnish (TDT)	tämä$_{PRON}$		talo$_{NOUN}$
Finnish (FTB)	tämä$_{DET}$		talo$_{NOUN}$
Hungarian	az$_{PRON}$	a$_{ART}$	ház$_{NOUN}$

8 Concluding remarks

Rule-based machine translation provides a fascinating basis for exploring real linguistic differences between the Uralic languages. However, as we

have shown, in current state-of-the-art tools, real linguistic differences are hidden behind a combination of incompatible tagsets and idiosyncratic traditional grammatical norms. We do not propose that the North Sámi adopt the Finnish norms or the Hungarians the Erzya norms, instead we propose developing a common morphological annotation scheme for the Uralic languages based on guidelines of the Universal dependencies project. It is not our aim for this to supercede national standards, but provide a common bridge between them to facilitate the cross-linguistic study and functional rule-based machine translation.

Acknowledgements

Heiki-Jaan Kaalep, Jack Rueter, László Tihany as well as the anonymous reviewers have all contributed to the language examples, the remaining mistakes are ours.

A Example of Universal dependencies for Uralic languages

Example is shown in table 4.

References

[1] Mikel L. Forcada, Mireia Ginestí Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Gema Ramírez-Sánchez, Felipe Sánchez-Martínez, and Francis M. Tyers. Apertium: a free/open-source platform for rule-based machine translation platform. *Machine Translation*, 2010.

[2] Pim Otte and Francis M.Tyers. Rapid rule-based machine translation between dutch and afrikaans. In *Proceedings of the 15th conference of the European Association for Machine Translation, 30-31 May 2011, Leuven, Belgium*, pages 153–160, 2011.

James	*ja*	*Mary*
PROPN	CONJ	PROPN
Number=Sing|Case=Nom		Number=Sing|Case=Nom
leaba	*gárdimis*	.
VERB	NOUN	PUNCT
Mood=Ind|Tense=Pres|Person=3|Number=Dual	Number=Sing|Case=Loc	

Джеймс	*марто*
PROPN	CONJ
Number=Sing|Case=Nom|Definite=Ind	
Марит	*садпиресэ-*
PROPN	NOUN
Number=Plur|Case=Nom|Definite=Ind	Case=Ine|Definite=Ind
-ть	.
VERB	PUNCT
Mood=Ind|Tense=Pres|Pers[subj]=3|Number[subj]=Plur	

James	*ja*	*Mary*
PROPN	CONJ	PROPN
Number=Sing|Case=Nom		Number=Sing|Case=Nom
ovat	*puutarhassa*	.
VERB	NOUN	PUNCT
Mood=Ind|Tense=Pres|Person=3|Number=Plur	Number=Sing|Case=Ine	

James	*ja*	*Mary*
PROPN	CONJ	PROPN
Number=Sing|Case=Nom		Number=Sing|Case=Nom
on	*aias*	.
VERB	NOUN	PUNCT
Mood=Ind|Tense=Pres|Person=3|Number=Plur	Number=Sing|Case=Ine	

James	*és*	*Mary*
PROPN	CONJ	PROPN
Number=Sing|Case=Nom		Number=Sing|Case=Nom
kértben	.	
NOUN	PUNCT	
Number=Sing|Case=Ine		

Table 4: An example of applying universal part-of-speech tags and morphological features to the Uralic languages. Note how the massive differences in annotation are reduced to only the linguistically relevant compared to Table 1.

[3] Jack Rueter. *Adnominal person in the morphological system of Erzya*. PhD thesis, 2010.

[4] Tommi A Pirinen. Omorfi–Free and open source morphological lexical database for Finnish. In *Nordic Conference of Computational Linguistics NODALIDA 2015*, page 313, 2015.

[5] V. Trón, A. Kornai, G. Gyepesi, L. Németh, P. Halácsy, and D. Varga. Hunmorph: open source word analysis. In *Proceedings of the Workshop on Software*, pages 77–85. Association for Computational Linguistics, 2005.

[6] Aarne Ranta. *Grammatical framework: Programming with multilingual grammars*. CSLI Publications, Center for the Study of Language and Information, 2011.

[7] Ryan T McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al. Universal dependency annotation for multilingual parsing. In *ACL (2)*, pages 92–97. Citeseer, 2013.

[8] Sampo Pyysalo, Jenna Kanerva, Anna Missilä, Veronika Laippala, and Filip Ginter. Universal dependencies for finnish. In *Nordic Conference of Computational Linguistics NODALIDA 2015*, page 163, 2015.

[9] Kadri Muischnek, Kaili Müürisep, Tiina Puolakainen, Eleri Aedmaa, Riin Kirt, and Dage Särg. Estonian dependency treebank and its annotation scheme. In *Proceedings of 13th Workshop on Treebanks and Linguistic Theories (TLT13)*, pages 285–291, 2014.

[10] Veronika Vincze, Dóra Szauter, Attila Almási, György Móra, Zoltán Alexin, and János Csirik. Hungarian dependency treebank. In *LREC*, 2010.

Obugric Database: Corpus and Lexicon Databases of Khanty and Mansi Dialects

Axel Wisiorek
Ludwig Maximilian University of Munich
Institute for General Linguistics and Language Typology
IT Group for the Humanities
axel.wisiorek@lmu.de

Zsófia Schön
Ludwig Maximilian University of Munich
Institute for Finno-Ugric Studies
zsofia.schoen@gmail.com

December 31, 2015

Abstract

This paper aims to present a comprehensive web-based framework for the storage and advanced retrieval of annotated corpora and corpus-based lexical databases of Khanty and Mansi dialects within the framework of the project *Ob-Ugric database: analysed text corpora and dictionaries for less described Ob-Ugric dialects* (OUDB). The strength of this approach lies in combining semi-automatic annotation using established documentation and analysis tools with modern web technologies and relational databases.

Key aspects are: Extensive annotation, which covers different levels of linguistic description as well as language internal variation; performing intricate concordance searches based on the annotational linguistic metadata, using a well-adapted relational database scheme that allows complex but nonetheless fast and scalable queries over indexed data; making it possible to identify not only single token forms but *constructional patterns* on various linguistic levels, allowing cutting-edge usage-based research including new corpus evaluation methods

such as 'collostructional' analysis; offering a web interface which provides comprehensive access to the corpus and lexicon data from any up-to-date browser, the client-server framework guaranteeing platform independency; establishing a collaborative research platform with a differentiated user management system which enables contributing researchers to upload their material to the database; providing output that conforms to linguistic standards that is simultaneously suitable as an export format for sharing and archiving data.

As OUDB is work in progress, not all of these features have been fully implemented yet, but the main functionality of the projected framework is existent and operational.

1 Introduction

The project *Ob-Ugric database: analysed text corpora and dictionaries for less described Ob-Ugric dialects* (OUDB, since July 2014)[1] and its framework presented in this paper focus, among other things, on developing semi-automatically tagged corpora and lexical databases for dialects of the Khanty and Mansi languages, belonging to the Ob-Ugric branch of the Finno-Ugric language family. Currently, the size of the glossed corpus is about 30,000 tokens, with the total corpus having over 200,000 tokens in approximately 400 texts in IPA transliteration/transcription.

The corpora and databases were initially set up in the course of the project *Ob-Ugric languages: conceptual structures, lexicon, constructions, categories* (BABEL, August 2009–July 2012), which contained two Khanty (Kazym and Surgut) and two Mansi (Northern and Southern) dialects. As this initial project of the universities of Munich, Vienna, Szeged and Helsinki primarily dealt with already published written material, the documentation and analysis software FieldWorks Language Explorer (FLEx)[2] was chosen for the data analysis, which proposes annotations based on the prior input.

As the number of dialects covered grew with OUDB – a cooperation between the universities of Munich and Vienna – data not only increased in volume, but also became more and more heterogeneous: while the extinct Pelym and North-Vagilsk dialects of Mansi are represented by only text editions from the end of the 19th century, the Yugan dialect of Khanty mostly relies on transcribed sound recordings from fieldwork in the 21st century. To accomodate this circumstance, the annotation tool ELAN was added to our tool set for data handling.

[1] http://www.oudb.gwi.uni-muenchen.de/
[2] http://fieldworks.sil.org/flex

2 Technological Framework

OUDB is hosted and maintained by the IT Group for the Humanities of the LMU Munich (ITG), which offers an Apache web server as well as a MySQL server, thus providing a perfect environment for establishing a web-based research platform such as OUDB. Main advantages of this client-server model are platform independency, long-term availability and easy international collaboration [1, 2, p. 45 ff.]. The fundamental database structure and the PHP-based website (including a backend for cooperating researchers) were established in the first phase of the project (BABEL). On this basis, OUDB continues to develop advanced corpus and lexicon tools[3], with expanded filter possibilities, a new interface, faster and more complex queries, and enriched audio data. It features elaborated interlinear glosses of complete texts, an innovative concordancer which makes the annotated corpus data highly searchable for various patterns (phonetic, morphologic, syntactic, semantic, pragmatic), as well as a corpus-based electronic dictionary connected with the concordance module. The following presentation will mainly focus on the database representation of the annotated corpus data and the characteristics of the concordance search.

3 Structure of the Database

3.1 Importing the Data

Audio files are uploaded to the database, together with textual metadata and an IPA transliteration/transcription via the internal section. Each database entry is indexed in the process. The FLEx annotated data is imported via a stand-alone PHP script originally written by Susanne Grandmontagne (ITG) in the first project phase (BABEL) and adapted to the new requirements of the current project, especially to the characteristics of the latest FLEx release (8.2.4.). The XML-encoded FLEx export file is parsed and the lexical or textual information retrieved in the process is imported according to the established database scheme, using the unique flex-generated IDs as primary and foreign keys.

3.2 Data Scheme

Figure 1 shows the representation of the data in the relational database: there is one table containing the textual metadata, one containing the IPA transliteration/transcription

[3]Tools will be provided by the authors on request and are envisaged to be published at completion of the project under a Creative Commons Licence.

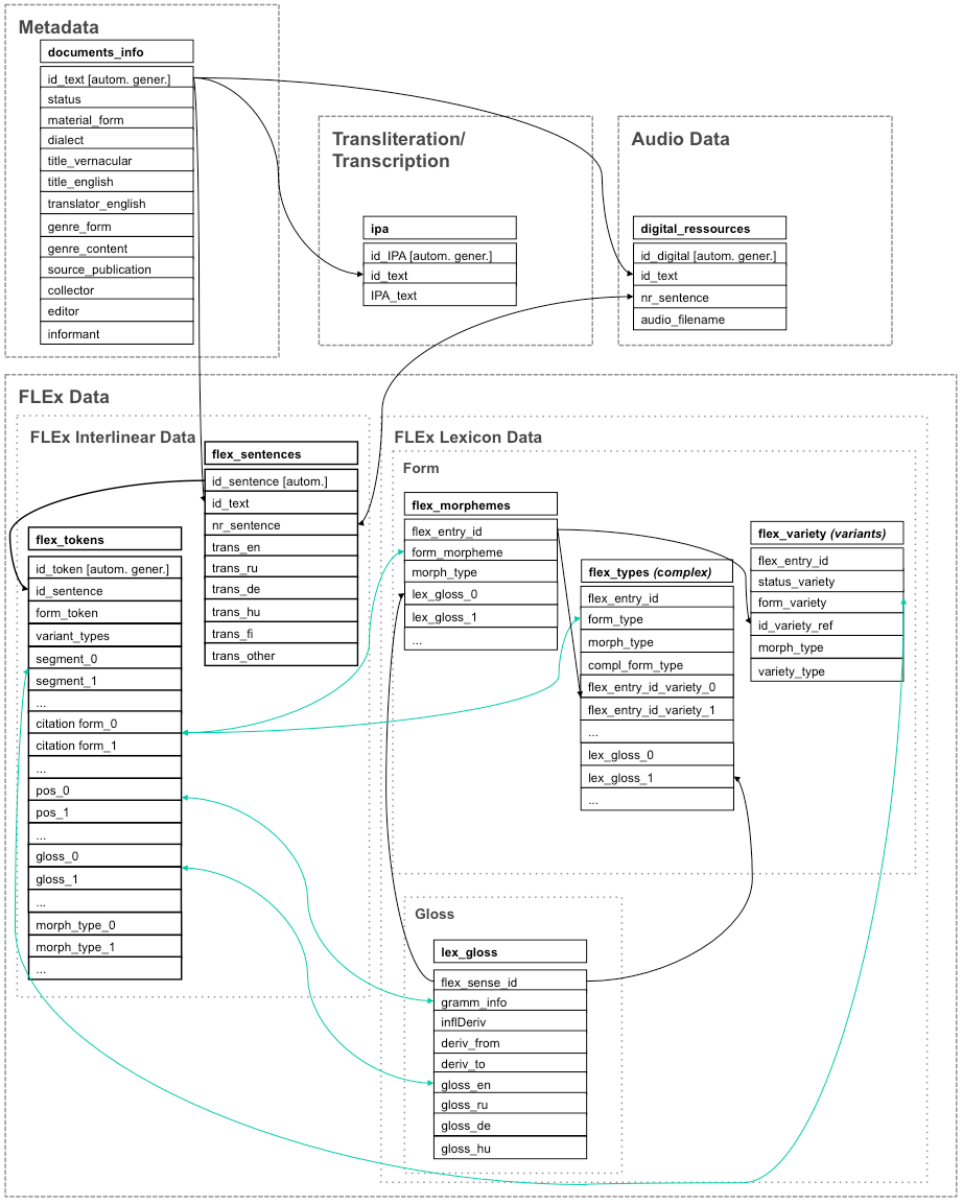

Figure 1: Representation of corpus and lexicon data in the database

data and one containing the audio data. The FLEx annotated corpus and lexicon data are stored in several tables: an annotated token list (a segmentation of each token as well as the citation form, part of speech tag, morpheme type and gloss of each segment) and a list of sentence translations containing the corpus data as well as several tables for the lexical data including morphemes, complex forms, variations of these primary lexicon entries and their semantic values. The aforementioned glosses are either meta-language equivalents for word stems or grammatical category labels for affixes. The foreign key relationships between the data stored in the corresponding tables are indicated by black arrows in Figure 1. For instance, the corpus metadata is connected with the primary corpus data via the `flex_sentences` table based on the unique text and sentence IDs. In a further step the ELAN annotated audio data will be connected with the FLEx data using sentence numbers, which will allow a sentence by sentence triggering of the audio recordings via javascript[4].

As FLEx does not offer the possibility to export text and lexicon data in combination, the information on the relationships between the corpus and the corpus-based lexicon data (indicated by blue arrows in Figure 1) is not part of the imported data. Retrieving corresponding corpus and dictionary entries (e.g. for a concordance result of a dictionary entry) is therefore accomplished by building ad hoc junction tables of the indexed lexicon and corpus data. The relevant columns are indexed using B-trees [3, p. 317–327], allowing fast and scalable searches [1, p. 46]. Like this, the database can grow without the need to change the routines and queries and the architecture of the relational database corpus arising. The lexicon framework is transferable in principle; storing the data in accordance with the relational database model keeps the data usable for later data-mining [4]. The multiple advantages of using relational database storage and querying for large corpora in particular are shown e.g. by Davies [5] (cf. [6, p. 13] and [7, p. 13]); the two main advantages for OUDB are data consistency/integrity through determining constraints and scalability through relational indexing.

4 Analyzing the Data

4.1 User Interface

There are two ways to access the corpus data via the OUDB website: the 'Text Corpus' section (where the texts are available according to their metadata) and the 'Concordance' section (which the following description will be about).

[4]View Text 'piːtʲəŋkəliɣən-oːpisɐɣən A' (ID 732, Surgut Khanty), „Audio + Metadata"; this tool can be adapted for video files as well since it uses HTML 5 standard elements.

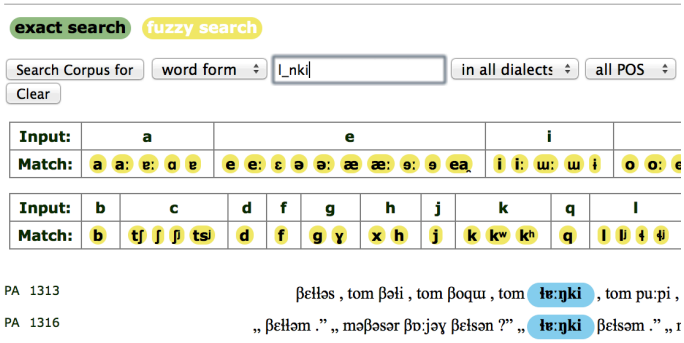

Figure 2: Details of a concordance search

The two main control elements of the concordance web interface (see Figure 2) are the *search bar* with an input field and drop-down menus, which allows the user to filter and sort the search results, and the *IPA input toolbar*. This virtual keyboard allows users to enter IPA characters (client side processed via javascript), and also serves as a matching chart for a fuzzy search within the corpus using ASCII characters as cover symbols for defined IPA character classes (see Figure 2). For matching classes of Ob-Ugric IPA characters with ASCII characters, we use an associative array as a data structure with the ASCII cover symbols as keys and arrays of the matching IPA symbols as related values. We make heavy use of regular expressions within the SQL queries. The results can be sorted in numerous ways, including a reverse alphabetical ordering of the left context (right-to-left). The principal sorting order for the IPA transliteration/transcription of Ob-Ugric languages is implemented in the SQL queries using a sorting array[5].

Using the concordance module to generate a lexicon entry-specific concordance, the corpus-based dictionary provides alternative access to the corpus data in addition to the concordance interface.

4.2 Querying the Corpus Data

Corpus searches rely on SQL as query language. Our framework makes use of the data relations applied in the database scheme to retrieve the relevant tokens. The context of a token is retrieved by multiple self-joins of the token table using the token IDs. Each result of a concordance query is linked with the corresponding location in the corpus, where the relevant token is highlighted (see Figure 3). The corpus is

[5]E.g. reverse sorting of the left context in a KWIC: `FIELD(left(reverse(lk),1),$alph)`.

searchable for word forms, morphemes (stems and affixes) and glosses. It is possible to specify the part-of-speech category of the token in search; wildcards (* or % for an unspecified number of characters, _ for exactly one single character) can be used as well. Regular expressions in queries can be used to search for word forms and lemmata.

The *multiple glosses* search option expands the search from one token form or its gloss (or the glosses of its individual morphemes) to more detailed searches for multiple values in one token or values in different tokens[6]. The user enters a string with two arguments (the two search terms), whereas the optional third argument specifies the window size; without specification, the standard search radius is sentence-wide. There is an 'exact' option, which restricts the search to the given distance of the two tokens instead of a search window of the given size. There is also a 'left/right' option, which takes the order of elements into consideration. Combined with the wildcard % and the part-of-speech restriction for the base token (first argument), advanced and versatile queries are possible, e.g. a search for morphosyntactical patterns such as specific preverbal or postpositional constructions, cf. [8, 9]:

1. `% PTCP.PRS 1`, pos=`preverb` + `right` ⟶ *preverbal present participle construction*
2. `% PTCP% 1`, pos=`pstp` + `left` ⟶ *postpositional participle construction*
3. `%DAT% PASS%`, pos=`ppron` ⟶ *passive construction featuring a pronominal indirect object* (window-size=`sentence`)
4. `LOC PASS% 2`, pos=`subs` + `right` ⟶ *passive construction with locative coded agentive-like argument following immediately or with distance ≤ 2 from the verb*, cf. [10], see Figure 3.

A search for the occurrence of two different glosses in the same token is possible as well, by defining a window size of `0`. This way, in combination with a wildcard, the concordance can not only be used to search for a specific form or gloss (or a combination of these), but for all occurrences of a part-of-speech category:

1. `%SG% LOC 0` ⟶ *morpheme chain with any singular possessive suffix and a locative case suffix*
2. `% % 0`, pos=`prvb` ⟶ *complete concordance of the preverbs in corpus.*

[6] This search type uses self-joins of the token list according to established criteria, e.g. `join on (t1.id_token = t2.id_token-1 OR t2.id_token = t1.id_token-1)` for a search window size of 1 or `join on t1.id_sentence = t2.id_sentence` for a sentence wide search, and complex `where`-restrictions using joins on the metadata, e.g. `where (t1.gls_0 LIKE 'squirrel' OR t1.gls_1 LIKE 'squirrel' ...) AND (t2.gls_0 LIKE 'LOC' OR t2.gls_1 LIKE 'LOC' ...)`.

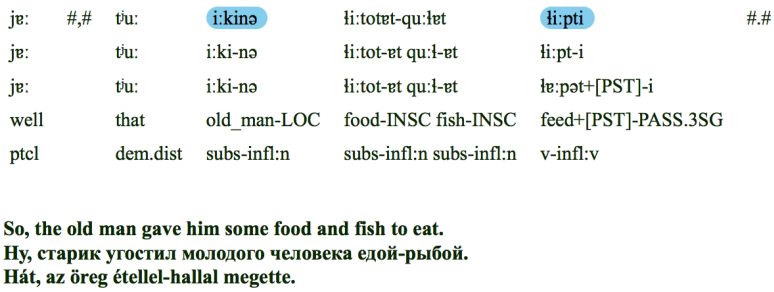

Figure 3: Passive construction with locative coded agentive-like argument

This presented search syntax is only a sketch of what will follow. It will be expanded and universalized, allowing the definition of window size and part-of-speech categories directly in the input and keeping existing query syntaxes like BNC or CQP in mind (cf. [5] and [2]). Our main goal will be the expansion of this multiple gloss search framework to a generalized construction search framework in which each base token of a construction represents this construction (as its head) and can be recursively be part of a bigger construction, establishing a free morphosyntactic constructional search syntax that will be much more adaptable than a linear selection of categories e.g. via selection menus. This expanded search functionality will feature nested queries, each subquery embodied by bracketing and corresponding in principle one binary *multiple glosses* SQL query as shown above, where each base token will function as an identifier for each sub-construction in the complex construction query. Here are two examples for possible nested construction queries:

1. `((%=v %=prvb)1-left %LOC%=ppron)clause` ⟶ *a clause represented by a verb phrase featuring a preverb and a locative coded pronoun*

2. `(PST=v (%=pstp PTCP.PRS=v)clause)sentence` ⟶ *a complex monofinite sentence construction featuring an anteriority postpositional participle construction.*

Exploiting the multilayered, structured representation of the linear speech data in the relational database (e.g. clause/sentence IDs in combination with token IDs), it becomes possible to express a combination of morphologic, syntactic as well as pragmatic or semantic features in one query, forming a complex linguistic pattern and displaying this construction in context. For the given corpus of about 30,000 tokens, the queries show a good performance[7].

[7]For instance it takes 75 ms runtime for the query for preverbal present participle constructions (see above). As the OUDB framework is developed primarily as an integrated research environment connecting

5 Output of Data

The glossed corpus data is compiled and displayed on the website sentence-by-sentence in an interlinearized display style following the Leipzig Glossing Rules [11], with additional lemmatization and part-of-speech data, including English, German, Russian and Hungarian translations. Each token and sentence is accessible by its ID, which is used to connect a KWIC result with the glossed text and to highlight the relevant token(s) (see Figure 3).

6 Future Goals

As OUDB is work in progress, there will be a constant expansion of the range of functionalities offered by our frameworks. As regards the corpus, there will be two main updates. Firstly, an export tool for the preparation of structured data for client-based evaluation as well as for possible archiving, and the accompanying XML output implementation, will be realized. Secondly, we will develop a syntactic and pragmatic annotation system compatible with our existing database scheme. This forthcoming semi-automatic annotation tool, which is already rudimentary implemented, will use the existing FLEx data (esp. part of speech data) for providing a parenthesized annotation line of each sentence using constituent analysis rules. This annotation line can be manually checked and complemented by the annotator with additional syntactic and pragmatic tags as well as additional levels of syntactic analysis (clause). The parenthesized annotation data is then saved in an extra table in the database and simultaneously parsed in a multidimensional array[8], which is used to update the entries in the `flex_tokens` table with their corresponding syntactic and pragmatic annotations. These additional layers of annotation (which will be included in the interlinearized presentation of the corpus)[9] expand the search functionality for constructions even further. Through providing a clause-specific search window, a much more precise identification of syntactical patterns will be possible.

Regarding the concordancer, we are planning an extension which will enable ad-

corpus, lexicon and audio data of the small heterogeneous corpora of the Ob-Ugric languages (e.g. including language specific IPA-ASCII-translation rules in the corpus and lexicon search tools), the application for bigger corpora is not main objective, but we are generally working on improving the performance through extended indexing and enhanced queries on the basis of which the applicability for larger corpora will be evaluated.

[8] This php parsing module will equally be used in the intended construction query system.

[9] In this context, a script for the online visualization of syntactic trees developed at the ITG (LMU Munich) for the *Biblia Hebraica transcripta* (Richter, Eckardt, Specht, Argenton, Zirkel, Riepl, Teuber) will be adapted.

vanced statistical testing. As the basic implementation of a collocational analysis is already implemented in the concordancer with the *multiple glosses* option (see above), the frequency data of these query results can easily be obtained and processed with statistical algorithms, also incorporating measurements of effect size. Thanks to the (already implemented, and in the future expanded) construction search functionality this framework is especially suitable for new construction-based corpus analysis methods such as the 'collostructional' analysis, a constructional grammar-based extension of collocational analysis proposed by Stefanowitsch and Gries [12] where the p-values of a Fisher's exact test resp. the odds ratio are used as a measure of the association strength of a lexeme in a construction. We will be looking into the possibility of using n-gram frequency tables (resp. views) as proposed by Davies [5] for faster collocational analysis, as well as possible construction tables, building a kind of 'constructicon' [13], e.g. containing frequency information of lexical units concerning a certain slot of a construction.

7 Conclusion

As outlined in this paper, OUDB aims to give researchers around the world a server-based – thus client-independent – corpus and lexicon tool that will make corpora of the less described Ob-Ugric dialects available and accessible in connection with lexical and audio data. Thus, this multipurpose corpus data will serve not only language documentation [6, p. 13 f.], but can also serve as research material for typologists and variational or cognitive linguists. In using free Software such as MySQL and PHP, the framework we developed imposes no restrictions on providing and sharing modules.

Using the indexed, semi-automatically annotated (and thus very accurate) corpus data, complex constructional pattern queries are possible, allowing users to tackle advanced morphosyntactic questions. Through the planned standard format export function, researchers will be able to retrieve data for their own evaluation (using R, Perl etc.). OUDB can be considered part of a greater research program which aims to provide and share corpus data in a standardized way and builds on extensive annotation as a way of enriching the primary speech data, thus allowing sophisticated linguistic investigation of complex patterns of language use.

References

[1] Tony McEnery and Andrew Hardie. *Corpus Linguistics: Method, Theory and Practice.* Cambridge Textbooks in Linguistics. Cambridge University Press, 2011.

[2] Andrew Hardie. CQPweb—combining power, flexibility and usability in a corpus analysis tool. *International journal of corpus linguistics*, 17(3):380–409, 2012.

[3] Thomas Ottmann and Peter Widmayer. *Algorithmen und Datenstrukturen*. Spektrum, Heidelberg, 1996.

[4] Michael Stonebraker and Joey Hellerstein. What goes around comes around. *Readings in Database Systems*, 4, 2005.

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

[6] Stefan Th. Gries. What is corpus linguistics? *Language and linguistics compass*, 3(5):1–17, 2009.

[7] Stefan Th. Gries and Andrea L. Berez. Linguistic annotation in/for corpus linguistics. http://www.linguistics.ucsb.edu/faculty/stgries/research/InProgr_STG_alb_lingannotcorpling_hboflingannot.pdf, September 2015.

[8] Zsófia Schön. On the Road to a Dialect Dictionary of Khanty Postpositions. In *Septentrio Conference Series*, pages 99–107, 2015.

[9] Jeremy Bradley. Corpus. mari-language. com: A Rudimentary Corpus Searchable by Syntactic and Morphological Patterns. In *Septentrio Conference Series*, pages 57–68, 2015.

[10] Andrey Filtchenko. The Eastern Khanty locative-agent constructions. In *Demoting the Agent: Passive, Middle and Other Voice Phenomena*, pages 47–82. 2006.

[11] Bernard Comrie, Martin Haspelmath, and Balthasar Bickel. The Leipzig Glossing Rules. Conventions for interlinear morpheme by morpheme glosses. https://www.eva.mpg.de/lingua/pdf/Glossing-Rules.pdf, November 2015.

[12] Anatol Stefanowitsch and Stefan Th. Gries. Collostructions: Investigating the interaction of words and constructions. *International journal of corpus linguistics*, 8(2):209–243, 2003.

[13] Charles J. Fillmore. Border conflicts: FrameNet meets construction grammar. In *Proceedings of the XIII EURALEX International Congress (Barcelona, 15-19 July 2008)*, pages 49–68, 2008.

Building grammatical analysers for Uralic languages – a tutorial

Trond Trosterud
Giellatekno
UiT The Arctic university of Norway
trond.trosterud@uit.no

November 15, 2015

Abstract

The tutorial gives an introduction to the Giellatekno – Divvun infrastructure for building language technology for morphology-rich languages. The whole infrastructure and all needed compilers are available under open licenses. In addition to a setup for grammatical analysers, the infrastructure also makes it possible to build proofing tools, e-dictionaries and ICALL programs.

1 Tutorial

The Giellatekno and Divvun groups at UiT The Arctic University of Norway have developed an infrastructure for making programs for grammatical analysis based upon finite-state transducers, cf. [1] and [2]. The infrastructure may be downloadet and set up on local machines[1]

In addition to a modular setup for the different parts of the analysers, the infrastructure also offers a way of making and conducting a wide range of regression and developmental tests.

Via the infrastructure there is also a direct setup for a wide range of applications: e-dictionaries, [3], ICALL applications, [4], and setup for proofing tools [5].

[1] Download and installation of necessary auxiliary programs are explain at http://giellatekno.uit.no/doc/infra/GettingStarted.html.

The tutorial will look at various Uralic languages as test cases, and show how to make analysers and practical applications for them.

References

[1] Sjur N. Moshagen, Tommi A. Pirinen, and Trond Trosterud. Building an open-source development infrastructure for language technology projects. In *Proceedings of the 19th Nordic Conference of Computational Linguistics, NODALIDA 2013, May 22-24, 2013, Oslo University, Norway*, pages 343–352, 2013.

[2] Sjur N. Moshagen, Tommi A. Pirinen, Trond Trosterud, and Francis M. Tyers. Open-source infrastructures for collaborative work on under-resourced languages. In *CCURL 2014: Collaboration and Computing for Under- Resourced Languages in the Linked Open Data Era*, pages 71–77, 2014.

[3] Lene Antonsen, Ryan Johnson, and Trond Trosterud. Using finite state transducers for making efficient reading comprehension dictionaries. In *Proceedings of the 19th Nordic Conference of Computational Linguistics, NODALIDA 2013, May 22-24, 2013, Oslo University, Norway*, pages 59–71, 2013.

[4] Lene Antonsen, Ryan Johnson, Trond Trosterud, and Heli Uibo. Generating modular grammar exercises with finite-state transducers. In *Proceedings of the 19th Nordic Conference of Computational Linguistics, NODALIDA 2013, May 22-24, 2013, Oslo University, Norway*, pages 27–38, 2013.

[5] Sjur N. Moshagen. A language technology test bench – automatized testing in the divvun project. In *Proceedings of the Workshop on NLP for Reading and Writing – Resources, Algorithms and Tools*, pages 19–21, 2008.

Universal Dependencies for Finno-Ugric Languages

Veronika Vincze[1], Filip Ginter[2], Tommi Pirinen[3], Francis Tyers[4]
[1]University of Szeged
vinczev@inf.u-szeged.hu
[2]University of Turku
figint@utu.fi
[3]Dublin City University
tommi.pirinen@computing.dcu.ie [4]University of Tromsø
francis.tyers@uit.no

July 14, 2016

Part-of-speech tagging and syntactic parsing have been popular research areas in natural language processing. Recently, several shared tasks have been organized that aimed at the morphological and syntactic parsing of several languages [1, 2]. However, comparison of results achieved for different languages is not straightforward due to the use of language-specific morphological tagsets, language-specific syntactic labels and language-specific annotation principles. To overcome these difficulties, researchers within the project Universal Dependencies (UD) have been developing a "universal", i.e. a language-independent morphological and syntactic representation that can be successfully applied in multilingual morphological and syntactic parsing [3]. At this time, treebanks have been created for 33 languages and many more are to be expected in the next months.

In this tutorial, we focus on treebanks for Finno-Ugric (FU) languages that have been made available by the UD community, i.e. Finnish, Hungarian and Estonian. We first give a short intorduction to UD morphology and syntax, then we discuss specific morphological features and values for FU languages, e.g. in the case of possessive

markers and object-verb agreement. Later, we analyze in detail the specific dependency labels and annotation practice for FU languages. The linguistic phenomena to be discussed include empty copulas, multiword named entities and extended dependency labels used for adverbials.

In the second part of the tutorial, we practically show how to build UD treebanks and the audience will have the chance to annotate some sentences according to the UD principles, using the annotation tool BRAT[1].

References

[1] Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Yuval Marton, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, and Alina Wróblewska. Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 146–182, Seattle, Washington, USA, October 2013. ACL.

[2] Djamé Seddah, Sandra Kübler, and Reut Tsarfaty. Introducing the SPMRL 2014 Shared Task on Parsing Morphologically-rich Languages. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 103–109, Dublin, Ireland, August 2014. Dublin City University.

[3] Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[1] `http://brat.nlplab.org/`