7th SaLTMiL Workshop on Creation and use of basic lexical resources for less-resourced languages

LREC 2010, Valetta, Malta, 23 May 2010

Workshop programme

09.00 **Registration**

09.30 **Opening**

09.45 **Invited talk: Marc Kemps-Snijders**, LAT team at the Max Planck Institute at Nijmegen: "Relish: rendering endangered languages lexicons interoperable through standards harmonisation"

10.30 **Coffee break**

11.00 **Invited talk: Antton Gurrutxaga, Igor Leturia, Eli Pociello, Iñaki San Vicente, and Xabier Saralegi**, Elhuyar Foundation: "Exploiting Internet to build language resources for less resourced languages"

11.45 **Oral papers (20+5 min.):**

Tommi A. Pirinen and Krister Lindén: "Finite-State Spell-Checking with Weighted Language and Error Models–Building and Evaluating Spell-Checkers with Wikipedia as Corpus"

Aric Bills, Lori S. Levin, Lawrence D. Kaplan, and Edna Agheak MacLean: "Finite-State Morphology for Iñupiaq"

12.35 **Poster session:**

Marco Passarotti: "Leaving Behind the Less-Resourced Status. The Case of Latin through the Experience of the *Index Thomisticus* Treebank"

Anna Björk Nikulásdóttir and Matthew Whelpton: "Extraction of Semantic Relations as a Basis for a Future Semantic Database for Icelandic"

Gábor Prószéky, Attila Novák, István Endrédy, Beatrix Oszkó, László Fejes, Sándor Szeverényi, Zsuzsa Várnai and Beáta Wagner-Nagy: "Nganasan – Computational Resources of a Language on the Verge of Extinction"

Géraldine Walther and Benoît Sagot: "Developing a large-scale lexicon for a less-resourced language: Sorani Kurdish"

Hrafn Loftsson, Jökull Yngvason, Sigrún Helgadóttir and Eiríkur Rögnvaldsson: "Developing a PoS-tagged corpus using existing tools"

13.20 **Panel**: Less resourced languages and Language technology. Short- and medium-term objectives (SaLTMiL)

14.00 **Closing**

Workshop Organisers

Kepa Sarasola, Euskal Herriko Unibertsitatea, Donostia
Francis M. Tyers, Universitat d'Alacant, Alacant
Mikel L. Forcada, Dublin City University, Dublin

Workshop Programme Committee

Iñaki Alegria, Euskal Herriko Unibertsitatea, Donostia
Núria Bel, Universitat Pompeu Fabra, Barcelona
Lars Borin, Göteborgs universitet
Hrafn Loftsson, Reykjavík University
Felipe Sánchez-Martínez, Universitat d'Alacant
Kevin Scannell, Saint Louis University, USA
Trond Trosterud, Universitetet i Tromsø, Norway

Additional Referees

Per Langgård, Oqaasileriffik (Language Secretariat, Nuuk, Greenland)
Paul Meurer, Universitet i Bergen
Sjur Moshagen, Divvun (Norwegian Sámi Parliament)
Eva Navas, Euskal Herriko Unibertsitatea
David Tomás, Universitat d'Alacant

Table of Contents

Author Index

Relish: rendering endangered languages lexicons interoperable through standards harmonisation

Marc Kemps-Snijders

Max Planck Institute for Psycholinguistics
PO Box 310
6500 AH Nijmegen
The Netherlands

When a lexicon constitutes the only record of a dying or already extinct language, it can contribute unique linguistic and cultural information to our store of scientific knowledge. And making it interoperable with other lexical data becomes a critical research priority. However, despite the support accorded to initiatives to develop digital standards for language documentation within both the US and Germany, there still exist major barriers to lexicon interoperability. The most significant barrier is that standards-setting bodies have arrived at different standards for format and markup on the two sides of the Atlantic. On the European side the main focus has been towards the ISO 24623 Lexical Markup Framework (LMF) and the ISO 12620 Data Category Registry (DCR) while at the American side the Lexicon Interchange Format (LIFT) and GOLD have been the centre of attention. As a consequence, within each national community, divergences exist in lexicon format and markup, in part because field linguists have hitherto relied on software which does not offer the linguist adequate support in choosing structural or linguistic categories.

The Relish project will promote language-oriented research by addressing a two-pronged problem: (1) the lack of harmonization between digital standards for lexical information in Europe and America, and (2) the lack of interoperability among existing lexicons of endangered languages, in particular those created with the Shoebox lexicon building software. Focusing on six to eight lexicons of endangered languages, the project will establish a unified way of referencing lexicon structure and linguistic concepts, and develop a procedure for migrating these heterogeneous lexicons to a standards-compliant format. Once developed, the procedure will be generalizable to the large store of lexical resources involved in the LEGO and DoBeS projects. The project will produce significant benefits both to the user community and to the organizations which support their research.

As a first step the linguistic concepts expressed in GOLD will be harmonized with those already present in the Data Category Registry thus providing a unified and persistent reference point for concepts used on both sides of the Atlantic. Also, the most commonly used Shoebox markers for the Multi Dictionary Formatter (MDF) will be made available as data categories in the Data Category Registry to provide further support for lexica created using the Shoebox tool. Focusing on six to eight lexicons of endangered languages, the project will establish a unified way of referencing lexicon structure and linguistic concepts, and develop a procedure for migrating these heterogeneous lexicons to a standards-compliant format. To complement this bottom-up approach the Relish project uses a top-down approach analyzing existing standards for lexical resources (GOLD/LIFT and DCR/LMF) to identify commonalities and differences at the conceptual and structural level. An attempt is made to harmonize these standard approaches to come to a single interchange format making it possible to exchange lexica in a unified manner. Existing software tools as LEXUS and SOLID will be modified to support the interchange scenarios.

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 1

Exploiting the Internet to build language resources for less resourced languages

Antton Gurrutxaga, Igor Leturia, Eli Pociello, Iñaki San Vicente, Xabier Saralegi
Elhuyar Foundation
Zelai Haundi kalea 3
Osinalde Industrialdea
20170 Usurbil, Spain
E-mail: {a.gurrutxaga, i.leturia, e.pociello, i.sanvicente, x.saralegi }@elhuyar.com

Abstract

This paper aims to present a general view of the Elhuyar Foundation's strategy to build several types of language resources for Basque out of the web in a cost-efficient way. We collect various types of corpora (specialized, general, comparable, parallel…) from the Internet by means of automatic tools, and then other kinds of resources (terminology, ontologies, etc.) are built out of them also using other automatic tools that we have developed. We have also built a web-as-corpus tool to query the web directly as if it were a corpus. In the end of the paper, we describe two experiments that we have performed to prove the validity of the approach: one that automatically collects specialized corpora in Basque and extracts terminology out of them, and another one that automatically collects a comparable corpus and extracts bilingual terminology out of it, using web-derived contexts to improve the results. In our opinion, the strategy is very interesting and attractive for other less resourced languages too, provided they have enough presence on the web.

1. Motivation

Any language aiming to survive in a world that is becoming more intercommunicated and global day by day, and to be used normally in education, media, etc., must necessarily have at its disposal language resources such as dictionaries or corpora, preferably in digital form. The ever-growing presence of ICTs in everyday life adds to these requisites the existence of language technologies and NLP tools for that language, which in turn also need electronic dictionaries and corpora in order to be developed. Therefore, the need for lexical resources and corpora of any language intending to be modern is undeniable.

Besides, modern lexicography and terminology is hardly done based solely on experts' knowledge or intuition; empirical evidence is needed or previous use at least is studied, and these are provided by corpora. And there are many tools that ease the process of building lexical or terminological dictionaries by making use of NLP and statistical methods to automatically extract candidates out of corpora.

So it is clear that corpora of any kind (monolingual, parallel, comparable...) are a very valuable resource for many aspects of the development of a language. And generally, the bigger the corpora, the better the results obtained from them. But less resourced languages are not exactly rich in corpora, let alone big corpora: on the one hand, building a corpus in the classical way, i.e. out of printed texts, is normally a very costly process; on the other, the number of language experts or researchers dealing with these languages is much smaller than that of major languages.

However, the Internet provides a huge number of texts in a digital and easy to manipulate standard format. For any less resourced language there are bound to be many more texts on the web than in any corpus. That is why turning to the Internet to build corpora (and, through them, other kinds of resources such as dictionaries, terminology lists or statistical machine translation systems) is a very attractive and logical choice for less resourced languages.

The Elhuyar Foundation has been exploring this path for the last few years in order to build language resources for the Basque language. In the following sections we will explain the problems we have encountered and the approaches we have followed for each kind of resource, the former presumably being similar to those that other less resourced languages might encounter, and the latter hopefully being applicable to them too.

2. Using the web to build corpora

2.1 Monolingual specialized corpora

Specialized corpora, that is, corpora made out of texts belonging to a certain domain or topic, are a very valuable resource for terminology tasks as well as for most NLP tasks. Major languages often build specialized corpora by simply crawling one website, or a few, dedicated to the topic and which contain a large number of texts on it. Sometimes this method is combined with some machine-learning filter tailor-made for the specific topic, in order to follow links to external sites, too. But for Basque (and most likely for many other less-resourced languages) there are not many websites that are specialized in a topic and which contain a significant number of texts, or at least there are not for any topic one can think of. And the process of building machine-learning filters is too costly due to the lack of training data.

Hence, for Basque a whole web-wide approach must be used, using search engines. The *de facto* standard process major languages use for collecting web-wide specialized corpora, which was first used by the BootCaT tool (Baroni & Bernardini, 2004), consists of starting from a given list of words, asking APIs of search engines for

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 3–11

random combinations of them and downloading the returned pages. However, the topic precision that can be obtained by this methodology has scarcely been measured, and a small evaluation performed on the original BootCaT paper hints that one third of the texts could be unrelated to the topic. And this precision is much worse when searching for corpora in the Basque language. Some experiments we have performed show that this can drop to only 25% (Leturia et al., 2008a).

The main reasons for this are two: one is that no search engine offers the possibility of returning pages in Basque alone, so when looking for technical words (as is often the case with specialized corpora), it is very probable that they exist in other languages too, and thus the queries return many pages that are not in Basque; the other is that Basque is a morphologically rich language and any lemma has many different word forms, so looking for the base form of a word alone, as search engines do, brings fewer results.

Many other languages suffer from these problems regarding search engines. Less than fifty languages are treated properly by Google, Yahoo or Bing. In the case of Basque, we have solved them to some extent (Leturia et al., 2008b). For the former, we use the language-filtering words method, consisting of adding the four most frequent Basque words to the queries within an AND operator, which raises language precision from 15% to over 90%. For the latter, we solve it by means of morphological query expansion, which consists of querying for different word forms of the lemma, obtained by morphological generation, within an OR operator. In order to maximize recall, the most frequent word forms are used, and recall is improved by up to 60% in some cases.

These two techniques raise the topic precision to the baseline of other languages (roughly 66%). Nevertheless, we have developed a method to try to further improve topic precision and have implemented it in a system to automatically collect Basque specialized corpora from the Internet called AutoCorpEx (Leturia et al., 2008a). Its operation is explained below.

The system is fed with a sample mini-corpus of documents that covers as many sub-areas of the domain as possible –10-20 small documents can be enough, depending on the domain. A list of seed terms is automatically extracted from it, which can be manually edited and improved if necessary. Then combinations of these seed words are sent to a search engine, using morphological query expansion and language-filtering words to obtain better results for Basque, and the pages returned are downloaded. Next, the various cleaning and filtering stages necessary in any corpus collecting process involving the web are performed. Boilerplate is stripped off the downloaded pages (Saralegi and Leturia, 2007) which are then passed through various filters: size filtering (Fletcher, 2004), paragraph-level language filtering, near-duplicate filtering (Broder, 2000) and containment filtering (Broder, 1997). After that we have added a final topic-filtering stage, using the initial sample mini-corpus as a reference and employing document similarity techniques (Saralegi and Alegria, 2007) based on keyword frequencies (Sebastiani, 2002). A manual evaluation of this tool showed that it can obtain a topic precision of over 90%.

2.2 Multilingual domain-comparable corpora

Multilingual corpora are considered comparable if the subcorpora of each of the different languages share some common feature, such as domain, genre, time period, etc. Specifically, the texts of a domain-comparable corpora are all in the same domain. These kinds of resources are very useful for automatic terminology extraction, statistical machine translation training, etc., although they are more difficult to exploit than parallel corpora (because of their smaller alignment level, there is less explicit knowledge to extract). However, parallel corpora of significant size are scarce, especially for less resourced languages, and since comparable corpora are easier to obtain, more and more research is heading towards the exploitation of these kinds of corpora.

With the method described in section 2.1 for collecting monolingual specialized corpora, domain-comparable corpora can also be built (Leturia et al., 2009): we can use a sample mini-corpus for each language and launch the corpus collecting process independently for each of them; if the sample mini-corpora that are used for the domain filtering are comparable or similar enough (ideally, a parallel corpus would be best), the corpora obtained will be comparable to some extent, too. We have implemented this methodology in a tool called Co3 (Comparable Corpora Collector).

We have also developed and tried another variant of this method; it uses only a sample mini-corpus in one of the languages, and translates the extracted seed words (they are manually revised) and the keyword vectors used in the domain-filtering to the other language by means of a bilingual dictionary.

This method, theoretically, presents two clear advantages: firstly, the sample mini-corpora are as similar as can be (there is only one), so we can expect a greater comparability in the end; and secondly, we only need to collect one sample corpus. However, it presents some problems too, mainly the following two: firstly, because dictionaries do not cover all existing terminology, we may have some Out Of Vocabulary (OOV) words and the method may not work so well; secondly, we have to deal with the ambiguity derived from dictionaries, and selecting the right translation of a word is not so easy. To reduce the amount of OOV words, the ones that have been POS-tagged as proper nouns are included as they are in the translated lists, since most of them are named entities. And for resolving ambiguity, for the moment, we have used a naïve “first translation” approach, widely used as a baseline in NLP tasks that involve translation based on dictionaries. An evaluation showed that the results of the dictionary-based method were no worse than those of the two sample mini-corpora method.

2.3 Monolingual general corpora

The web is also used as a source for large general corpora, which are very interesting for tasks such as language standardization, general lexicography, discourse analysis, etc. Again, two approaches exist, one based on crawling and the other on search engines. The crawling method is used in the projects of the WaCky initiative (Baroni et al., 2009), which have collected gigaword-size corpora for German (Baroni and Kilgarriff, 2006), Italian (Baroni and Ueyama, 2006) and English (Ferraresi et al., 2008), with many others on the way. Search engines are used for example by Sharoff (2006), sending combinations of the 500 most frequent words of the language.

Currently, we have ongoing projects for collecting large general corpora for Basque using both methods. The usual cleaning and filtering is done in all cases, and the search engine-based approach uses the aforementioned morphological query expansion and language-filtering words techniques. So far, the crawling-based method has gathered a 250-million-word corpus and the search engine-based method a 100 million word corpus.

2.4 Other kinds of corpora

We have already mentioned that parallel corpora (multilingual corpora made out of texts that are translations, preferably aligned at the sentence level, such as translation memories) are very useful for machine translation, terminology extraction, etc., but are not easy to obtain. However, the web is full of websites with versions in more than one language; specifically, most corporate or public websites that are in a less resourced language also include a version in one or more major languages. This fact has already been exploited for automatically building parallel corpora (Resnik, 1998). In the same line of work, we have an ongoing project, called PaCo2 (Parallel Corpora Compiler) to automatically collect Basque-Spanish or Basque-English parallel corpora from the Internet.

For the near future, we also have an interest in genre-specific corpora. *A priori*, we can expect to be able to collect these kinds of corpora by crawling, at least for some genres such as journalism, blogs, administration, since there are websites with large amounts of content of those genres. For others, genre filters or classifiers would have to be developed. Such tools have been built for major languages, which use punctuation signs or POS trigrams as filtering features (Sharoff, 2006); tests have yet to be carried out to see whether these features work for an agglutinative language like Basque.

3. Building other kinds of resources

3.1 A web-as-corpus tool

A common use of corpora is to use them for linguistic research: querying for one or more words and looking at their counts, contexts, most frequent surrounding words, etc. Some of these data can be obtained by querying a search engine directly; although this has its drawbacks (ambiguity caused by its non-linguistically-tagged nature, you cannot query for the POS, the sort order is anything but linguistically guided, redundancy...), it also has its advantages (the corpus is huge, constantly updated...). Thus, some services that ease the use of the web as a direct source of linguistic evidence, namely WebCorp (Renouf et al., 2007) or KWiCFinder (Flectcher, 2006), have appeared. They query the APIs of search engines for the words the user enters, download the pages they return and show occurrences of the word in a KWiC way.

Such a service is very interesting for Basque or for any language not rich in corpora, but since they rely on APIs of search engines, they pose the problems we have already stated. So we have built a service called CorpEus (Leturia et al., 2007), which solves these by means of morphological query expansion and language-filtering words. It is available for querying at http://www.corpeus.org.

3.2 Terminology

The Elhuyar Foundation has developed several tools to automatically extract monolingual or multilingual terminology out of different kinds of corpora, using a combination of linguistic and statistical methods.

Erauzterm (Gurrutxaga et al., 2004) is a tool for automatic term extraction from Basque corpora, implemented by the Elhuyar Foundation in collaboration with the IXA group. It has reported F measure results of 0.4229 for multi-word terms and 0.4693 for single word terms, and precision values of up to 0.65 for multi-word terms and up to 0.75 for single word terms for the first 2,000 candidates over a corpus on electricity & electronics.

Elexbi (Alegria et al., 2006) extracts pairs of equivalent terms from Spanish-Basque translation memories. It is based on monolingual candidate extraction in Basque (Erauzterm) and Spanish (Freeling), and consequent statistical alignment and extraction of equivalent pairs. It has reported results of up to 0.9 precision for the first 4,000 candidates processing a parallel corpus of 10,900 segments.

AzerHitz (Saralegi, et al., 2008a; Saralegi, et al., 2008b) is a tool to automatically extract pairs of equivalent terms from Basque-English or Basque-Spanish domain-comparable corpora based on context similarity, obtaining a precision of 58% in top 1 and 79% in top 20 for high-frequency words.

The combination of these terminology extraction tools with the corpora collection tools we have mentioned above, provides some semi-automatic ways of building dictionaries out of the web:

- AutoCorpEx collects Basque specialized corpora from the web, and then we obtain lists of terms in Basque by applying Erauzterm to them.
- Co3 can gather English-Basque comparable corpora out of the web, and by applying AzerHitz to them we obtain English-Basque terminology lists.

- PaCo2 will, in a near future, collect Spanish-Basque parallel corpora from the web and then Elexbi will extract Spanish-Basque terminology from them.

The next section describes some experiments we have conducted using the first two, since the corpus collection tool of the third approach is still under development.

3.3 Ontologies

There is also an ongoing project for automatically extracting specialized terminology out of a Basque corpus, in order to automatically (or semi-automatically) enrich existing concept taxonomies such as WordNet, or in order to build domain-specific ontologies. The specialized corpora to be used in this project can also be collected automatically out of the web.

4. Experiments

In this section we will show some experiments we have performed to use the web as "raw material" to build language resources such as corpora and term lists. Our first task will be to explore the possibilities that the web offers for the compilation of terminological dictionaries in Basque, via automatic term extraction from web-corpora. We will use AutoCorpEx for collecting specialized web corpora in Basque and Erauzterm as the Basque term extraction tool. In the second experiment, we enter the field of comparable corpora, and present some experiments that envisage the construction of multilingual terminological resources for language pairs with scarce parallel corpora such as Basque. We use Co3 for compiling the domain-comparable corpora and AzerHitz for extracting bilingual terminology out of them. The experiment aims to improve the performance of the terminology extraction by using the web for collecting additional data on the fly to improve context-similarity computation.

4.1 Monolingual specialized web corpora

The goal of the first experiment is to evaluate the domain precision of the web corpora built with Co3 and of the term lists extracted out of them with Erauzterm.

4.1.1. Design

We collected three specialized corpora in the domains of Computer Science, Biotechnology and Atomic & Particle Physics. The collection of the corpora from the Internet did not have a target size, because the Internet in Basque is not as big as that in other languages, and the number we would want to collect for a particular domain might not exist. So we simply launched the collecting processes and stopped them when the growing speed of the corpora fell to almost zero, thus obtaining corpora that were as large as possible.

Then we applied the terminology extraction process to the corpora and obtained the three term lists. These lists were automatically validated against a recently compiled specialized dictionary, ZT Hiztegia or Basic Dictionary of Science and Technology (http://zthiztegia.elhuyar.org), which contains 25,000 terms, and the online version of Euskalterm, the Basque Public Term Bank (http://www1.euskadi.net/euskalterm/indice_i.htm). The terms not found in those terminological databases were manually validated by experts up to a certain number.

Table 1 shows the size of the corpora obtained, the number of terms extracted and the number of terms validated manually or by the dictionary, for each of the three domains.

4.1.2. Evaluation and results

Firstly, we evaluated the domain precision of the lists obtained from the Internet, by analyzing the distribution of the terms across the domains, taking the domains of the specialized dictionary as a reference. The results of this evaluation are shown in Figure 1.

We can observe that all three lists show peaks in or around their respective domains, which proves that the corpora are indeed specialized to some extent and that the term lists automatically extracted belong mainly to the desired domains.

On the other hand, the Biotechnology corpus appears to be the less specialized one, as its distribution is flatter than the others'. Besides, in that corpus and especially in the Computer Science one, the presence of terms not belonging to the area of science and technology is remarkable. The explanation for this could be that they both are technology domains, and hence are closely related to their application areas; not surprisingly, terms from those applications areas occur in those texts more frequently than in pure science documents.

Figure 2 shows the domain precision of the term extraction for each corpus (relative to valid terms). A distinction between General Physics and Atom & Particle Physics has been made. An explanation for the fact that precision results are considerably better for the former could be that many general terms in Physics occurred along with atomic and particle terminology. We may be able to understand this if we take into account the fact that most of the texts are not the product of communication among specialists, but of popular science or teaching materials.

Regarding recall relative to the ZT Hiztegia (Figure 3), the best results are obtained for Atomic & Particle Physics, while the recall for Biotechnology is the lowest. The overall conclusion could be that the three web corpora are lacking representativeness, and are not good enough for compiling a quality dictionary. There is no single possible explanation for that. For example, in the case of Atomic & Particle Physics, out of the 474 terms included in the dictionary, 150 were not extracted from the web corpus (31.64%). We checked the presence of those 150 terms in the Internet, and 42 of them were not retrieved by Google (using CorpEus). 4 terms are in the Internet, but not in the web corpus, and finally, 104 terms in the web corpus were not extracted by Erauzterm (101 occurring only once).

So the main problem is the recall of the Basque Internet itself (Erauzterm could hardly be blamed for not being

able to extract 101 terms with f = 1).

One possible explanation for this fact could lie in the current situation of Basque terminology and text production. Although Basque began to be used in Science and Technology thirty years ago, it cannot be denied that there is a given amount of highly specialized terminology that is published *ex novo* in dictionaries, with little document support if any. That could be the reason why several terms chosen by experts and published in the dictionary do not occur or occurred only once in the Internet.

Finally, as we can see in Table 2, the manual validation process provided new terms not included in the dictionary. This suggests that the process proposed could be interesting for enriching or updating already existing specialized dictionaries.

More details and results of this experiment can be found in a paper entirely dedicated to it (Gurrutxaga et al., 2009).

Corpus	**Atomic and Particle Physics**	**Computer Science**	**Biotechnology**
Sample corpus size	32 docs, 26,164 words	33 docs, 34,266 words	55 docs, 41,496 words
Obtained corpus size	320,212	2,514,290	578,866
Extracted term list size	46,972	163,698	34,910
Dictionary validated	6,432	8,137	6,524
First 10,000 candidates	2,827	2,755	2,403
Manually evaluated	869	904	628
Terms	628	512	432
Not terms	241	392	196

Table 1. Corpus and term list sizes obtained for each of the three domains

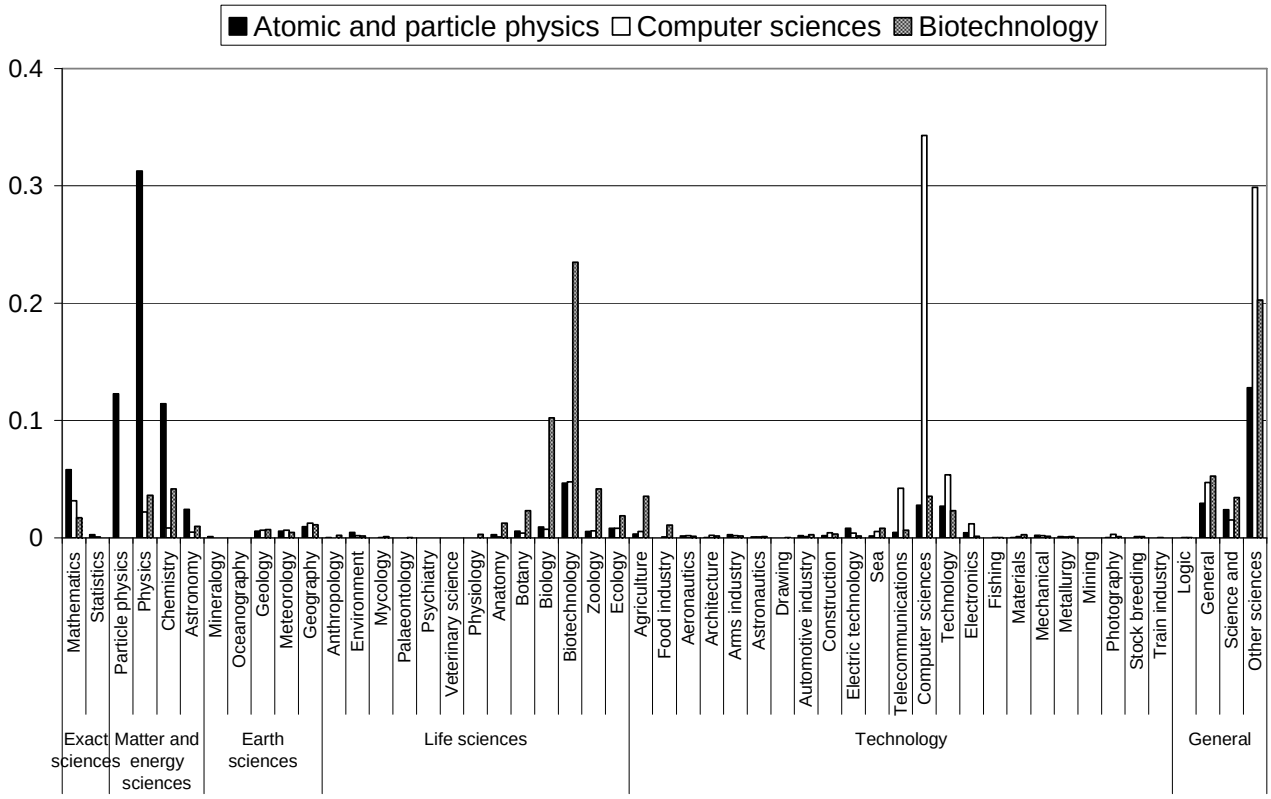

Figure 1. Domain distribution of the extracted term lists

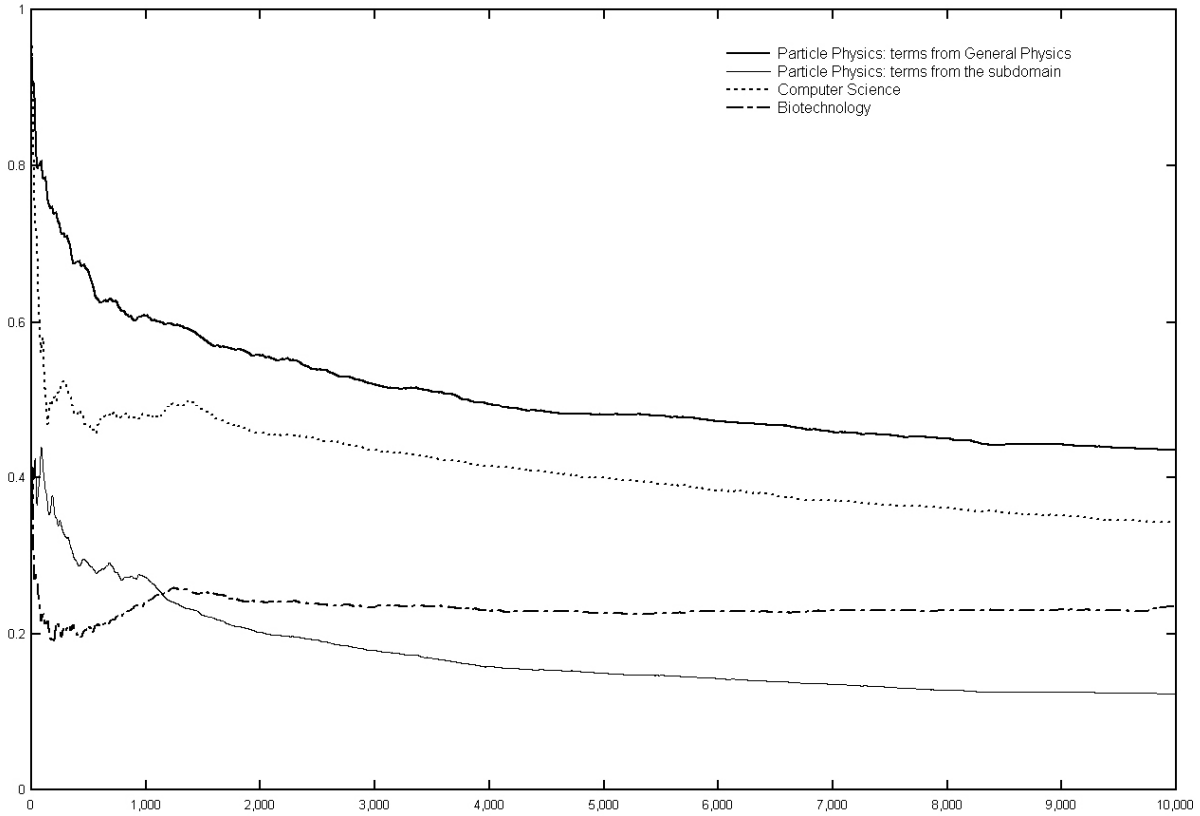

Figure 2. Domain precision of term extraction from each web corpus (relative to validated terms)

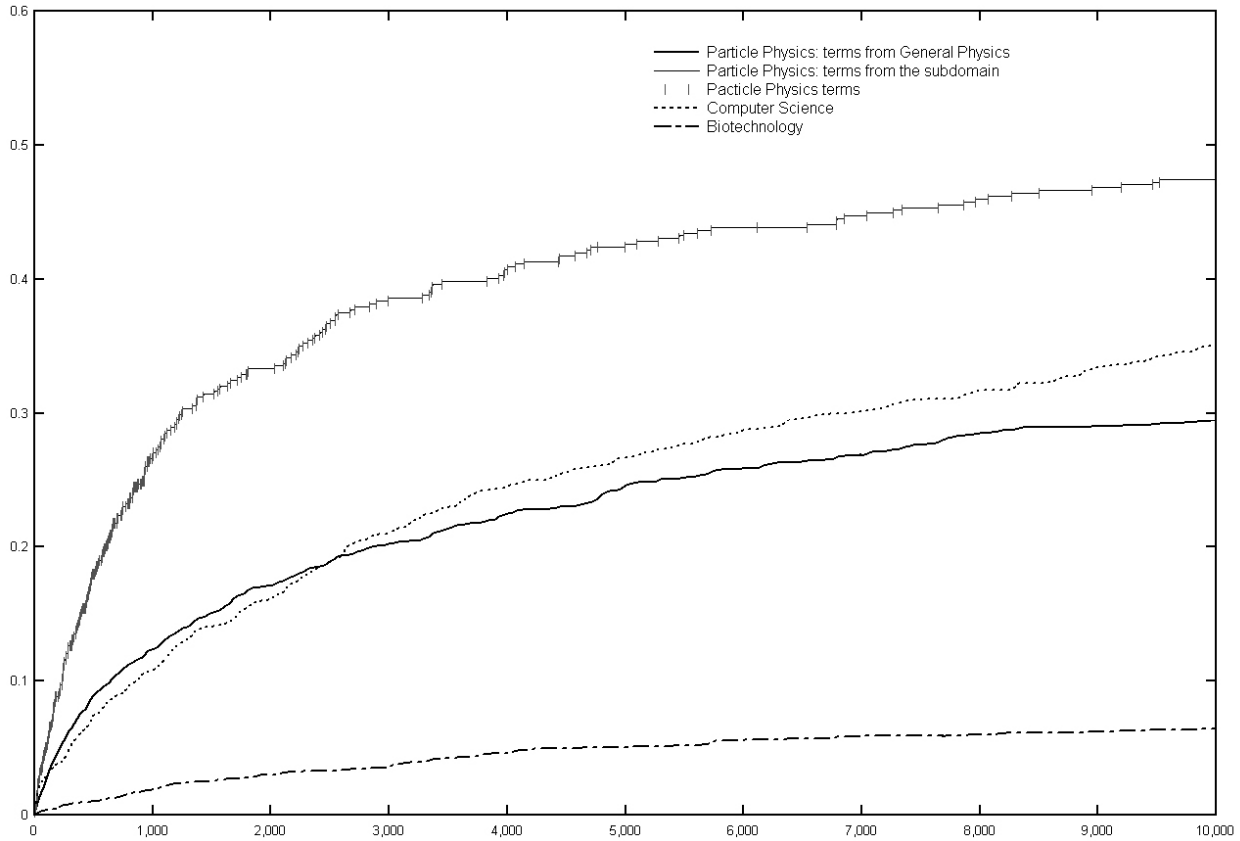

Figure 3. Domain recall of each term extraction

Atomic and Particle Physics		Computer Science		Biotechnology	
Physics	377	Computer Science	348	Biotechnology	146
Atomic and Particle Physics	109	General	112	Biology	99
Chemistry	56	Telecommunications	22	General	92
Others	86	Others	30	Others	95
Total	628	Total	512	Total	432

Table 2. Distribution of the new terms obtained by manual validation of the candidates extracted from the web corpora

4.2 Multilingual domain-comparable web corpora

This second experiment evaluates the improvement obtained in AzerHitz by enhancing the contexts of words with Internet searches. For this purpose, we have extracted bilingual terminology lists automatically with the AzerHitz tool from a Basque-English comparable corpus in the Computer Science domain automatically collected by Co3. Previous research done within AzerHitz is explained in (Saralegi et al., 2008a; Saralegi et al., 2008b). It must be noted that this research is currently ongoing and that the results presented here are preliminary.

4.2.1. Design

There are several reasons for choosing the Computer Science domain. On the one hand, terminology in this domain is constantly increasing. On the other, it is easy to obtain Computer Science documents from the Internet. Hence, terminology extraction from comparable corpora in this domain offers us a versatility that parallel corpora do not offer, because terminologically updated corpora can be easily obtained from the Internet.

For building the corpus, we provided a sample corpus consisting of 5,000 words for each language and launched the Co3 tool with them. Table 3 shows the size of the subcorpora collected.

In order to automatically extract terminology from comparable corpora, the AzerHitz system is based on cross-lingual context similarity. The underlying idea is that the same concept tends to appear with the same context words in both languages, in other words, it maintains many collocates. The algorithm used by AzerHitz is explained next.

AzerHitz starts the process by selecting those words which are meaningful (nouns, adjectives and verbs), henceforth content words. Each of them is then represented by a "context document". The context document of a word is composed by the content words appearing in the contexts of the word throughout the whole corpus. Those contexts are limited by a maximum distance to the word and by the punctuation marks. Context documents of all of the target language words are indexed by Lemur IR toolkit as a collection using the Indri retrieval model. To be able to compute the similarity between context documents of different languages, the documents in the source language are translated using a bilingual machine readable dictionary. We try to minimise the number of out-of-vocabulary words by using cognate detection, and ambiguity is tackled by using a first translation approach. To find the translation of a source word, its translated context document is sent as a query to the IR engine which returns a ranking of the most similar documents. In addition, a cognate detection step can be performed over the first ranked candidates. If a cognate is detected, the corresponding candidate will be promoted to the first position in the ranking. This can be useful in some domains in which the presence of loanwords is high.

The main problem of the context similarity paradigm is that the majority of the words do not have enough context information to be represented properly. To mitigate this problem, we propose that the Internet be used as a big comparable corpus. In this way, we expand the contexts of a word obtained from the initial corpus with new context words retrieved from web concordancers such as WebCorp (Renouf et al., 2007) or CorpEus (Leturia et al., 2007) to get a richer representation of the context. The contexts of both source and target language words are expanded. However, expanding all the contexts in the target language is computationally too expensive, and that is why, we only apply the expansion to the first translation candidates ranked by the IR engine.

The expansion may seem as a trivial task, but it has to address certain difficulties. We can not just expand with any context we get, because we may add noisy data. The contexts added must refer to the same sense of the word represented by the corpus contexts. In order to guarantee information with a good quality we use domain control techniques when retrieving contexts from the web concordancers.

4.2.2. Evaluation and results

We have evaluated the increase in performance obtained in AzerHitz by applying the enhancement of contexts using the web.

The evaluation of the system has been done over a set of 100 words, taken randomly from the corpus and which are not in the dictionary used. The words are translated manually in order to set up the reference for performing an automatic evaluation.

The following setups have been evaluated:

- Baseline: Only contexts obtained from the corpus.
- Baseline + Cognates: Cognate detection is performed on the first 20 ranked candidates.
- WaC: Web contexts expansion is performed.
- WaC + Cognates: Both context expansion and cognate detection among the first 20 ranking candidates are performed (in that order).

Table 4 shows the results of the experiments. Although these are only preliminary results, we can see that the expansion of the contexts using web data outperforms the results achieved when the context alone is retrieved from the corpus. These results show that the expansion helps to represent the word contexts better and, in turn, a better representation helps to compute more accurate context similarity and find correct translations.

We can also observe that adding the identification of cognates among the first 20 ranked candidates greatly improves the precision of the final ranking. The high presence of these kinds of translations accounts for this improvement.

Subcorpus	Words	Documents
Basque	2.6 M	2 K
English	2.6 M	1 K

Table 3. Computer science comparable corpus

Setup	top1	top5	top10	top15	top20
Baseline	0.32	0.54	0.60	0.62	0.66
WaC	0.36	0.56	0.68	0.72	0.72
Baseline + cognates	0.54	0.62	0.62	0.64	0.66
WaC + cognates	0.58	0.66	0.70	0.72	0.72

Table 4. Precision for top rankings

5. Conclusions

A common problem of less resourced languages is that the economic resources devoted to the development of NLP tools are also scarce. So the use of the Internet for building language resources such as corpora and, through them, other resources and NLP tools, is very attractive indeed. Nevertheless, the hypothesis that the Internet is a valuable and profitable source for developing language resources for less resourced languages must be tested in order to set up initiatives and projects with that objective.

It goes without saying that any attempt to build web corpora in a given language is conditioned by the size of the web in the target domains or genres. We consider that the results of the experiments that we have presented for Basque are encouraging. The size of the specialized web corpora we have compiled with our tools and the domain-precision achieved gives us some evidence that the Basque Internet, although not in any way comparable with the webs of major languages, can be large enough in specialized domains to be considered as a data source. Also, the fact that the use of web-derived contexts improves the results of terminology extraction from comparable corpora is further proof of this. This optimism should not hide the fact that, for the time being, some domains and genres may not have enough representation in the web.

In view of all this, the Elhuyar Foundation will go on working with the web as a source of corpora of many kinds and other types of language resources for Basque.

6. References

Alegria, I., Gurrutxaga, A., Saralegi, X., Ugartetxea, S. (2006). Elexbi, a basic tool for bilingual term extraction from Spanish-Basque parallel corpora. In *Proceedings of Euralex 2006*. Torino: Euralex, pp. 159--165.

Baroni, M., Bernardini, S. (2004). BootCaT: Bootstrapping corpora and terms from the web. In *Proceedings of LREC 2004*. Lisbon, Portugal: ELDA, pp. 1313--1316.

Baroni, M., Kilgarriff, A. (2006). Large linguistically-processed Web corpora for multiple languages. In *Proceedings of EACL 2006*. Trent, Italy: EACL, pp. 8--90.

Baroni, M., Ueyama, M. (2006). Building general- and special purpose corpora by Web crawling. In *Proceedings of the 13th NIJL International Symposium*. Tokyo, Japan: NIJL, pp. 31--40.

Baroni, M., Bernardini, S., Ferraresi, A., Zanchetta E. (2009). The WaCky Wide Web: A Collection of Very Large Linguistically Processed Web-Crawled Corpora. *Language Resources and Evaluation Journal*, 43(3), pp. 209--226.

Broder, A.Z. (2000). Identifying and filtering near-duplicate documents. In *Proceedings of Combinatorial Pattern Matching: 11th Annual Symposium*. Montreal, Canada: Springer, pp. 1--10.

Broder, A.Z. (1997). On the resemblance and containment of documents. In *Proceedings of Compression and Complexity of Sequences 1997*. Los Alamitos, CA: IEEE Computer Society, pp. 21--29.

Fletcher, W.H. (2004). Making the web more useful as a source for linguistic corpora. In U. Connor & T. Upton (Eds.), *Corpus Linguistics in North America 2002*. Amsterdam, The Netherlands: Rodopi.

Fletcher, W. H. (2006). Concordancing the Web: Promise and Problems, Tools and Techniques. In M. Hundt, N. Nesselhauf & C. Biewer (Eds.), *Corpus Linguistics and the Web*. Amsterdam, The Netherlands: Rodopi, pp. 25--46.

Gurrutxaga, A., Saralegi, X., Ugartetxea, S., Lizaso, P., Alegria, I., Urizar, R. (2004). A XML-based term extraction tool for Basque. In *Proc. of fourth*

international conference on Language Resources and Evaluation (LREC). Lisbon, Portugal: ELRA, pp. 1733--1736.

Gurrutxaga, A., Leturia, I., Pociello, E., Saralegi, X., San Vicente, I. (2009). Evaluation of an automatic process for specialized web corpora collection and term extraction for Basque. In *Proceedings of eLexicography in the 21st century*. Louvain-la-Neuve, Belgium: EURALEX & SIGLEX.

Leturia, I., Gurrutxaga, A., Alegria, I., Ezeiza, A. (2007). CorpEus, a 'web as corpus' tool designed for the agglutinative nature of Basque. In *Proceedings of Web as Corpus 3 workshop*. Louvain-la-Neuve, Belgium: ACL-SIGWAC, pp. 69--81.

Leturia, I., San Vicente, I., Saralegi, X., Lopez de Lacalle, M. (2008a). Collecting Basque specialized corpora from the web: language-specific performance tweaks and improving topic precision. In *Proceedings of the 4th Web as Corpus Workshop*. Marrakech, Morocco: ACL SIGWAC, pp. 40--46.

Leturia, I., Gurrutxaga, A., Areta, N., Pociello, E. (2008b). Analysis and performance of morphological query expansion and language-filtering words on Basque web searching. In *Proceedings of LREC 2008*. Marrakech, Morocco: ELRA.

Leturia, I., San Vicente, I., Saralegi. X. (2009). Search engine based approaches for collecting domain-specific Basque-English comparable corpora from the Internet. In *Proceedings of 5th International Web as Corpus Workshop (WAC5)*. Donostia, Spain: ACL-SIGWAC, pp. 53--61.

Renouf, A., Kehoe, A., Banerjee, J. (2007). WebCorp: an Integrated System for WebText Search. In M. Hundt, N. Nesselhauf & C. Biewer (Eds.), *Corpus Linguistics and the Web*. Amsterdam, The Netherlands: Rodopi, pp. 47--67.

Resnik, P. (1998). Parallel strands: A preliminary investigation into mining the web for bilingual text. In *Proceedings of the Third Conference of the Association for Machine Translation in the Americas on Machine Translation and the Information Soup*. Langhorne, USA: AMTA, pp.72--82,

Saralegi, X., Alegria, I. (2007). Similitud entre documentos multilingües de carácter científico-técnico en un entorno web. *Procesamiento del Lenguaje Natural*, 39, pp. 71--78.

Saralegi, X., Leturia, I. (2007). Kimatu, a tool for cleaning non-content text parts from HTML docs. In *Proceedings of the 3rd Web as Corpus workshop*. Louvain-la-Neuve, Belgium: Presses universitaires de Louvain, pp. 163--167.

Saralegi, X., San Vicente, I., Gurrutxaga, A. (2008a). Automatic Extraction of Bilingual Terms from Comparable Corpora in a Popular Science Domain. In *Proceedings of Building and using Comparable Corpora workshop*. Marrakech, Morocco: ELRA.

Saralegi, X., San Vicente, I., López de Lacalle, M. (2008b). Mining Term Translations from Domain Restricted Comparable Corpora. *Procesamiento del Lenguaje Natural*, 41, pp. 273--280.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1), pp. 1--47.

Sharoff, S. (2006). Creating General-Purpose Corpora Using Automated Search Engine Queries. In Marco Baroni & Silvia Bernardini (Eds.), *WaCky! Working Papers on the Web as Corpus*. Bologna, Italy: Gedit Edizioni, pp. 63--98.

Finite-State Spell-Checking with Weighted Language and Error Models—Building and Evaluating Spell-Checkers with Wikipedia as Corpus

Tommi A Pirinen, Krister Lindén

University of Helsinki—Department of Modern Languages
Unioninkatu 40 A—FI-00014 Helsingin yliopisto
{tommi.pirinen,krister.linden}@helsinki.fi

Abstract

In this paper we present simple methods for construction and evaluation of finite-state spell-checking tools using an existing finite-state lexical automaton, freely available finite-state tools and Internet corpora acquired from projects such as Wikipedia. As an example, we use a freely available open-source implementation of Finnish morphology, made with traditional finite-state morphology tools, and demonstrate rapid building of Northern Sámi and English spell checkers from tools and resources available from the Internet.

1. Introduction

Spell-checking is perhaps one of the oldest most researched application in the field of language technology, starting from the mid 20th century (Damerau, 1964). The task of spell-checking can be divided into two categories: isolated non-word errors and context-based real-word errors (Kukich, 1992). This paper concentrates on checking and correcting the first form, but the methods introduced are extendible to context-aware spell-checking.

To check whether a word is spelled correctly, a *language model* is needed. For this article, we consider a language model to be a one-tape finite-state automaton recognising valid word forms of a language. In many languages, this can be as simple as a word list compiled into a suffix tree automaton. However, for languages with productive morphological processes in compounding and derivation that are capable of creating infinite dictionaries, such as Finnish, a cyclic automaton is required. In order to suggest corrections, the correction algorithm must allow search from an infinite space. A nearest match search from a finite-state automaton is typically required (Oflazer, 1996). The reason we stress this limitation caused by morphologically complex languages is that often even recent methods for optimizing speed or accuracy suggest that we can rely on finite dictionaries or acyclic finite automata as language models.

To generate correctly spelled words from a misspelled word form, an *error model* is needed. The most traditional and common error model is the Levenshtein edit distance, attributed to Levenshtein (1966). In the edit distance algorithm, the misspelling is assumed to be a finite number of operations applied to characters of a string: deletion, insertion, change, or transposition[1]. The field of *approximate string matching* has been extensively studied since the mid 20th century, yielding efficient algorithms for simple string-to-string correction. For a good survey, see Kukich (1992). Research on approximate string matching has also provided different fuzzy search algorithms for finding the nearest match in a finite-state representation of dictionaries.

[1]Transposition is often attributed to an extended Levenshtein-Damerau edit distance given in (Damerau, 1964)

For the purpose of the article, we consider the error model to be any two-tape finite state automaton mapping any string of the error model alphabet to at least one string of the language model alphabet. As an actual implementation of Finnish spell-checking, we use a finite-state implementation of a traditional edit distance algorithm. In the literature, the edit distance model has usually been found to cover over 80 % of the misspellings at distance one (Damerau, 1964). Furthermore, as Finnish has a more or less phonemically motivated orthography, the existence of homophonic misspellings are virtually non-existent. In other words, our base assumption is that the greatest source of errors for Finnish spell-checking is the slip-of-the-finger style of typo, for which the edit distance is a good error model.

The statistical foundation for the language model and the error model in this article is similar to the one described by Norvig (2010), which also gives a good overview of the statistical basis for the spelling error correction problem along with a simple and usable python implementation.

For practical applications, the spell-checker typically needs to provide a small selection of the best matches for the user to choose from in a relatively short time span, which means that when defining corrections, it is also necessary to specify their likelihood in order to rank the correction suggestions. In this article, we show how to use a standard weighted finite-state framework to include probability estimates for both the language model and the error model. For the language model, we use simple unigram training with a Wikipedia corpus with the more common word forms to be suggested before the less common word forms. In the error model, we design the weights in the edit distance automaton so that suggestions with a greater Levenshtein-Damerau edit distance are suggested after those with fewer errors.

To evaluate the spell-checker even in the simple case of correcting non-word errors in isolation, a corpus of spelling mistakes with expected corrections is needed. Constructing such a corpus typically requires some amount of manual labour. In this paper, we evaluate the test results both against a manually collected misspelling corpus and against automatically misspelled texts. For a description of the error generation techniques, see Bigert (2005).

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 13–18

2. Goal of the paper

In this article, we demonstrate how to build and evaluate a spell-checking and correction functionality from an existing lexical automaton. We present a simple way to use an arbitrary string-to-string relation transducer as a misspelling model for the correction suggestion algorithm, and test it by implementing a finite-state form of the Levenshtein-Damerau edit distance relation. We also present a unigram training method to automatically rank spelling corrections, and evaluate the improvement our method brings over a correction algorithm using only the edit distance. The paper describes a work-in-progress version of a finite state spell-checking method with instructions for building the speller for various languages and from various resources. The language model in the article is an existing free open-source implementation of Finnish morphology[2] (Pirinen, 2008) compiled with HFST (Lindén et al., 2009)—a free, featurewise fully compliant implementation of the traditional Xerox-style LexC and TwolC tools[3].

One aim of this paper is to demonstrate the use of Wikipedia as a freely available open-source corpus[4]. The Wikipedia data is used in this experiment for training the lexical automaton with word form frequencies, as well as collecting a corpus of spelling errors with actual corrections.

The field of spell-checking is already a widely researched topic, cf. the surveys by Kukich (1992) and Schultz and Mihov (2002). This article demonstrates a generic way to use freely available resources for building finite-state spell-checkers. The purpose of using a basic finite-state algebra to create spell-checkers in this article is two-fold. Firstly, the amount of commonly known implementations of morphological language models under different finite-state frameworks suggest that a finite-state morphology is feasible as a language model for morphologically complex languages. Secondly, by demonstrating the building of an application for spell-checking with a freely available open-source weighted finite-state library, we hope to outline a generally useful approach to building open-source spell-checkers.

To demonstrate the feasibility of building a spell-checker from freely available resources, we use basic composition and n-best-path search with weighted finite-state automata, which allows us to use multiple arbitrary language and error models as permitted by the finite-state algebra. To the best of our knowledge, no previous research has used or documented this approach.

To further evaluate plausibility of rapid conversion from morphological or lexical automata to spell checkers we also sought and picked up a free open implementation of the Northern Sámi morphological analyzer[5] as well as a word list of English from (Norvig, 2010), and briefly tested them with the same methods and similar error model as for Finnish. While the main focus of the article is on the creation and evaluation of a Finnish finite-state spell-checker, we also show examples of building and evaluating spell-checkers for other languages.

3. Methods

The framework for implementing the spell-checking functionality in this article is the finite-state library HFST (Lindén et al., 2009). This requires that the underlying morphological description for spell-checking is compiled into a finite-state automaton. For our Finnish and Northern Sámi examples, we use a traditional linguistic description based on the Xerox LexC/TwolC formalism (Beesley and Karttunen, 2003) to create a lexical transducer that works as a morphological analyzer. As the morphological analyses are not used for the probability weight estimation in this article, the analysis level is simply discarded to get a one-tape automaton serving as a language model. However, the word list of English is directly compiled into a one tape suffix tree automaton.

As mentioned, the original language model can be as simple as a list of words compiled into a suffix tree automaton or as elaborate as a full-fledged morphological description in a finite-state programming language, such as Xerox LexC and TwolC[6]. The words that are found in the transducer are considered correct. The rest are considered misspelled.

It has previously been demonstrated how to add weights to a cyclic finite-state morphology using information on base-form frequencies. The technique is further described by Lindén and Pirinen (2009). In the current article, the word form counts are based on data from the Wikipedia. The training is in principle a matter of collecting the corpus strings and their frequencies and composing them with the finite-state lexical data. Deviating from the article by Lindén and Pirinen (2009), we only count full word forms. No provisions for compounding of word forms based on the training data are made, i.e. the training data is composed with the lexical model. This gives us an acyclic lexicon with the frequency data for correctly spelled words.

The actual implementation goes as follows. Clean up the Wikipedia dump to extract the article content from XML and Wikipedia mark-up by removing the mark-up and the contents of mark-up that does not constitute running text, leaving only the article content untouched. The tokenization is done by splitting text at white space characters and separating word final punctuation. Next we use the spell-checking automaton to acquire the correctly spelled word forms from the corpora, and count their frequencies. The formula for converting the frequencies f of a token in the corpus to a weight in the finite-state lexical transducer is $W_t = -\log \frac{f_t}{CS}$, where CS is the corpus size in tokens. The resulting strings with weights can then be compiled into paths of a weighted automaton, i.e. into an acyclic tree automaton with log probability weights in the final states of the word forms. The original language model is then weighted by setting all word forms not found in the corpus

[2] `http://home.gna.org/omorfi`

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

[4] Database dumps available at `http://download.wikimedia.org`

[5] `http://divvun.no`

[6] It is also possible to convert aspell and hunspell style descriptions into transducers. Preliminary scripts exist in `http://hfst.sf.net/`.

to a weight greater than the word with frequency of one, e.g. $W_{max} = -\log \frac{1}{CS+1}$. The simplest way to achieve this is to compose the $\Sigma^\star$ automaton with final weight W_{max} with the unweighted cyclic language model. Finally, we take the union of the cyclic model and the acyclic model. The word forms seen in the corpus will now have two weights, but the lexicon can be pruned to retain only the most likely reading for each string.

For example in the Finnish Wikipedia there were 17,479,297 running tokens[7], and the most popular of these is 'ja' *and* with 577,081 tokens, so in this language model the $W_{ja} = -\log \frac{577081}{17479297} \approx 4.44$. The training material is summarized in the Table 1. The token count is the total number of tokens after preprocessing and tokenization. The unique strings is the number of unique tokens that belonged to the language model, i.e. the size of actual training data, after uniqification and discarding potential misspellings and other strings not recognized by the language model. For this reason the English training model is rather small, despite the relative size of the corpus, since the finite language model only covered a very small portion of the unique tokens.

Language	Finnish	Northern Sámi	English
Token count	17,479,297	258,366	2,110,728,338
Unique language strings	968,996	44,976	34,920
Download size	956 MiB	8.7 MiB	5.6 GiB
Version used	2009-11-17	2010-02-22	2010-01-30

Table 1: Token counts for wikipedia based training material

For finding corrections using the finite-state methodology, multiple approaches with specialized algorithms have been suggested, e.g. (Oflazer, 1996; Schulz and Mihov, 2002; Huldén, 2009). In this article, we use a regular weighted finite-state transducer to represent a mapping of misspellings to correct forms. This allows us to use any weighted finite-state library that implements composition.

One of the simplest forms of mapping misspellings to correct strings is the edit distance algorithm usually attributed to Levenshtein (1966) and furthermore in the case of spell-checking to Damerau (1964). A finite-state automaton representation is given by e.g. Schulz and Mihov (2002). A transducer that corrects strings can be any arbitrary string-to-string mapping automaton, and can be weighted. In this article, we build an edit distance mapping transducer allowing two edits.

Since the error model can also be weighted, we use the weight W_{max} as the edit weight, which is greater than any of the weights given by the language model. As a consequence, our weighted edit distance will function like the traditional edit distance algorithm when generating the corrections for a language model, i.e. any correct string with edit distance one is considered to be a better correction than a misspelling with edit distance two. For example assuming misspelling 'jq' for 'ja', the error model would find 'ja' at an edit distance of W_{max}, but also e.g. 'jo' *already* and so on. In this case the frequency data obtained from Wikipedia will give us the popularity order of 'ja' > 'jo'. A fraction of the weighted edit distance two transducer is given in Figure 1. The transducer in the figure displays a full edit distance two transducer for a language with two symbols in the alphabet; an edit distance transducer for a full alphabet is simply a union of such transducers for each pair of symbols in the language[8].

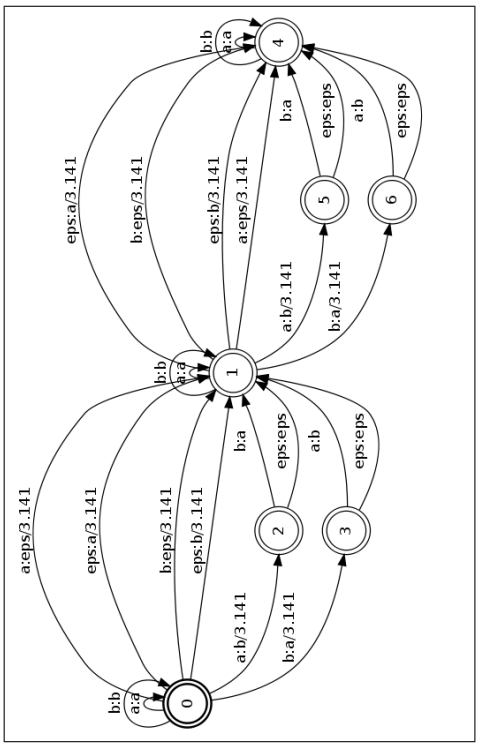

Figure 1: Edit distance transducer of alphabet a, b length two and weight π.

To get a ranked set of spelling correction suggestions, we simply compile the misspelled word into a path automaton T_{word}. The path automaton is composed with the correction relation T_E—in this case the weighted edit distance two transducer—to get an automaton that contains all the possible spelling corrections $T_{sug} = T_{word} \circ T_E$. We then compose the resulting automaton with the original weighted lexical data T_L to find the string corrections that are real words of the language model $T_f = T_{sug} \circ T_L$. The resulting transducer now contains a union of words with the combined weight of the frequency of the word form and the weight of the edit distance. From this transducer, a ranked list of spelling suggestions is extracted by a standard n-best-path algorithm listing unique suggestions.

4. Test Data Sets

For the Finnish test material, we use two types of samples extracted from Wikipedia. First, we use a hand-picked selection of 761 misspelled strings found by browsing the

[7]We used relatively naive preprocessing and tokenization, splitting at spaces and filtering html and Wikipedia markup

[8]For the source code of the Finnish edit distance transducer in the HFST framework, see `http://svn.gna.org/viewcvs/omorfi/trunk/src/suggestion/edit-distance-2.text`

strings that the speller rejected. These strings were manually corrected using a native reader's best judgement from reading the misspelled word in context to achieve a gold standard for evaluation.

Another larger set of approximately 10,000 evaluation strings was created by using the strings from the same Wikipedia corpus, and automatically introducing spelling errors similar to the approach described by Bigert et al. (2003), using isolated word Damerau-Levenshtein type errors with a probability of approximately 0.33 % per character. This error model could also be considered an error model applied in reverse compared to the error model used when correcting misspelled strings. As there is nothing limiting the number of errors generated per word except the word length, this error model may introduce words with an edit distance greater than two.

As the Northern Sámi gold standard, we used the test suite included in the svn distribution[9]. It seems to contain a set of common typos.

As the English gold standard for evaluation, we use the Birkbeck spelling error corpus referred to in (Norvig, 2010). The corpus has restricted free licensing, but the restrictions prohibit its use as training material in a free open source project.

5. Evaluation

To evaluate the correction algorithm, we use the two data sets introduced in the previous section. However, we use a slightly different error model to automatically correct misspellings than we use for generating them, i.e. some errors exceeding the edit distance of two are unfixable by the error model we use for correction.

The evaluation of the correction suggestion quality is given in Tables 2 and 3. The Table 2 contains precision values for the spelling errors from real texts, and Table 3 for the automatically introduced spelling errors. The precision is measured by ranked suggestions. In the tables, we give the results separately for ranks 1—4, and for the remaining lower ranks. The lower ranks ranged from 5—440 where the number of total suggestions ranged from 1—600. In the last column, we have the cases where a correctly written word could not be found with the proposed suggestion algorithm. The tables contain both the results for the weighted edit distance relation, and for a combination of the weighted edit distance relation and the word form frequency data from Wikipedia.[10]

As a first impression we note that mere Wikipedia training does improve the results in all cases; the number of suggestions in first position rises in all test sets and languages. This suggests that more mistakes are made in common words than in rare ones, since the low ranking word counts did not increase as a result of Wikipedia training.

In the Finnish tests, haplological cases like 'kokonaismalmivaroista' *from total ore resources* spelled as 'konaismalmivaroista' came in at the bottom of the list for both methods, because the correct word form is probably nonexistent in the training corpus, and the multi-part productive compound with an ambiguous segmentation produces lots of nearer matches at edit distance one. A more elaborate error model considering haplology as a misspelling with a weight equal or less than a single traditional edit distance would of course improve the suggestion quality in this case. The number of words getting no ranks is common to both methods. They indicate the spelling errors for which the correct form was neither among the ones covered by the error model for edit distance two nor in the language model. A substantial number are cases which were not considered in the error model, e.g. a missing space causing run-on words ('ensisijassa' instead of 'ensi sijassa' *in the first place*). A good number also comes from spoken or informal language forms for very common words, which tend to deviate more than edit distance two ('esmeks' instead of 'esimerkiksi' *for example*), with a few more due to missing forms in the language model. E.g. 'bakterisidin' is one edit from 'bakterisidina' *as bactericide*, but the correction is not made because the word does not exist in the language model. These error types are correctable by adding words to the lexicon, i.e. the language model, e.g. using special-purpose dictionaries, such as spoken language or medical dictionaries. Finally there is a handful of errors that seem legitimate spelling mistakes of more than two edits ('assosioitten' instead of 'assosiaatioiden'). For these cases, a different error model than the basic edit distance might be necessary.

Material	Rank 1	2	3	4	Lower	No rank	Total
Weighted edit distance 2							
Finnish	371	118	65	33	103	84	761
	49 %	16 %	9 %	4 %	14 %	11 %	100 %
Northern	2221	697	430	286	2743	2732	9115
Sámi	24 %	8 %	5 %	3 %	30 %	30 %	100 %
English	8739	2695	1504	940	3491	17738	35106
	25 %	8 %	4 %	3 %	10 %	51 %	100 %
Wikipedia word form frequencies and edit distance 2							
Finnish	451	105	50	22	62	84	761
	59 %	14 %	7 %	3 %	8 %	11 %	100 %
Northern	2421	745	427	266	2518	2732	9115
Sámi	27 %	8 %	5 %	3 %	28 %	30 %	100 %
English	9174	2946	1489	858	2902	17738	35106
	26 %	8 %	4 %	2 %	8 %	51 %	100 %

Table 2: Precision of suggestion algorithms with real spelling errors

Material	Rank 1	2	3	4	Lower	No rank	Total
Weighted edit distance 2							
Finnish	4321	1125	565	351	1781	1635	10076
	43 %	11 %	6 %	3 %	18 %	16 %	100 %
Northern	1269	257	136	80	528	7730	10000
Sámi	13 %	3 %	1 %	1 %	5 %	77 %	100 %
English	4425	938	337	290	1353	2657	10000
	44 %	10 %	3 %	3 %	14 %	27 %	100 %
Wikipedia word form frequencies and edit distance 2							
Finnish	4885	1128	488	305	1407	1635	10076
	49 %	11 %	5 %	3 %	14 %	16 %	100 %
Northern	1726	253	76	29	186	7730	10000
Sámi	17 %	3 %	1 %	1 %	2 %	77 %	100 %
English	5584	795	307	196	461	2657	10000
	56 %	8 %	3 %	2 %	5 %	27 %	100 %

Table 3: Precision of suggestion algorithms with generated spelling errors

[9] https://victorio.uit.no/langtech/trunk/gt/sme/src/typos.txt

[10] For full tables and test logs, see http://home.gna.org/omorfi/testlogs.

For the Northern Sámi spelling error corpus we note that a large amount of errors is not covered by the error model. This means that the error model is not sufficient for Northern Sámi spell checking as we can see a number of errors with edit distance greater than 2, e.g. 'sáddejun' instead of 'sáddejuvvon'.

Comparing our English test results with previous research using implementations of the same language and error model, we reiterate that a great number of words are out of reach by an error model of mere edit distance 2. Some of the test words are even word usage errors, such as 'gone' in stead of 'went', but unfortunately they were intermixed with the other spelling error material and we did not have time to remove them from the test corpus. The rest of the spelling errors beyond edit distance 2 are mostly caused by English orthography being relatively distant from pronunciation, such as 'negoshayshauns' in stead of 'negotiations', which usually are corrected with very different error models such as soundex and other phonetic keys as demonstrated by e.g. Mitton (2009). The results of the evaluation of the correction suggestions show a similar tendency as the one found in the original article by Norvig (2010).

The impact on performance when using non-optimized methods to check spelling and get suggestion lists was not thoroughly measured, but to give an impression of the general applicability of the methods, we note that for the Finnish material of generated misspellings, the speed of spell-checking was 3.18 seconds for 10,000 words or approx. 3,000 words per second, and the speed of generating suggestion lists, i.e. all possible corrections for 10,000 misspelled words took 10,493 seconds, i.e. on the average it took approx. 1 second to generate all the suggestions for each misspelled word, when measured with the GNU `time` utility and the `hfst-omor-evaluate` program from the HFST toolkit, which batch processes spell-checking tasks on tokenized input and evaluates precision and recall against a correction corpus. The space requirements for the Finnish spell checking automata are 9.2 MiB for the Finnish morphology and 378 KiB for the Finnish edit distance two automaton with an alphabet size of 72. As a comparison, the English language model obtained from the word list is only 3.2 MiB in size, and correspondingly the error model 273 KiB with an alphabet size of 54.

6. Discussion

The obvious and popular development is to extend the language model to support n-gram dictionaries with $n > 1$, which has been shown to be a successful technique for English e.g. by Mays (1991). The extension using the same framework is not altogether trivial for a language like Finnish, where the number of forms and unique strings are considerable giving most n-grams a very low-frequency.

Even if the speed and resource use for spell-checking and correction was found to be reasonable, it may still be interesting to optimize for speed as has been shown in the literature (Oflazer, 1996; Schulz and Mihov, 2002; Huldén, 2009). At least the last of these is readily available as an open-source finite-state implementation in foma[11], and is expandable for at least a non-homogeneous non-unit-weight edit distance with context restrictions. However, it does not yet cater to general weighted language models.

Other manual extensions to the spelling error model should also be tested. Our method ensures that arbitrary weighted relations can be used. Especially the use of a non-homogeneous non-unit-length edit distance can easily be achieved. Since it has been successfully used in e.g. hunspell[12], it should be further evaluated. Other obvious and common improvements to the edit distance model is to scale the weights of the edit distance by the physical distance of the keys on a QWERTY keyboard.

The acquisition of an error model or probabilities of errors in the current model is also possible (Brill and Moore, 2000), but this requires the availability of an error corpus containing a large (representative) set of spelling errors and their corrections, which usually are not available nor easy to create. One possible solution for this may of course be to implement an adaptive error model that modifies the probabilities of the errors for each correction made by user.

The methods were only evaluated on languages with substantial resources, but the use of freely available language resources and toolkits makes the proposed methods for creating spell-checkers interesting for less resourced languages as well, since most written languages already have text corpora, word lists and inflectional descriptions.

The next step is to improve the free open-source Voikko[13] spell-checking library with the HFST transducer-based spell-checking library. Voikko has been successfully used in open-source software such as OpenOffice.org, Mozilla Firefox, and the Gnome desktop (in the enchant spell-checking library).

7. Conclusions

In this article we have demonstrated an approach for creating spell-checkers from various language models—ranging from simple word lists to complex full-fledged implementations of morphology—built into a finite-state automata. We also demonstrated a simple approach to training the models using word frequency data extracted from Wikipedia. Further, we have presented a construction of a simple edit distance error model in the form of a weighted finite-state transducer, and proven usability of this basic finite state approach by evaluating the resulting spell-checkers against both manually collected smaller and automatically created larger error corpora. Given the amount of finite-state implementations of morphological language models it seems reasonable to expect that general finite-state methods and language models can support spell-checking for a large array of languages. The methods may be especially useful for less resourced languages, since most written languages already have text corpora, word lists and inflectional descriptions.

The fact that arbitrary weighted two-tape automata may be used for implementing error models suggests that it is relatively easy to implement different error models with available open-source finite-state toolkits. We also showed that

[11] `http://foma.sf.net`

[12] `http://hunspell.sf.net`

[13] `http://voikko.sf.net/`

combining the basic edit distance error model with a simple unigram frequency model already improves the quality of the error corrections. We also note that even using a basic finite-state transducer algebra from a freely available finite-state toolkit and no specialized algorithms, the speed and memory requirements of the spell-checking seems sufficient for typical interactive usage.

8. Acknowledgements

We thank our colleagues in the HFST research team and the anonymous reviewers for valuable suggestions regarding the article.

9. References

Kenneth R Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI publications.

Johnny Bigert, Linus Ericson, and Antoine Solis. 2003. Autoeval and missplel: Two generic tools for automatic evaluation. In *Nodalida 2003*, Reykjavik, Iceland.

Johnny Bigert. 2005. *Automatic and unsupervised methods in natural language processing*. Ph.D. thesis, Royal institute of technolog (KTH).

Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293, Morristown, NJ, USA. Association for Computational Linguistics.

F J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM*, (7).

Måns Huldén. 2009. Fast approximate string matching with finite automata. *Procesamiento del Lenguaje Natural*, 43:57–64.

Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Comput. Surv.*, 24(4):377–439.

V. I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics—Doklady 10, 707710. Translated from Doklady Akademii Nauk SSSR*, pages 845–848.

Krister Lindén and Tommi Pirinen. 2009. Weighting finite-state morphological analyzers using hfst tools. In Bruce Watson, Derrick Courie, Loek Cleophas, and Pierre Rautenbach, editors, *FSMNLP 2009*, 13 July.

Krister Lindén, Miikka Silfverberg, and Tommi Pirinen. 2009. Hfst tools for morphology—an efficient open-source package for construction of morphological analyzers. In Cerstin Mahlow and Michael Piotrowski, editors, *sfcm 2009*, volume 41 of *Lecture Notes in Computer Science*, pages 28—47. Springer.

Eric Mays, Fred J. Damerau, and Robert L. Mercer. 1991. Context based spelling correction. *Inf. Process. Manage.*, 27(5):517–522.

Roger Mitton. 2009. Ordering the suggestions of a spellchecker without using context*. *Nat. Lang. Eng.*, 15(2):173–192.

Peter Norvig. 2010. How to write a spelling corrector. Web Page, Visited February 28th 2010, Available `http://norvig.com/spell-correct.html`.

Kemal Oflazer. 1996. Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. *Comput. Linguist.*, 22(1):73–89.

Tommi Pirinen. 2008. Suomen kielen äärellistilainen automaattinen morfologinen analyysi avoimen lähdekoodin menetelmin. Master's thesis, Helsingin yliopisto.

Klaus Schulz and Stoyan Mihov. 2002. Fast string correction with levenshtein-automata. *International Journal of Document Analysis and Recognition*, 5:67–85.

Finite-State Morphology for Iñupiaq

Aric Bills[1], Lori S. Levin[2], Lawrence D. Kaplan[3], Edna Ahgeak MacLean[4]

[1,3,4]Alaska Native Language Center, [2]Language Technologies Institute
[1,3,4]University of Alaska Fairbanks, [2]Carnegie Mellon University
[1]aric.bills@gmail.com, [2]lsl@cs.cmu.edu, [3]ldkaplan@alaska.edu, [4]edna.maclean@gmail.com

Abstract

This paper describes ongoing work to develop a finite-state computational morphology of North Slope Iñupiaq, an indigenous North American language with exceptionally productive derivational morphology, complex inflection, and considerable morphologically conditioned phonological phenomena. The language-independent Xerox Finite-State Tools serve as the underlying engine for our lexical transducer and ultimately make this project possible, but a language-specific abstraction layer implemented above the *lexc* finite-state lexicon definition language has made it possible to develop the morphology more quickly and in a more natural way, which we believe will lead to improved maintainability and scalability.

1. Introduction

This paper describes ongoing work to develop a computational morphology of North Slope Iñupiaq, a language with exceptionally productive derivational morphology, complex inflection, and considerable morphologically conditioned phonological phenomena. Our work expands on earlier work on Iñupiaq computational morphology by Per Langgård and Trond Trosterud. In this paper we wish specifically to draw attention to ways in which a format tailored to the nature of Iñupiaq morphophonology has made this process simpler and more natural, and therefore, we hope, easier to maintain and expand.

2. Iñupiaq

Iñupiaq is an Eskimo-Aleut language and the westernmost member of the Inuit dialect continuum, which extends from northern Alaska to Greenland. With approximately 2144 living speakers from a community of 15,700 (Krauss, 2007, p. 409), Iñupiaq is considered endangered. The present work concerns the North Slope dialect, which is spoken in the Alaskan villages of Kivalina, Point Hope, Point Lay, Wainwright, Atqasuk, Barrow, Nuiqsut, Kaktovik, and Anaktuvuk Pass (MacLean, 1986a, p. x). North Slope Iñupiaq exhibits both lexical and phonological differences from other dialects. Among other differences, it is the only dialect which has all of the following: a series of palatal consonants; surface clusters of vowels of different qualities (that is, vowel clusters in addition to phonetically long vowels); and no consonant clusters that do not agree in terms of voicing (i.e., both consonants are voiced or neither are voiced) and continuancy (i.e., both are obstruents or neither are obstruents).

North Slope Iñupiaq, like all Eskimo languages, is highly polysynthetic and has a elaborate inflectional system. Its phonology is generally more conservative and more complex than Canadian and Greenlandic Inuit dialects. Most suffixes trigger morphophonological alternation at morpheme boundaries; additionally, "a great many Inupiaq suffixes exhibit allomorphy for which no one proposes a synchronic phonological account" (Kaplan, 1981, p. 232).

Iñupiaq examples in this paper will be given in the standard orthography.[1]

2.1. Iñupiaq morphology

Iñupiaq grammatical categories include nouns, verbs, demonstratives, personal pronouns, and particles. Of most interest morphologically are nouns and verbs, which allow both extensive derivational morphology and complex inflection; demonstratives also allow rich inflection as well as limited derivational morphology.

The basic structure of an Iñupiaq noun, verb, or demonstrative is *base + zero or more postbases (bound derivational suffixes) + inflectional ending + zero or more enclitics*. The process of postbase attachment may be considered a recursive, stem-deriving process; a stem may be defined as either a base or a stem plus a postbase. The main morphotactic constraint on Iñupiaq stems (aside from semantic considerations, which will not be taken into account here) is that postbases and inflectional endings must match the category of the stem to which they attach; in other words, nominal suffixes attach to nominal stems, verbal suffixes attach to verbal stems, etc. Postbases which attach to a particular category may derive stems of a different category; for example, postbase *-qaq-* 'have' attaches to nominal stems and derives verbal ones, as in *qamutiqaqtuŋa* 'I have a car', from *qamun* 'car' + *-qaq-* 'have' + *-tuŋa* (indicative present 1st person singular).

Nouns are inflected for case and grammatical number (singular, dual, or plural) and for grammatical person and number of their possessor, if any. Verbs are inflected for mood and grammatical person and number of their subject, as well as grammatical person and number of any definite direct object. Demonstratives may be inflected as pronouns, in which case they are inflected for case and the grammatical number of their antecedent, or as adverbs, in which case they are inflected for case only. Verbal inflections are explicitly transitive or intransitive, so an additional, long-distance morphotactic constraint is that verbal inflection be compatible with stem valence. Additionally, some noun stems are restricted

[1]For a guide to pronunciation, see `http://www.alaskool.org/Language/inupiaqhb/Inupiaq_Handbook.htm`.

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 19–26

as to the grammatical numbers for which they may be inflected; for example, *kamikłuuk* 'pants' cannot bear singular inflection.

Enclitics are words which are syntactically distinct from other words but phonologically (and orthographically) bound to the previous word. A distinction can be made between "reduced forms"—enclitics which have full-word counterparts—and "true" enclitics, which have no such counterparts. An example of a reduced form is *=una* 'this' as in *sunauna* 'what is this?'; *una* may also occur as an independent word, as in *una qimmiq siñiktuq* 'this dog is sleeping'. In contrast, a "true" enclitic such as *=lu* 'and' as in *tuttulu qimmiġlu* 'the caribou and the dog' cannot be separated from the words to which it attaches; **tuttu lu qimmiq lu* is ungrammatical.

2.2. Iñupiaq morphophonology

Different suffixes in Iñupiaq trigger different morphophonological alternations at their left boundaries, and the pattern of alternations a suffix will trigger is not entirely predictable from the phonetic form of the suffix. For example, postbase *-saġataq-* 'for a long time' attaches directly to stems without deleting anything; postbase *-sugruk-* 'a lot' deletes stem-final consonants; and postbase *-siññaq-* 'only' deletes stem-final /t/ but not /k/ or /q/. Other patterns include deleting penultimate /i/, deleting stem-final back consonants, or deleting stem-final syllables. Some suffixes trigger gemination of the onset of the preceding syllable. Edna MacLean, in her Iñupiaq pedagogical materials (1986a; 1986b; unpublished), indicates the morphophonological attachment pattern of a suffix with one of eight symbols; for example, '-' indicates that a stem-final consonant is deleted, while '+' indicates that no stem-final segment is deleted.

Many suffixes also have phonologically conditioned allomorphs, and different morphemes are sensitive to different environments. For example, postbase *-tiq-* 'quickly or abrubtly' becomes *-liq-* when preceded by a vowel (suffix *-tuksrau-* 'must' becomes *-ruksrau-* in the same environment); postbase *-suk-* 'want to' becomes *-uk-* when preceded by a back consonant; absolutive plural marker *-t* becomes *-it* after a /k/ or certain lexically conditioned instances of /q/, and before other instances of /q/ it optionally triggers gemination.

3. Finite-State Morphology

Finite-state transducers are directed graphs representing rational relations between sets of strings, and are an elegant way to model morphology computationally (Beesley, 2004b, p. 3). They are compact, fast, and inherently bidirectional (meaning that a single morphological transducer can be used equally well for generation as for analysis). A lexical transducer is a transducer that maps surface forms of words onto abstract, morphologically decomposed "underlying" forms[2], by combining a lexicon (consisting of underlying forms of words) together with a set of phonological (or, more accurately, graphemic) rules (Karttunen et al., 1992).

The Xerox Finite State Toolkit (XFST) is probably the most widely used software for creating lexical transducers (Kornai, 1999, p. 4). It provides two languages, *xfst* and *lexc*, designed to be used in tandem. *xfst* is a language with a rich calculus for specifying regular expressions, most commonly used to model phonological rewrite rules. *lexc* is a right-recursive phrase-structure grammar (Beesley and Karttunen, 2003, p. 203) for specifying lexicons in an underlying form via morpheme concatenation. Because a grammar based on concatenation alone cannot easily restrict the co-occurrence of non-adjacent morphemes within a word, *lexc* also provides a mechanism called a "flag diacritic." Flag diacritics set or query memory registers and can be associated with specific morphemes. Any word containing two morphemes with incompatible flag diacritics is effectively filtered out of the lexicon (Beesley and Karttunen, 2003, pp. 339–373).

4. Langgård and Trosterud's transducer

Per Langgård of Okaasileriffik (the Greenland Language Secretariat) and Trond Trosterud of the University of Tromsø have developed a proof-of-concept Iñupiaq transducer,[3] and generously furnished us with the XFST source code at the beginning of our project. We referred to this code frequently in the early stages of development of our transducer and incorporated several key features from it, including the use of the symbol '>' as a morpheme boundary marker and the definition of sets of characters (vowels, plosives, voiced fricatives) to be used for convenience in morphographemic rules. Two additional techniques adopted from this transducer are especially pertinent to the discussion at hand: first, flag diacritics are used to ensure that verb inflections reflect the valence of their stem (in other words, that intransitive-only verb stems not be inflected with transitive endings); and second, extensive use is made of "rule triggers"—special tags attached to morphemes to indicate that the morpheme conditions a particular alternation (see Uí Dhonnchadha, 2003, p. 46).

5. Implementation

5.1. Morphographemics

Productive phonological processes are modeled using morphographemic rewrite rules written in *xfst*. Langgård and Trosterud's rules were rewritten to correspond more closely to Edna MacLean's analysis of Iñupiaq phonology, to facilitate the inclusion of lexical material from her work.[4] In particular, each of MacLean's suffix combination patterns (see Section 2.2.) was implemented as a cascade of rules sensitive to the presence of a specific rule trigger, which is deleted as the last step in the cascade. Rules also exist for the

[2]For example, a lexical transducer for English might map the surface form *hidden* onto the underlying form *hide+PastParticiple*; one for Iñupiaq might map surface form *tautuŋniaġiga* 'she/he will see me' onto underlying form *tautuk+niaq+IndicativePresent+3Sg+1SgObj*.

[3]`http://giellatekno.uit.no/ipk.html`

[4]Langgård and Trosterud's transducer is based on the Greenlandic tradition of Eskimo analysis, with which the first author was unfamiliar and which would have made it more difficult to use MacLean's work; rewriting the rules also allowed the first author to come to grips with *xfst*. The rewriting was not due to any inaccuracy in Langgård and Trosterud's code.

formation of the absolutive dual stem (which serves as the basis for several other dual forms), demonstrative-specific alternations, gemination, palatalization, assimilation, and the conversion between the transducer-internal format and standard Iñupiaq orthography.

5.2. Lexicon

The lexicon is defined in a series of text files whose format was designed to optimize data entry; the contents of these files are converted first to XML, then to *lexc* format. The underlying data model of the source files is compatible with the phrase-structure grammar of *lexc*, but the format of the files themselves is quite different. In *lexc*, files are structured as lists of word formatives called LEXICONs. Each member of a LEXICON can specify a "continuation class"—another LEXICON whose members it will accept as suffixes. Members of a LEXICON may be empty strings, in which case the members of the specified continuation class essentially become members of the empty string's LEXICON.

Beesley (2004b; 2004a; 2003) advises against creating *lexc* lexicons from scratch, calling them a "dead-end" (Beesley, 2004a, p. 2) because the format is specific to the Xerox Finite-State Tools and the data are too sparse to be very useful to other applications. Instead, Beesley recommends creating lexical resources in XML, which can be used to represent data sets of arbitrary complexity. However, writing XML by hand is cumbersome and error-prone. As a compromise, we have created a set of lightweight, text-based formats designed to allow us to enter essential information about each morpheme in a quick, natural way, and a Tcl script to convert this information into XML; from that format it is then converted to *lexc* format.[5] At present, the XML representation of the lexical data contains very little information other than what is needed to build a lexical transducer, but others who require Iñupiaq lexical data should find it reasonably easy to use and expand upon. Because the XML itself is rather unremarkable, no more will be said about it in this paper.

Separate source files exist for bases, postbases, inflectional suffixes, and enclitics. Each file begins with a metadata section where shorthand morpheme category codes are defined and associated with LEXICON names, continuation classes, and flag diacritics.[6] With the exception of the inflectional suffix file, morphemes are then listed in the order in which they appear in the dictionary or grammar book, without any special grouping by category. In the stem file, each entry consists of an orthographic form followed by a category code; one may optionally specify one or more of the following: separate lexical (underlying) and surface forms (separated by a colon), a comment (beginning with an exclamation point), an English-language gloss (beginnning with a hash mark), and a reduced form of the stem (beginning with a tilde). Figure 1 presents sample stem entries from a variety of categories: *aaġlu* 'killer whale' is a noun stem; *aasii* 'and [then]' is a conjunction; *ikayuq-* 'help' is a verb stem which may be either intransitive (e.g., *ikayuqtuq* 'he/she is helping') or transitive (e.g. *ikayuġaaŋa* 'he/she is helping me'); *iraqtu-* 'be wide' is an intransitive-only verb stem (e.g. *iraqturuq* 'it is wide'); *ñiaq* 'don't do that!' is an interjection; and *suna* 'what' is an interrogative pronoun. The entry for *suna* shows how one specifies separate lexical and surface forms; surface form *suna* is specified here as a special absolutive singular form of the stem *su-* ('what').

```
aaġlu n # killer whale
aasii conj ~asii # and [then]
ikayuq it # to help [someone]
iraqtu i # to be wide
ñiaq interj # don't do that
su>+Pro+Abs+Sg:suna pro # what
     ! interrogative pronoun
```

Figure 1: Example stem entries.

In the postbase and enclitic files, each entry includes an orthographic form, a membership category code (denoting the class to which the morpheme belongs; these include 'n' [noun], 'i' [intransitive verb], 't' [transitive verb], and 'it' [ambitransitive verb]), and a continuation category code (specifying the morpheme's continuation class; in addition to the membership category codes, code 'same' marks verb-attaching postbases which derive verbs of the same valence, whatever that may be). Entries may optionally specify English-language glosses, comments, and separate lexical and surface forms. Example postbase definitions are given in Figure 2; sample words containing these postbases are given in examples 1–6, which will be discussed below. All examples are from the personal files of Edna MacLean unless otherwise noted.

```
+ġruiññaq n n # merely, only, just a
-qaq n i # have
+-qasIq i t # to V at the same time with Obj
{?C -+sima ?V +ma} it same #it is now known that
+[s]uk it same # want to
+t//liq it same # quickly, abruptly
```

Figure 2: Example postbase entries.

(1) *iqalugruiññaq*
iqaluk-ġruiññaq
fish-merely
'merely a fish'

[5] An anonymous reviewer points out that there are XML editors which allow data to be entered as simply as with the non-XML formats described here.While our format allowed lexical data to be entered easily and represented in a natural way, this could and probably should have been done directly in XML.

[6] We adopt Langgård and Trosterud's practice of using flag diacritics to enforce valence restrictions on verb stems; additionally, we use them to enforce grammatical number restrictions on certain noun stems, such as *kamikłuuk* (see Section 2.1.).

(2) *kamiqaqtuŋa*
kamik-qaq-tuŋa
boot-have-IND.PRS.1SG
'I have boots/a boot' or 'I am wearing boots' (MacLean, 1986a, 50)

(3) a. *savaqasiġaa*
savak-qasiq-kaa
work-at.same.time.with-IND.PRS.3SG>3SGOBJ
'he/she is working with her/him'
b. *aqpatqasiqsaġa*
aqpat-qasiq-taġa
run-at.same.time.with-IND.PST.1SG>3SGOBJ
'I ran with her/him'

(4) a. *naviksimaruaq*
navik-sima-tuaq
break-it.is.now.known-IND.PST.3SG
'it did break'
b. *naatchimaruat*
naatchi-sima-tuat
finish-it.is.now.known-IND.PST.3PL
'they did finish'

(5) a. *ilausukpit*
ilau-suk-pit
be.included-want.to-INT.2SG
'do you want to be included?'
b. *tautugukkiga*
tautuk-suk-kiga
see-want.to-IND.PRS.1SG>3SGOBJ
'I would like to see it'

(6) a. *naviktiqtuq*
navik-tiq-tuq
break-quickly-IND.PRS.3SG
'it broke instantaneously'
b. *ikuliqtuq*
iku-tiq-tuq
get.into-quickly-IND.PRS.3SG
'he/she quickly got in [e.g., a car]/on [e.g., an airplane]'

In the lexical data files, surface forms of suffixes (postbases, inflections, and enclitics) begin with a rule trigger symbol indicating the morphophonological alternation pattern conditioned by the suffix (see Sections 2.2. and 5.1.). For example, symbol '+' indicates that no stem-final segments are deleted; if the suffix begins with two consonants and is affixed to a stem ending in a consonant, the first consonant of the suffix is deleted. Thus, in example 1, suffix-initial *ġ* is deleted and stem-final *k* remains, becoming *g* due to assimilation with the following *r*. Symbol '-' indicates that any stem-final consonant is deleted; this can be seen in example 2, where the *k* of *kamik* is deleted. Symbol '+-' indicates that stem-final *k* or *q* is deleted (see example 3.a), but not stem-final *t* (see example 3.b).
Many postbases and inflectional endings have multiple phonologically conditioned allomorphs. These are specified within curly braces as a list of alternating "condition codes" and forms or form lists (a form list is enclosed within an additional pair of curly braces). In Figure 2, postbase *-[si]ma-* 'it is now known that' is defined; condition code `?C` indicates that allomorph *-sima-* occurs following a consonant (as in example 4.a); condition code `?V` indicates that allomorph *-ma-* occurs after a vowel (as in example 4.b). When allomorphs are specified, the first allomorph listed will be used as the lexical form of all allomorphs, so that all allomorphs are analyzed as the same morpheme. Shorthand notation exists for two common allomorphy patterns, eliminating the need for curly braces or condition codes. The notation `[C]` (e.g., `+[s]uk` 'want to') indicates that the bracketed consonant appears following a vowel or /t/ (see example 5.a) and is omitted otherwise (see example 5.b; note that the stem-final *k* of *tautuk-* becomes *g* due to assimilation). The notation `C//C` (e.g., `+t//liq` 'quickly, abruptly') means that the allomorph beginning with the consonant to the left of the double slash (in this case, *-tiq-*) is used if the preceding segment is a consonant (see example 6.a); otherwise the allomorph with the consonant to the right of the double slash (here, *-liq-*) is used (see example 6.b).
Inflectional endings are specified in two-dimensional "tables." An example table implementing unpossessed and possessed absolutive singular noun endings is given in Figure 3.

```
Table n +N {
  Columns {
    {}
    +1Sg  +1Du  +1Pl
    +2Sg  +2Du  +2Pl
    +3Sg  +3Du  +3Pl
    +3RSg +3RDu +3RPl
  }
  Row +Abs+Sg {
    {}
    {?V +ga ?C +a} +kpuk +kput
    {?kQ :iñ ?Otherwise -n} +ksik +ksi
    {?Always -ŋa ?notVthenV :a}
        {?Always -ŋak ?notVthenV :ak}
        {?Always -ŋat ?notVthenV :at}
    -nI +ktik {?Always {+ktiŋ -riŋ}}
  }
}
```

Figure 3: Absolutive singular inflectional suffixes defined as a table.

The keyword `Table` signals the beginning of a new table definition; this is followed by a category code to be associated with each suffix in the table, and a string of grammatical tags which apply to the table as a whole. In Figure 3, the category code is '`n`' and the grammatical tag string is `+N`. The rest of the table definition is enclosed in curly braces. A table contains exactly one `Columns` declaration and one or more `Row` declarations. The `Columns` declaration specifies one string of grammatical tags for each column in the table. In the example table, the columns in the table correspond to grammatical possessors; the first column is for unpossessed forms, and thus the grammatical tag string for this

Case & number	*Possessor* none	1Sg	1Du	1Pl	2Sg	2Du	2Pl	3Sg	3Du	3Pl	3RSg	3RDu	3RPl
Abs. Sg.	∅	+ga[a], +a[b]	+kpuk	+kput	:iñ[c], -n[d]	+ksik	+ksi	-ŋa[e], :a[f]	-ŋak[e], :ak[f]	-ŋat[e], :at[f]	-nI	+ktik	+ktiŋ[e], -riŋ[e]

[a]used with vowel-final stems; [b]used with consonant-final stems; [c]used with stems ending in a strong consonant (*k* and some *q*); [d]used with stems ending in a vowel or a weak consonant (*t* and some *q*); [e]can be used in any phonological context; [f]more conservative form; cannot be used with stems ending in a consonant cluster

Figure 4: Contents of Figure 3 presented as a row in an inflection table.

	Possessor number		
Possessor person	Singular	Dual	Plural
no possessor	∅		
1st	+ga[a], +a[b]	+kpuk	+kput
2nd	:iñ[c], -n[d]	+ksik	+ksi
3rd	-ŋa[e], :a[f]	-ŋak[e], :ak[f]	-ŋat[e], :at[f]
3rd reflexive	-nI	+ktik	+ktiŋ[e], -riŋ[e]

(see Figure 4 for footnotes)

Figure 5: Alternative representation of Figure 3 row contents as a two-dimensional table.

```
LEXICON NounInflection
+N+Abs+Sg:0 Enclitics
+N+Abs+Sg+1Sg:%?V%+ga Enclitics
+N+Abs+Sg+1Sg:%?C%+a Enclitics
+N+Abs+Sg+1Du:%+kpuk Enclitics
+N+Abs+Sg+1Pl:%+kput Enclitics
+N+Abs+Sg+2Sg:%?kQ%:iñ Enclitics
+N+Abs+Sg+2Sg:%?Vt%-n Enclitics
+N+Abs+Sg+2Du:%+ksik Enclitics
+N+Abs+Sg+2Pl:%+ksi Enclitics
+N+Abs+Sg+3Sg:%-ŋa Enclitics
+N+Abs+Sg+3Sg:%?notVthenV%:a Enclitics
+N+Abs+Sg+3Du:%-ŋak Enclitics
+N+Abs+Sg+3Du:%?notVthenV%:ak Enclitics
+N+Abs+Sg+3Pl:%-ŋat Enclitics
+N+Abs+Sg+3Pl:%?notVthenV%:at Enclitics
+N+Abs+Sg+3RSg:%-nI Enclitics
+N+Abs+Sg+3RDu:%+ktik Enclitics
+N+Abs+Sg+3RPl:%+ktiŋ Enclitics
+N+Abs+Sg+3RPl:%-riŋ Enclitics
```

Figure 6: One possible representation of the contents of Figure 3 in *lexc* format.

column is empty. Each `Row` declaration specifies a string of grammatical tags that apply to that row followed by a list of the surface forms of the suffixes in that row. The row in Figure 3 defines absolutive singular endings and is accordingly tagged `+Abs+Sg`. In the actual inflection file, the table of noun inflections contains 24 rows, one for each possible combination of case and grammatical number, but due to space constraints only one row is reproduced in Figure 3. Conceptually, the contents of this figure correspond to the table shown in Figure 4. Thinking in terms of a thirteen-column table can be daunting; we have dealt with this challenge by strategically inserting white space and newlines in both the column list and the row contents, as can be seen in Figure 3. This extra space is ignored by the software converting the tables to XML and *lexc*, but allows humans editing the file to visualize each row in terms of a more compact table, such as the one presented in Figure 5.

Like postbases, inflectional suffixes may exhibit allomorphy, and the same notation used for postbases with allomorphs is used in inflection tables. The special condition code `?Always` is used to denote variants which are not phonologically conditioned, and the code `?Otherwise` indicates that an allomorph occurs in all environments where no other allomorphs occur. Condition code `?kQ` specifies an allomorph that attaches to stems ending in *k* or "strong" *q*.[7] Condition code `?notVthenV` prohibits an allomorph from attaching to a stem ending in a vowel cluster.

The lexical form of each inflectional ending is the concatenation of table tags + row tags + column tags. For example, in Figure 3, the tag string `+N+Abs+Sg+1Pl` (absolutive singular noun with first person plural possessor) corresponds to the suffix `+kput`.

The table mechanism provides a practical alternative to representing inflectional information directly in *lexc*, as for example in Figure 6. The *lexc* representation involves considerable redundancy, both in the tag strings and in the continuation classes (although other langauges might require a more complex continuation class structure than the one shown here). On a more subjective note, we believe the table structure is easier to read and (assuming one is entering inflectional data from tabular printed sources) considerably easier to write.

5.3. Condition codes

The specification of allomorphic variants in the source files is simple enough, but the back-end implementation of this feature is somewhat more complex. Allomorphic variants might be handled within *lexc* by creating separate LEXICONs for morphemes belonging to each conditioning environment and implementing continuation classes that respect the restrictions associated with each environment. For example, vowel-final verb stems and postbases would continue

[7]Some instances of *q* at the end of noun stems are considered "strong" and interact with certain suffixes in the same way as *k* and differently from how "weak" instances of *q* interact with the same suffixes. The distinction between strong and weak *q* is partially conditioned by phonological factors and partially an idiosyncratic attribute of specific stems. In all cases, the morphemes sensitive to this distinction are noun inflection suffixes.

to a LEXICON containing allomorphs sensitive to that environment and excluding allomorphs which attach only to consonant-final stems. This approach is further complicated by the interaction of different conditioning environments. For example, the imperative singular intransitive ending has allomorphs sensitive to the following environments: stems of the form (C)VCV-, stems ending in *k* or *q* or consisting of more than two syllables and ending in a vowel, stems ending in *t*, and stems ending in a two-vowel cluster. Stems of the form (C)VCV would need to continue to a class containing not only suffixes sensitive to the form (C)VCV-, but also suffixes conditioned by a preceding vowel, and suffixes without conditions. Stems ending in a two-vowel cluster would need to continue to a distinct LEXICON containing suffixes conditioned by -VV-, suffixes conditioned by a preceding vowel, and suffixes without conditions. It should be clear that handling phonologically conditioned allomorphy via the architecture of a *lexc* grammar would require a complex maze of LEXICONs and continuation classes.

Fortunately, *xfst* offers an elegant alternative which leverages its pattern-matching strengths. In the script which converts source files into *lexc* format, each allomorph is tagged with a rule trigger corresponding to its condition code. LEXICONs are then constructed on the basis of morphotactics alone, without phonological considerations. A lexicon defined in this way will overgenerate, attaching suffix allomorphs to stems regardless of whether those stems fulfill the requisite phonological conditions. To address this problem, the lexicon is filtered through a series of rules, defined in a separate file, which are sensitive to the condition code triggers; these rules eliminate any string where the characters preceding the trigger do not match a specified pattern. After accepting a string, the rules remove the trigger from the string. This approach allows a clean separation between morphotactic and phonological constraints.

Since this filtering mechanism requires us to write a number of *xfst* rules, one might wonder whether it would not be simpler to create a set of rewrite rules to produce the appropriate allomorphs directly. For some languages (particularly with language documentation written in a certain style), this may be the best approach. In our case, this approach would have two considerable drawbacks. First, it is unlikely that the set of rules required to produce all allomorphs could be smaller than the set of filter rules currently in place; in addition to specifying specific alternations, rewrite rules would need to implement the environments currently specified in the filter rules, and in many cases a single filter would need to correspond to multiple rewrite rules. Second, the filter system is a natural implementation of the way allomorphs are specified in MacLean's Iñupiaq language materials (1981; 1986a; 1986b; unpublished), which serve as the bulk of our source material; translating this into a set of rewrite rules would have required significant additional work.

Two particularly important comments from an anonymous reviewer deserve to be addressed here. First, although we have treated Iñupiaq inflection in terms of phonologically conditioned allomorphs, it is also possible to conceive of it in terms of lexically conditioned inflection classes. The strongest evidence for this analysis is the fact that, for some inflectional endings, a different form is used for stems ending in "strong *q*" than for stems ending in "weak *q*" (see footnote 7). The treatment of inflection classes is commonly and properly done in *lexc*, and this is the approach taken by Langgård and Trosterud. On the other hand, although strong and weak *q* might not properly be considered distinct phonemes, it is trivial and unproblematic to treat them as if they were, and having done so, allomorphs of inflectional endings may be chosen entirely on the basis of the preceding (pseudo)phonological environment. The majority of Iñupiaq inflectional endings have a single allomorph (or a set of allomorphs produced entirely from fully productive phonological processes), so for Iñupiaq, the inflection class treatment would necessitate a large amount of duplication, which can be avoided by treating different forms of inflectional endings as phonologically-conditioned variants. The reviewer's other criticism is that any advantage that the filter approach may have over a LEXICON/continuation class approach in terms of elegance must be accompanied by a concomitant decrease in performance, and that this is performance hit will become more severe as the lexicon increases in size. This is certainly true at compile time; although we have no hard numbers, when we switched from a *lexc*-based approach to the filter approach, we noticed that compilation took perhaps two minutes when before it was well under 30 seconds (on a modest computer built in 2005). However, we do not notice or expect there to be an important difference in performance at runtime, since in both cases the end result of compilation is a highly optimized finite-state transducer the traversal of which is straightforward. The primary problem then, compile time, is an issue for developers but not for end users. Although Moore's Law[8] cannot make this problem go away, it does suggest that the impact on developers will diminish over time. When considering performance issues, one must also bear in mind that the *lexc* approach may be considered to impose a performance hit during development, in that the construction of appropriate LEXICONs and continuation classes requires the developer to perform by hand (or accomplish in some other automated way, which would also take time) the filtration which in our system is done by *xfst*.

6. Current status and future plans

Currently, the North Slope Iñupiaq transducer implements most of the lexical, morphotactic, morphophonological, and inflectional information contained in Edna MacLean's *Abridged Iñupiaq and English Dictionary* (1981) and three-year university-level Iñupiaq curriculum (1986a; 1986b; unpublished), as well as most North Slope Iñupiaq entries from Donald H. Webster and Wilfried Zibell's *Iñupiat Eskimo Dictionary* (1970). We are in the process of adding additional stems and postbases from the private files of Edna MacLean.

At present, the transducer does not attempt to handle proper nouns or recent loanwords, which are subject to slightly different morphophonological rules (MacLean, 1986a, pp. 154–155). More work also remains to be done to expand the

[8] Moore's Law predicts that the number of transistors on an integrated circuit will double roughly every two years. The practical implication of this trend is that new computers are consistently and increasingly more performant than their predecessors.

set of stems and postbases in the lexicon. We are working to develop a stem-guessing transducer (Beesley and Karttunen, 2003, pp. 445–448) which may help with that process; a postbase guesser is also not out of the question.

The transducer has been informally tested against a corpus of texts for university-level second language learners of Iñupiaq. At present, the transducer generates at least one analysis for 3414 out of 4406 tokens (77.49%) and 2021 out of 2887 unique types (70.00%). So far, no systematic attempt has been made to evaluate the accuracy of the analyses produced by the transducer. An additional 5000 words, unseen by the developers, have been set aside for additional testing.

As the transducer becomes more mature and able to recognize more words, we hope to incorporate it in additional technologies that may benefit the Iñupiaq community. In particular, we hope to develop a spell-checker, which Iñupiaq language learners have expressed an interest in, and Iñupiaq-aware OCR software (Yoo, 2008) to help the community digitize existing materials in text format. We would also like to develop tools geared toward the academic community, such as a lemmatizer. The architecture of the transducer could also be easily and naturally applied to other dialects of Iñupiaq and probably to languages within the Yupik branch of the Eskimo language family as well.

7. Conclusions

This paper describes some of the specific challenges presented by the Iñupiaq language which a computational morphology must address, and how we have dealt with those points by creating language-specific formats for representing lexical and morphophonological information. Our system provides customized treatment for each of the fundamental morpheme types found in Iñupiaq: bases, postbases, inflectional endings, and enclitics. In these formats, data need not be regrouped according to grammatical category or phonological structure; it can be entered in the order in which it appears in the dictionary. Inflectional endings are specified in a two-dimensional format corresponding more closely to the way linguists conceive of inflectional paradigms. Special mechanisms have been developed to handle allomorphy and long-distance dependencies (valence restrictions on verbs and number restrictions on certain nouns) in a natural way.

While the formats used for lexical data in this project are probably geared too specifically to Iñupiaq to be reused for any but the most closely related languages, the concept of defining data in a convenient, language-specific format and converting this data into the format required by a general tool such as *lexc* has merit for a wide variety of languages. Additionally, languages with complex phonologically conditioned allomorphy may benefit from a lexical treatment similar to the one described here, where an overgenerating lexicon with tagged allomorphs is filtered through a set of rules enforcing the conditions associated with those allomorphs.

Acknowledgements

This work was supported by the US National Science Foundation, Award 0534217. We also gratefully acknowledge Ida Mayer, J. Eliot DeGolia, Sai Venkateswaran, Paul Lundbland, and Shinjae Yoo, who prepared the corpus of Iñupiaq texts used to test the transducer, and three anonymous reviewers for their valuable feedback.

Abbreviations used

1	1st person
2	2nd person
3	3rd person
3R	3rd person reflexive
DU	dual
IND	indicative
INT	interrogative
OBJ	direct object
PL	plural
PRS	present
PST	past
SG	singular

8. References

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI, Stanford, California.

Kenneth R. Beesley. 2003. Finite-state morphological analysis and generation for Aymara. In *Proceedings of the Workshop on Finite State Methods in Natural Language Processing, 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL 10), April 13–14 2003, Budapest, Hungary*, pages 19–26.

Kenneth R. Beesley. 2004a. Downtranslation of XML dictionaries to lexc LEXICONs: Third draft, October 12. Published online: `http://www.stanford.edu/~laurik/fsmbook/clarifications/xmldowntrans.html`.

Kenneth R. Beesley. 2004b. Morphological analysis and generation: A first step in natural language processing. In Julie Carson-Berndsen, editor, *First Steps in Language Documentation for Minority Languages: Computational Linguistic Tools for Morphology, Lexicon and Corpus Compilation, Proceedings of the SALTMIL Workshop at LREC 2004, May 24, Lisbon, Portugal*, pages 1–8.

Lawrence D. Kaplan. 1981. *Phonological Issues in North Alaskan Inupiaq*. Number 6 in Research Papers. Alaska Native Language Center, Fairbanks, Alaska. Published version of Kaplan's 1979 doctoral dissertation.

Lauri Karttunen, Ronald M. Kaplan, and Annie Zaenen. 1992. Two-level morphology with composition. In *COLING 1992, 14th International Conference on Computational Linguistics, August 23–28, Nantes, France*, pages 141–148.

András Kornai. 1999. Extended finite state models of language. In Adrás Kornai, editor, *Extended Finite State Models of Language*, Studies in Natural Language Processing, pages 1–5. Cambridge University Press, Cambridge, England and New York, New York.

Michael E. Krauss. 2007. Native languages of Alaska. In Osahito Miaoka, Osamu Sakiyama, and Michael E. Krauss, editors, *The Vanishing Voices of the Pacific Rim*. Oxford University Press, Oxford, England.

Edna Ahgeak MacLean. 1981. *Iñupiallu Tanŋiḷḷu Uqaluŋisa Iḷaŋich = Abridged Iñupiaq and English Dictionary*. Alaska Native Language Center, Fairbanks, Alaska.

Edna Ahgeak MacLean. 1986a. *North Slope Iñupiaq Grammar: First Year*. Alaska Native Language Center, Fairbanks, Alaska, third edition.

Edna Ahgeak MacLean. 1986b. *North Slope Iñupiaq Grammar: Second Year (Preliminary Edition for Student Use Only)*. Alaska Native Language Center, Fairbanks, Alaska.

Edna Ahgeak MacLean. unpublished. North Slope Iñupiaq grammar: Third year. Draft manuscript.

Elaine Uí Dhonnchadha. 2003. Finite-state morphology and Irish. In *Proceedings of the Workshop on Finite State Methods in Natural Language Processing, 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL 10), April 13–14 2003, Budapest, Hungary*, pages 43–49.

Donald H. Webster and Wilfried Zibell. 1970. *Iñupiat Eskimo Dictionary*. Summer Institute of Linguistics, Fairbanks, Alaska. Digitized by Alaskool.org: `http://www.alaskool.org/language/dictionaries/inupiaq/default.htm`.

Shinjae Yoo. 2008. A smart OCR for Inupiaq. Final presentation of graduate-level semester project in language and information technologies at Carnegie Mellon University, December 12.

Leaving Behind the Less-Resourced Status.
The Case of Latin through the Experience of the *Index Thomisticus* Treebank

Marco Passarotti
Università Cattolica del Sacro Cuore
Largo Gemelli 1, 20123 Milan, Italy
E-mail: marco.passarotti@unicatt.it

Abstract

Despite its key role in the history of computational linguistics, thanks to the pioneering work by Roberto Busa SJ on the *Index Thomisticus*, Latin can still be considered as a less-resourced language. Although during the last decades several Latin texts have been digitized, only a few of them have been linguistically tagged, while most still lack linguistic tagging at all. However, while the less-resourced status affects historical languages in general, over the past few years a number of language resources for Latin and other historical languages have been started, among which are several treebanks. Presenting the experience of the *Index Thomisticus* Treebank project and, particularly, its valency lexicon, this paper reports some general insights about the creation and use of language resources for less-resourced languages, showing that, although creating from scratch a language resource for a less-resourced language still remains a labor-intensive and time-consuming task, today this is simplified by exploiting the results provided by previous similar experiences in language resources development.

1. Introduction

Despite its key role in the history of computational linguistics, thanks to the pioneering work by Roberto Busa SJ on the *Index Thomisticus* (IT; 1974-1980), Latin can still be considered as a less-resourced language, lacking powerful NLP tools and a broad suite of state-of-the-art language resources (LRs) such as annotated corpora and lexica.

However, while the less-resourced status affects historical languages in general (because of reasons such as being not commercially interesting or lacking native speakers), over the past few years a number of LRs for Latin and other historical languages have been started. Among these LRs are treebanks for Middle, Early Modern and Old English, Early New High German, Medieval Portuguese, Ugaritic, Ancient Greek and several translations of the New Testament into Indo-European languages[1].

As far as Latin is concerned, while trying to meet the needs of the research community working on Latin to have better access to and understanding of textual data, we realized that basic Latin LRs and NLP tools were missing. This is the reason why in 2005 we designed a set of basic LRs and technologies for Latin and started to create a Latin treebank based on the IT data.

Moreover, the collaboration with other similar projects and the exploitation of tools developed over the years for the creation and use of LRs established a kind of virtuous circle for the development of further NLP tools and LRs for Latin, such as lexica. Indeed, the relation between annotated corpora and lexical resources should be circular: while linguistic annotation of textual data is supported and improved by the use of basic lexical resources, these latter can be induced from annotated data in a corpus-driven fashion. This is what we experienced in the *Index Thomisticus* Treebank (IT-TB) while creating a Latin valency lexicon from the annotated data.

Presenting the experience of the IT-TB project, this paper reports some general insights about the creation and use of LRs for less-resourced languages. The paper is organized as follows: section 2 describes the state of the art of the available LRs and NLP tools for Latin; section 3 presents a basic language resource kit for Latin; section 4 deals with some of the main features and achievements of the IT-TB project, presenting the data, the annotation style, the parsing procedures and, particularly, the valency lexicon; finally, section 5 draws some general conclusions on the creation and use of LRs for less-resourced languages and provides an outlook on the next steps of the project.

2. Survey of LRs and NLP tools for Latin

Although during the last decades several Latin texts have been digitized[2], only a few of them have been linguistically tagged, while most still lack linguistic tagging at all.

Only recently (namely, in 2005) two projects started to develop Latin treebanks. These are the IT-TB by the Catholic University in Milan on texts from the IT (McGillivray et al., 2009)[3] and the Latin Dependency Treebank (LDT) by the Perseus Digital Library in Boston on texts of the Classical era (Bamman & Crane, 2007).

[1] For references, see Bamman et al. (2009).

[2] See for instance the Perseus Digital Library at Tufts University in Boston, or the textual databases by CTLO in Turnhout (Centre "Traditio Litterarum Occidentalium") and by LASLA at the University of Liège (Laboratoire d'Analyse Statistique des Langues Anciennes).

[3] Busa started early in the '70s to plan a project aimed at the syntactic annotation of the IT data. Today, the IT-TB project has undertaken this task as part of the wider "Lessico Tomistico Biculturale" project (LTB), whose goal is the development of a Thomistic lexicon grounded on the IT data.

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 27–32

Later on, a third Latin treebank was started at the University of Oslo as part of the project PROIEL (Pragmatic Resources in Old Indo-European Languages), which is aimed at the syntactic annotation of the oldest extant versions of the New Testament in Indo-European languages: Latin, Greek, Gothic, Armenian and Old Church Slavonic (Haug & Jøndal, 2008).

The size of these treebanks is presently around 80,000 annotated words for IT-TB, 55,000 for LDT and 100,000 for the Latin section of the PROIEL corpus.

In regard to Latin lexical resources, many Latin dictionaries and lexica are today available on-line or on CD-ROM. Some of the most relevant are the Lewis-Short dictionary provided by Perseus, the *Thesaurus Linguae Latinae* from the Bayerische Akademie der Wissenschaften in Munich, the *Thesaurus Formarum* (TF-CILF) from the CTLO and the *Neulateinische Wortliste* by Johann Ramminger (http://www.lrz-muenchen.de/~ramminger/). Presently, the main project aimed at developing a Latin lexical resource is Latin WordNet (Minozzi, 2008), which is integrated within the wider MultiWordNet project (http://multiwordnet.fbk.eu).

However, although WordNet is a lexical resource that can be used for NLP tasks such as information extraction, data mining, word sense disambiguation and topic classification, the available NLP tools for Latin are still far from providing automatic processing of such tasks. In this domain, three morphological analysers of Latin are presently available, namely LEMLAT (Passarotti, 2004), Whitaker's *Words* (http://archives.nd.edu/words.html) and *Morpheus* (Crane, 1991), this latter being first created for Ancient Greek in 1985 and extended to support Latin in 1996. Specific tools for morpho-syntactic disambiguation and Part-of-Speech (PoS) tagging have been developed by LASLA for the annotation of their textual database, while a first attempt at Latin dependency parsing is described by Koch (1993), who reports on the enhancement for Latin of an existing dependency parser. Finally, Koster (2005) describes a rule-based top-down chart parser, automatically generated from a grammar and a lexicon built according to a two-level formalism (AGFL: Affix Grammar over a Finite Lattice).

3. A Basic Language Resource Kit for Latin

In order to identify the best strategy to follow over the upcoming years to move Latin from being a less-resourced language to being a language with basic LRs, we first defined a basic minimal set of underlying LRs and tools that are considered necessary for language technology applications working on Latin.

To achieve this aim, we grounded our decisions on the BLARK concept (Basic Language Resource Kit) consisting in defining "for every language a specification of the minimum general text or spoken corpus, basic tools to manipulate it and skills required to be able to do any pre-competitive research for the language" (Mapelli & Choukri, 2003, p. 4).

Since spoken data is missing for Latin, we sketched a set of basic components comprised of technologies for written languages only. Among human language technologies (HLT), BLARK distinguishes between modules (software components used for the development of HLT applications), applications (which make use of HLT) and data (used to create, refine and evaluate the modules). Following these requirements, a BLARK-like set was sketched for Latin, consisting of the following components.

Modules:

- Text pre-processing (tokenization and named-entity recognition)
- Lemmatization: morphological analysis and morpho-syntactic disambiguation (PoS taggers)
- Syntactic analysis: parsers and shallow parsing
- Anaphora resolution
- Semantic and pragmatic analysis

Applications:

- Entering and acquiring information: typing, digitization, annotation, OCR systems[4]
- Document management: automatic and computer-assisted indexing
- Information retrieval and presentation

Data:

- Unannotated corpus of text
- Syntactically annotated corpus of text (treebank)
- Monolingual lexicon (valency lexicon)
- Semantically and pragmatically annotated corpus of text

Considering the state of the art of Latin LRs and NLP tools, the following were recognized as the components most urgently needed in order to meet the requirements of the basic set.

- Modules: NLP tools for the automatic processing of the morpho-syntactic and syntactic layers of annotation (PoS taggers and parsers)
- Applications: tools for data annotation and for information retrieval
- Data: a treebank and a valency lexicon

Although components like semantic and pragmatic analysis, as well as anaphora resolution, are part of the set, we deferred their development, since we believe syntax to be an essential level of analysis in view of such "higher" tasks.

Moreover, the present availability of data-driven and language-independent NLP tools, such as probabilistic PoS taggers and parsers, strengthened our idea of starting to build the basic kit for Latin beginning first of all with the development of a Latin treebank. Indeed, our short-term perspective was to exploit the data from the treebank in two ways: (a) to train NLP tools for morpho-syntactic disambiguation and syntactic analysis, and (b) to use the data as the basis for several subsequent layers of annotation, including anaphora resolution, and semantic and pragmatic analysis. Similarly, we wanted to enhance the syntactic annotation with valency

[4] Specific OCR systems for printed and handwritten characters are particularly required for digital and computational philology purposes.

information, in order to induce a valency lexicon from the treebank data.

4. The *Index Thomisticus* Treebank

4.1 The *Index Thomisticus*

Started by Roberto Busa SJ in 1949, the IT is a corpus containing the *opera omnia* of Thomas Aquinas (118 texts) as well as 61 texts by other authors related to Thomas, for a total of approximately 11 million words, each morphologically tagged and lemmatized by hand. The corpus can be browsed on CD-ROM or on-line at the following address: http://www.corpusthomisticum.org.

4.2 Annotation Style

Since the *Index Thomisticus* Treebank and the Latin Dependency Treebank were the first projects of their kind for Latin, no prior established guidelines were available to rely on for syntactic annotation. Rather than have each treebank project decide upon and record each decision for annotating the data, the two projects decided to pool their resources and create a single annotation manual that would govern both treebanks (Bamman et al., 2007). Rather than design the manual from scratch, we chose to follow the annotation style developed for the 'analytical layer' by the Prague Dependency Treebank of Czech (PDT; Hajič et al., 1999)[5]. Only minor changes were applied, for the treatment of specific or idiosyncratic constructions of Latin (Bamman et al., 2008).

PDT is a dependency-based treebank with a three-layer structure, ordered as follows: (1) a morphological layer: lemmatization and full morphological annotation; (2) an 'analytical layer': dependency-based superficial (surface) syntactic annotation; (3) a 'tectogrammatical layer': annotation of the underlying meaning of the sentence, based on the Functional Generative Description framework (FGD; Sgall et al., 1986).

In the IT-TB and LDT projects, we have chosen the PDT annotation style for both linguistic and "structural" reasons.

As far as the former are concerned, Latin and Czech share some relevant properties such as being richly inflected, having a moderately free word-order and an high degree of synonymity and ambiguity of the endings, and showing discontinuous phrases (i.e. phrases broken up by words of other phrases: 'non-projectivity')[6]. Both languages have 3 genders (masculine, feminine, neuter), cases with roughly the same meaning and no articles.

As for the latter, the PDT three-layer structure is ideal both for our present needs and for the perspectives of development of new Latin LRs and tools. Indeed, the analytical annotation in PDT is not meant to be a layer standing on its own, but is intended as a technical step towards the tectogrammatical annotation. The strict relation between the overall structure of the annotation workflow in PDT and a sound background theory like FGD allows us to consider each single layer of annotation as one part of a general framework that is driven by a functional perspective aimed at understanding the underlying meaning of the sentence. This task is performed through topic-focus articulation tagging, ellipsis resolution and semantic role labelling, this latter making use of labels (called 'functors') such as Actor, Patient, Addressee, Origin, Effect and several kinds of free adverbials (temporal, local, causal, manner, etc.). Pragmatic tagging (topic-focus articulation), anaphora resolution and, ultimately, semantic analysis are just components of the basic kit of Latin LRs and tools that are still missing.

Moreover, the adoption of PDT as the main reference framework not only provided our annotation efforts with a sound theoretical background, but also gave us the opportunity to re-use tools for annotation and retrieval which had been developed by PDT for its own purposes. Particularly, for IT-TB manual and semi-automatic annotation we adopted the tree editor TrEd by Petr Pajas, while on-line browsing of the IT-TB data can be performed through the searching and viewing interface Netgraph (Mírovský, 2006) at the IT-TB website: http://itreebank.marginalia.it.

4.3 Parsing and PoS Tagging

After an early phase of manual annotation, we started to exploit the available annotated data to train and test a number of probabilistic dependency parsers. This was done in order to increase the quality and speed of the annotation process. Indeed, in this way annotators no longer have to draw trees from scratch, but need only to check the automatically produced trees and to manually correct mistakes.

In our recent work (Passarotti & Dell'Orletta, 2010), we describe a number of modifications that we applied to DeSR parser (Dependency Shift-Reduce; Attardi, 2006), including the design of a feature model specific to Medieval Latin as well as revision and combination techniques. Using a training set of 61,024 tokens (2,820 sentences), this improved the previously available accuracy rates, reaching 80.02% for LAS, 85.23% for UAS and 87.79% for LA[7].

Since the IT data are morphologically tagged, our first priority has been automatic syntactic parsing. However, we also started to train PoS taggers, in order to automatically perform morpho-syntactic disambiguation of the IT morphological lemmatization. Bamman and

[5] Although they differ in some details, the PROIEL treebank annotation guidelines are quite similar to those governing IT-TB and LDT. An automatic conversion procedure from PROIEL to the IT-TB and LDT annotation style is ongoing.

[6] The condition of projectivity in a dependency tree says that if a node *a* depends on *b* and there is a node *c* between *a* and *b* in the linear ordering, *c* depends (directly or indirectly) on *b*. The non-projective nodes are those where such condition is not met.

[7] LAS (Labeled Attachment Score) is the percentage of tokens with correct head and relation label; UAS (Unlabeled Attachment Score) is the percentage of tokens with correct head; LA (Label Accuracy) is the percentage of tokens with correct relation label (Buchholz & Marsi, 2006).

Crane (2008) report accuracy rates of around 95% in resolving PoS, reached with a PoS tagger (TreeTagger; Schmid, 1994) trained on a set of approximately 47,000 tokens from LDT. Our preliminary results for PoS tagging, using the HMM-based HunPos tagger (Halácsy et al., 2007) and the IT-TB training set (61,024 tokens), were the following: 96.75% in correctly disambiguating coarse-grained PoS + fine-grained PoS, and 89.90% if morphological features are also considered[8].
Given the small training set, these are quite high rates, resulting from the use of language-independent NLP tools that were not specifically designed for IT-TB purposes.

4.4 Valency Lexicon

The present availability of Latin treebanks fosters the creation of new lexical resources for Latin that match with the annotated data. Indeed, the evidence provided by such corpora can be fully represented in lexical resources induced from the data. Subsequently, such resources can in turn be used to support the annotation of new textual data.
In particular, the creation of a lexicon can be pursued by both intuition-based and data-driven approaches, according to the role played by human intuition and by the empirical evidence provided by annotated corpora such as treebanks.
For instance, lexica like PropBank (Kingsbury & Palmer, 2002), FrameNet (Ruppenhofer et al., 2006) and PDT-VALLEX (Hajič et al., 2003) have been created in an intuition-based fashion and then checked and improved with examples excerpted from corpora.
On the other hand, research in lexical acquisition has recently made available a number of data-driven valency lexica automatically acquired from annotated corpora, such as VALEX (Korhonen et al., 2006) and LexShem (Messiant et al., 2008).
In the IT-TB project we followed a data-driven approach, inducing a valency lexicon for Latin verbs from IT-TB data (McGillivray & Passarotti, 2009). The notion of valency is generally defined as the number of obligatory complements required by a word: these complements are usually named 'arguments', while the non-obligatory ones are referred to as 'adjuncts'. Although valency can be assigned to different PoS (usually verbs, nouns and adjectives), scholars have mainly focused their attention on verbs, so that the notion of valency often coincides with verbal valency. Presently, the size of the IT-TB valency lexicon is 432 entries (corresponding to 5,966 verbal occurrences in the treebank)[9]. The lexicon is automatically updated as the amount of the annotated data increases.
A similar approach has been pursued by LDT. Bamman and Crane (2008) describe a Latin 'dynamic lexicon' automatically extracted from the Perseus Digital Library, using LDT data as training set. The lexicon reports qualitative and quantitative information on the subcategorization patterns and selectional preferences of each word as it is used in every Latin author of the corpus. Relying on morphological tagging and statistical syntactic parsing of a large corpus (around 3.5 million words), only the most common arguments and the most common lexical fillers of these arguments are shown, thus reducing the noise caused by the automatic pre-processing of data.
While PDT, as a project, represents the main reference model for IT-TB, in the development of the valency lexicon we did not follow the same approach. Indeed, while PDT-VALLEX was created before the annotation of PDT started and the annotated data were linked subsequently to the single items in the lexicon, the IT-TB valency lexicon results from the opposite procedure. The lexicon is created in an annotation-driven fashion and the valency of a lexical item is defined as annotators get through its first occurrence in the data. Furthermore, since the IT-TB valency lexicon relies on data annotated on the analytical layer (and not on the tectogrammatical one), it just reports for each entry the number of the arguments occurring on the surface syntactic structure, while no information on semantic roles is provided[10].
This approach has pros and cons. On the one hand, not grounding the annotation decisions on a previously available valency lexicon developed in an intuition-based fashion can lead to inconsistencies in annotation, since annotators do not make their decisions about valency on the basis of one common lexicon. On the other, the exploitation of the data in our approach allows an in-depth evaluation of the quality of the annotation, making it possible to discover inconsistencies and to make decisions on unclear cases. At any rate, our choice to develop a valency lexicon from an available treebank was strictly motivated by the less-resourced status of Latin, which requires that the creation of a new LR results from exploiting as much as possible the available resources.

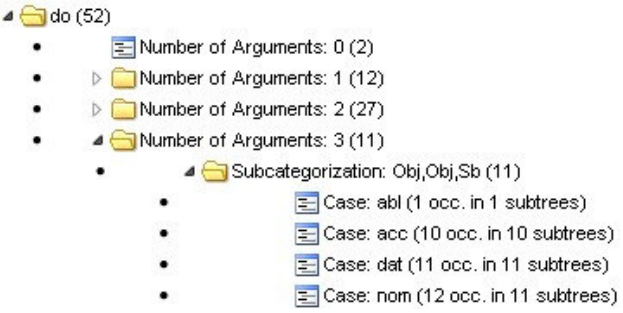

Figure 1: Mock-up of part of the entry for *do, -are*.

The lexicon will be soon made accessible at the IT-TB website through a user-friendly interface. Figure 1 shows

[8] In the IT tagset there are 5 different coarse-grained PoS tags and 14 fine-grained PoS tags.
[9] The only available list of Latin verbs which also provides their valencies is reported by Happ (1976, pp. 480--565), who created a lexicon on a basis of around 800 verbal occurrences excerpted from Cicero's *Orationes*.
[10] This means that, in the IT-TB valency lexicon, the arguments are distinguished according to their syntactic role, i.e. 'analytical' function: subject, (direct or indirect) object, nominal predicate and object complement.

an example of what a lexical entry looks like in the interface. It reports a part of the entry for the verb *do, -are* (*to give*). It is shown that in the IT-TB there are 52 instances of *do* , 2 of which occur with no arguments, 12 with 1 argument, 27 with 2 arguments and 11 with 3 arguments. In particular, the subcategorization pattern of all the 11 trivalent cases is formed by 2 objects and 1 subject; furthermore, information on the case of each argument is provided[11].

Clicking on each level of the lexical entry, all the sentences included in such level are shown and, for each of them, the place of its occurrence in the IT-TB and a subtree showing the verbal subcategorization pattern are provided. In each sentence, the verbal head and its arguments are highlighted (in different colours).

The lexicon can be browsed in many other ways, as for instance by number of arguments, subcategorization pattern, surface order of the arguments and lexical fillers. For example, the lexicon can be queried searching for all the verbal occurrences having 2 arguments, one of which is a subject and the other an indirect object (in dative), the latter occurring in preverbal position and being a form of a specific lemma[12].

5. Discussion and Conclusion

Although creating from scratch an LR for a less-resourced language still remains a labor-intensive and time-consuming task, today this is simplified by exploiting the results provided by previous similar experiences in LRs development. Such results, in terms of methods, data and tools, can be re-used with limited effort and applied to other languages.

Together with the use of those LRs and tools that are available for the less-resourced language in question, the re-use of previous research experience is even more helpful in those cases where they involve LRs for languages that share certain primary properties with the less-resourced language for which new LRs have to be created. These language relationships can be used for porting, in a a rapid and low-cost fashion, LRs and tools from one language to another, taking an approach to LRs creation and use that stands in the middle between knowledge-free approaches and knowledge-intensive ones.

The IT-TB is a case which shows how good results can be achieved in quite a short time, through adapting already existing language technologies developed over the years for well-resourced languages and particularly for Czech, which shares with Latin a number of linguistic properties. For instance, the re-use of tagging and browsing software applications that were created for PDT purposes allowed saving time and funds that otherwise should be spent to create such tools specifically for IT-TB.

Furthermore, the language-independent nature of available probabilistic NLP tools makes them extremely useful for the purposes of projects aimed at the creation of LRs for less-resourced languages, since there is no need to develop specific (usually, rule-based) NLP tools for the processing of just one language or, sometimes, for the aims of just one project.

Once a small amount of annotated data has been made available by the IT-TB project, this has been in turn used (a) to train probabilistic NLP tools, such as PoS taggers and parsers, achieving promising results despite a quite small training set, and (b) to induce another new LR for Latin, namely the IT-TB valency lexicon.

One of the advantages of working on a less-resourced language is the small number of people who are involved. Although these people usually work on different projects, they can easily collaborate to find common solutions to common problems. Such collaboration can start with the very beginning of the projects, as in the IT-TB and LDT cases, where common annotation guidelines were developed before the annotation of data was started. This allows the setting of standards that are really shared by the projects and not imposed on the projects in a top-down manner. In our case, collaboration with LDT is even further essential since Latin is a language with a long diachronic usage extending over more than two thousand years. While the two projects are dealing with Latin dialects separated by 13 centuries, sharing a single annotation manual proved to be very useful for comparison purposes, such as checking annotation consistency, making annotation decisions on a wider number and kind of examples, or diachronically studying specific syntactic constructions.

The overall design is important as well. Grounding a project on a sound theoretical framework (like FGD) and aiming at the creation of a pre-defined set of basic LRs motivates each step of the work, which is thus considered in a wider perspective.

Our goal in the near future is to apply named-entity recognition systems to the IT data and to enlarge the amount of analytically annotated data in IT-TB, relying on the good results provided by DeSR. Annotation of data at the tectogrammatical layer will be started as well, still grounding on PDT guidelines and using TrEd as annotation editor. This will also enrich the IT-TB valency lexicon, enhancing the current argument information with semantic roles (functors). Finally, since PROIEL is a multilingual resource providing syntactic annotation of the same texts from the New Testament in several different languages, the PROIEL annotated corpus is a good starting point for the development of a multilingual valency lexicon based on treebank data. This multilingual aspect will be further improved by linking the lexical entries of the IT-TB valency lexicon with the corresponding entries in the Latin WordNet and, from there, they will be linked to the WordNets of all the other languages included in the MultiWordNet project.

[11] In figure 1, 'abl' stands for 'ablative', 'acc' for 'accusative', 'dat' for 'dative' and 'nom' for 'nominative'. The ablative argument occurs in the passive use of *do*, whose agentive argument is a prepositional phrase headed by the preposition *a/ab* (*by*), which takes the ablative case.

[12] In those cases where an argument is not a single word or a prepositional phrase but a subordinate clause, the lexical filler reported in the lexicon is the verb heading this clause.

6. References

Attardi, G. (2006). Experiments with a Multilanguage Non-Projective Dependency Parser. In *Proceedings of the CoNLL-X*, pp. 166--170.

Bamman, D. & Crane, G. (2007). The Latin Dependency Treebank in a cultural heritage digital library. In *Proceedings of LaTeCH 2007. Prague, Czech Republic*, pp. 33--40.

Bamman, D. & Crane, G. (2008). Building a Dynamic Lexicon from a Digital Library. In *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2008)*.

Bamman, D., Crane, G., Passarotti, M. & Raynaud, S. (2007). *Guidelines for the Syntactic Annotation of Latin Treebanks.* Technical report. Boston: Tufts Digital Library.

Bamman, D., Mambrini, F. & Crane, G. (2009). An Ownership Model of Annotation: The Ancient Greek Dependency Treebank. In *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories (TLT8). Milan, Italy*, pp. 5--15.

Bamman, D., Passarotti, M., Busa, R. & Crane, G. (2008). The annotation guidelines of the Latin Dependency Treebank and *Index Thomisticus* Treebank. The treatment of some specific syntactic constructions in Latin. In *Proceedings of LREC 2008. Marrakech, Morocco*.

Buchholz, S. & Marsi, E. (2006). CoNLL-X Shared Task on Multilingual Dependency Parsing. In *CoNLLX, SIGNLL, 2006*.

Busa, R. (1974-1980). *Index Thomisticus*. Stuttgart-Bad Cannstatt: Frommann-Holzboog.

Crane, G. (1991). Generating and Parsing Classical Greek. *Literary and Linguistic Computing*, vol. 6, n. 4, pp. 243--245.

Hajič, J., Panevová, J., Buráňová, E., Urešová, Z. & Bémová, A. (1999). *Annotations at analytical level: Instructions for annotators*. Technical report. Prague, Czech Republic: ÚFAL MFF UK.

Hajič, J., Panevová, J., Buráňová, E., Urešová, Z., Bémová, A., Kolárová-Reznícková, V. & Pajas, P. (2003). PDT-VALLEX: Creating a Large Coverage Valency Lexicon for Treebank Annotation. In *TLT 2003 – Proceedings of the Second Workshop on Treebanks and Linguistic Theories*. Vol. 9 of *Mathematical Modelling in Physics, Engineering and Cognitive Sciences*, pp. 57--68.

Halácsy, P., Kornai, A. & Oravecz, C. (2007). HunPos – an open source trigram tagger. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pp. 209--212.

Happ, H. (1976). *Grundfragen einer Dependenz-Grammatik des Lateinischen*. Goettingen: Vandenhoeck & Ruprecht.

Haug, D.T.T. & Jøndal, M.L. (2008). Creating a Parallel Treebank of the Old Indo-European Bible Translations. In *Proceedings of LaTeCH Workshop - LREC 2008. Marrakech, Morocco*, pp. 27--34.

Kingsbury, P. & Palmer, M. (2002). From Treebank to Propbank. In *Proceedings of LREC 2002. Las Palmas – Gran Canaria, Spain*.

Koch, U. (1993). *The Enhancement of a Dependency Parser for Latin*. Technical Report n° AI-1993-03, Artificial Intelligence Programs, University of Georgia.

Korhonen, A., Krymolowski, Y. & Briscoe, T. (2006). A Large Subcategorization Lexicon for Natural Language Processing Applications. In *Proceedings of LREC 2006. Genoa, Italy.*

Koster, C.H.A. (2005). Constructing a Parser for Latin. In *Computational Linguistics and Intelligent Text Processing, Lecture Notes in Computer Science*. Berlin-Heidelberg: Springer, pp. 48--59.

Mapelli, V. & Choukri, K. (2003). *Report on a (minimal) set of LRs to be made available for as many languages as possible, and map of the actual gaps*. ENABLER project internal report, Deliverable 5.1.

McGillivray, B. & Passarotti, M. (2009). The Development of the *Index Thomisticus* Treebank Valency Lexicon. In *Proceedings of LaTeCH-SHELT&R Workshop 2009. March 30th 2009, Athens, Greece.*

McGillivray, B., Passarotti, M. & Ruffolo, P. (2009). The *Index Thomisticus* Treebank Project: Annotation, Parsing and Valency Lexicon. *Traitement Automatique des Langues*, 50 (2), pp. 103--127.

Messiant, C., Korhonen, A. & Poibeau, T. (2008). LexSchem: A Large Subcategorization Lexicon for French Verbs. In *Proceedings of LREC 2008. Marrakech, Morocco*.

Minozzi, S. (2008). La costruzione di una base di conoscenza lessicale per la lingua latina: Latinwordnet. In *Studi in onore di Gilberto Lonardi*. Verona: Fiorini, pp. 243--258.

Mírovský, J. (2006). Netgraph: a Tool for Searching in Prague Dependency Treebank 2.0. In *TLT 2006. Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories. December 1-2, 2006, Prague, Czech Republic*, pp. 211--222.

Passarotti, M. (2004). Development and perspectives of the Latin morphological analyser LEMLAT. In A. Bozzi, L. Cignoni & J.L. Lebrave (Eds.), *Digital Technology and Philological Disciplines. Linguistica Computazionale*, XX-XXI, pp. 397--414.

Passarotti, M. & Dell'Orletta, F. (2010). Improvements in Parsing the *Index Thomisticus* Treebank. Revision, Combination and a Feature Model for Medieval Latin. In *Proceedings of LREC 2010. La Valletta, Malta*.

Ruppenhofer, J., Ellsworth, M., Petruck, M.R.L., Johnson, C.R. & Scheffczyk, J. (2006). *FrameNet II. Extendend Theory and Practice*. E-book available at http://framenet.icsi.berkeley.edu/index.php?option=com_wrapper&Itemid=126.

Schmid, G. (1994). *TreeTagger - a language independent part-of-speech tagger*. Available at http://www.ims.uni-stuttgart.de/Tools/DecisionTreeTagger.html.

Sgall, P., Hajičová, E. & Panevová, J. (1986). *The meaning of the sentence in its semantic and pragmatic aspects*. Dordrecht: Reidel.

Extraction of Semantic Relations as a Basis for a Future Semantic Database for Icelandic

Anna B. Nikulásdóttir, Matthew Whelpton

University of Iceland
Reykjavík, Iceland
abn@hi.is, whelpton@hi.is

Abstract

This paper describes work in progress on semi-automatically constructing a semantic database for Icelandic. The focus is on methods for the extraction of semantic relations used to collect material for the database. Established methodologies have largely focused on English. As Icelandic is a less-resourced language with much richer inflection than English, we must make adjustments to these established methods to address linguistic and sparse data problems. As a general principle, we aim to develop methodologies which will be viable for other less-resourced languages, with the support of open source tools.

1. Introduction

Semantic resources are already an established part of natural language processing (NLP) applications for dominant languages. Following the Princeton WordNet (Fellbaum, 1998) for English, many other languages have created their own WordNet-like resources (cf. Global Wordnet Association[1]). However, for less-resourced languages like Icelandic, the situation is much less favourable. Icelandic language technology (LT) has really only existed for about a decade (Rögnvaldsson et al., 2009) and despite a rich lexicographic tradition there have until now been no specially LT-oriented semantic resources. Fortunately, over the last decade, the prerequisites for the application of (semi)-automatic methods in developing such semantic resources have now been created: a PoS-tagger, a shallow parser and a lemmatizer (Loftsson, 2008; Loftsson and Rögnvaldsson, 2007; Ingason et al., 2008). In 2007, a pilot study was run to extract semantic relations from an Icelandic dictionary (Nikulásdóttir and Whelpton, 2009; Nikulásdóttir, 2007); following the success of this study[2] and parallel developments in the field, a work-package for the creation of a database of semantic relations was incorporated into a major new project in Icelandic LT: in 2009, the project *Viable Language Technology beyond English - Icelandic as a Test Case* received a three year Grant of Excellence from the Icelandic Research Fund (RANNÍS)[3]. One central aim of the project is to experiment with known methods for the extraction of semantic relations and investigate how well they can be applied to Icelandic, given two significant characteristics of the language: (a) Icelandic is a highly inflected language; (b) there are as yet no large corpora for the language. Most of the research in this area has focused on English which differs from Icelandic in both respects. To as great an extent as possible, we aim to exploit and develop methodologies which will be generally viable for other less-resourced languages with the support of open source tools. It should be noted that a preliminary motivation for our work is the desire to build a database of "native" semantic relations for Icelandic, i.e. to extract information from Icelandic resources, reflecting distributional and collocational properties of Icelandic lexemes (cf. also DanNet, (Pedersen et al., 2009)) rather than to fit the Icelandic data to an external model, for instance by importing the ontological structure of the English WordNet by translation (cf. e.g. (Fernández-Montraveta et al., 2008)). Whether, and to what extent, these two methodologies produce significantly different results remains an open question but our aim is to contribute to the ultimate evaluation of such differences by contributing a native ontology for Icelandic — and to test the extent to which such an aim is achievable for languages like Icelandic with relatively limited resources.

In this paper we describe the context of our work and report our first experiments with the extraction from text of semantic information on nouns. We want to stress that we describe work in progress and that no formal evaluation data is available yet. In section 2 we review the inflectional properties of Icelandic nouns and describe the currently-available corpus. The final design of the Icelandic semantic database has not yet been established but in section 3 we consider several important issues relating to the structure of wordnets. Following the hybrid methodology developed in recent years (Cederberg and Widdows, 2003; Cimiano et al., 2005; Pantel and Pennacchiotti, 2008) we exploit different methods, both pattern-based and statistical, to extract semantic information. We describe those in sections 4 and 5. In section 6 some possibilities for validation and expansion of results are discussed, concluding with an assessment of prospects for future work.

2. A Corpus of Icelandic

Icelandic is a highly inflected language, which makes a PoS-tagged and lemmatized corpus essential for any further automatic processing of Icelandic text. Nouns in Icelandic inflect for case, number and gender, and so does the cliticised definite article[4]. An example of a lemmatized

[1] `http://www.globalwordnet.org`

[2] For reasons of space, we are unable to review this study here. Those interested in a detailed overview of this study and of the relations extracted should consult (Nikulásdóttir and Whelpton, 2009) and (Nikulásdóttir, 2007).

[3] `http://iceblark.wordpress.com`

[4] Icelandic does not have an indefinite article

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 33–39

PoS-tagged noun *börnunum* '(to) the children' is:

TOKEN	POS-TAG	LEMMA
börnunum	nhfþg	barn

The PoS-tag stands for noun (n), neuter (h), plural (f), dative (þ), and definite article (g). Additionally, a special tag for proper nouns is included in the tagset for nouns. At the moment a balanced PoS-tagged, lemmatized corpus, MIM, is being developed at the Árni Magnússon Institute for Icelandic Studies (Helgadóttir, 2004). The planned size of this corpus is about 25 million tokens, a reasonable size but still not especially large. For our present studies we use a subset of a preliminary version of this corpus (hereafter, SubMIM) containing 8.8 million tokens, including punctuation marks etc. The source of this data is mainly newspaper texts (*Morgunblaðið*), but further texts come from a public science web portal at the University of Iceland (*Vísindavefurinn*[5]), reports from Icelandic ministries, and from a medical Journal (*Læknablaðið*). Two versions of SubMIM are used for different automatic extraction methods: (a) the basic PoS-tagged and lemmatized version is used for the statistical methods and (b) a shallow-parsed version without lemmata is used for the pattern-based methods. The tagging, lemmatizing and parsing was performed using the PoS-tagger *IceTagger*, the lemmatizer *Lemmald*, and the shallow parser *IceParser*, all included in the open source IceNLP-toolkit[6]. The parsed version was parsed without using the option of marking grammatical functions, but another version including those tags will be useful for some further experiments (see section 5). The corpus is fully automatically processed and no manual correction has yet been performed.

Nevertheless, for the foreseeable future, Icelandic LT faces a serious sparse data problem compared to English; although we will supplement SubMIM (and MIM when it is complete) with web-based data, this problem will remain. Work on the lexicon needs very large corpora: even the British National Corpus[7] (BNC) with its 100 million tokens has been shown to be too small for a broad coverage statistical analysis of word occurrences (Kilgarriff and Grefenstette, 2003). We are encouraged by the development in Leipzig, Germany, of a 250 million token corpus of Icelandic (Hallsteinsdóttir et al., 2007), collected from all `.is` domains, and we hope to include this in our dataset. However, this still puts us far behind the corpus of billions of words which is now being developed for English through web crawling (Pomikálek et al., 2009).

3. Relations in wordnets

The primary relations between nouns in WordNet are synonymy and hyponymy (Fellbaum, 1998). Synonymous or near-synonymous words build synsets as labels of concepts. Other relations like hyponymy or meronymy hold between synsets, exceptionally other relations hold between words, like antonymy. This organisation is followed by all wordnets within the EuroWordNet[8] scheme, even if they include more relations, such as for instance "involved agent" (*violin* INVOLVED_AGENT *violinist*) or "role agent" (*passenger* ROLE_AGENT *journey*)[9]. DanNet (Pedersen et al., 2009) extends the EuroWordNet template with for instance the relations "concerns" (*goal* CONCERNS *sport*) and "has_hypernym ortho" (*road-side tree* HAS_HYPERNYM_ORTHO *tree*). All these databases have in common that all links between words and synsets are labelled with a defined relation. However, many NLP applications that use wordnets could benefit from a more dense structure of arcs, including nonclassical relations (Morris and Hirst, 2004; Zesch and Gurevych, 2009). There is indeed a plan to extend WordNet with directed, weighted arcs between synsets, that correspond to the "evoking" relation, i.e. how strongly one synset evokes another one according to a human (Boyd-Graber et al., 2006). One interesting resource in the family of semantic networks is SALDO, a Swedish Associative Thesaurus (Borin and Forsberg, 2009). It is strictly hierarchical, but relies on loosely characterized associative relations rather than classical semantic relations.

The structure of the semantic database for Icelandic LT has not yet been fixed but the aim is that the structure should be as data-driven as possible, i.e. as much as possible of the extracted semantic information should find its way into the database and thus it should not be limited to classical semantic relations.

4. Pattern-based methods for relation extraction

Pattern-based methods for relation extraction have been widely used since the publication of (Hearst, 1992). The essence of these methods is to use seed words known to be in a certain semantic relation to harvest syntactic and/or lexico-syntactic patterns indicating the relation in question. As an example, Hearst discovered several patterns for the extraction of hypernymy with the seed words *England - country*, e.g. the pattern `NP {, NP}* {,} and other NP` (Hearst, 1992, p. 541) from phrases like *England and other countries*. Such patterns are known to be reliable in the extraction of hypernyms (Cimiano et al., 2005) and have also been used for the extraction of meronyms (part-whole relations) (Berland and Charniak, 1999; Girju and Badulescu, 2006). Using few seed words requires a large corpus, since one needs (a) the words to occur several times togehter and (b) ideally in different patterns, but reliable patterns have been shown to be low frequency in English (Cimiano et al., 2005). As an experiment to deal with the sparse data problem we developed a method called validation of most common syntactic patterns. The motivation for this method is twofold: (a) to recognize as many patterns as possible from sparse data (b) without having to predefine the relations that are to be extracted. The method includes four steps: (i) extract syntactic patterns with their actual realizations from the corpus according to some predefined criteria - in this case every noun and prepositional

[5] `http://www.visindavefur.is`
[6] `http://sourceforge.net/projects/icenlp`
[7] `http://www.natcorp.ox.ac.uk`
[8] `http://www.illc.uva.nl/EuroWordNet`
[9] These examples are adapted and translated from DanNet, `http://wordnet.dk/dannet/lang`

phrase in the corpus; (ii) with the help of a special GUI, loosely validate the most common patterns according to possible indication of some semantic relation - in this case every pattern occurring at least 10 times in the corpus; (iii) combine similar patterns using an edit distance algorithm and regular expressions; (iv) extract related words from the corpus. In the following we will discuss each step in more detail.

4.1. Extraction of patterns

For the extraction of patterns the shallow-parsed version of SubMIM was used (see section 2). The aim was then to extract all noun phrases and prepositional phrases that could possibly include semantically related nouns and/or adjectives. Thus the criteria for the extraction of phrases were (a) all noun phrases including more than one noun or at least one noun and at least one adjective, (b) all coordinated noun phrases or adjective phrases, (c) all prepositional phrases including more than one noun, (d) all connected noun and prepositional phrases. Always the longest possible chain of phrases was extracted.
The following example shows how IceParser analyzes the phrase *feitur og kryddaður matur* 'greasy and spicy food', and the pattern extracted from the parser output:

IceParser output:
```
[NP[APs[AP feitur lkensf AP][CP og
c CP][AP kryddaður lkensf AP]APs]
matur nken NP]
```
Extracted pattern:
```
[NP[APs[AP lensf][CP og c][AP
lensf]] nen]
```

Here "NP" denotes a noun phrase, "APs" a sequence of adjective phrases, "AP" an adjective phrase and "CP" a coordinating conjunction. The PoS-tag starting with "l" stands for adjective, the one starting with "n" for noun and "c" stands for conjunction. We ignore the tag for gender so the "k" (for masculine) in the PoS-tags for nouns and adjectives is removed. In general, no words are included in the patterns except conjunctions and prepositions, since they can help identify semantic relations. In this case *og* 'and' is retained.
About 370,000 different patterns were extracted in this way. Of these, about 94,000 occur more than once and about 5,300 more than ten times in SubMIM. Only those patterns occurring more than ten times were validated. The reason for this high number of patterns is the large tagset of Icelandic, which contains about 700 tags. This high granularity is probably not necessary and has in many cases been ignored in the process of merging the patterns. The final tests will show which tags need to be kept in the patterns and which can be ignored.

4.2. Validation of patterns

For the validation of the most common patterns a simple GUI was developed. Selecting a pattern lists all instances of the pattern plus frequency. If some semantic relatedness is salient between nouns and/or adjectives in a majority of the instances, the validator assigns one predefined category to the pattern. The categories were defined after a first look at the most common patterns and they are thought of as a rough partition of the patterns, which are to be tested further through relation extraction. Five of the categories refer to the syntactic structure of a pattern: genitive construction, attributive construction (adjective(s) plus noun), and coordinated nouns, adjectives or proper nouns; and three refer to a semantic relation: superordinate, location and role. If none of these categories apply to a pattern, but the validator thinks it might indicate some semantic relatedness, the category "other" is chosen. Less than the half of all examined patterns (i.e. 2,275 of 5,268 patterns) were classified as possibly indicating a semantic relation.
Since the GUI only takes results from the pattern extraction as input and does not process this input in any way, it is totally language independent. The final version of the GUI will be open source, as is consistent with our general aim of building up a shared set of methodologies and tools for less-resourced languages.

4.3. Merging of patterns

As the number of positively validated patterns was much higher than expected, they must be simplified and merged in some way. The tag for number was removed from the patterns and for pronouns and adjectives all tags except the ones marking the word class were removed. The patterns are then merged and generalized using the minimum edit distance algorithm and then further merged using regular expressions. The method for the computation of minimum distance between patterns and their generalization was adaptded from (Ruiz-Casado et al., 2005). Given for example two patterns expressing a genitive construction `[NP nn][NP ne]` (nominative noun - indefinite genitive noun) and `[NP nn][NP neg]` (nominative noun - definite genitive noun), the distance is 1, due to the single difference of definiteness `ne` vs. `neg`. The generalization algorithm then makes one pattern out of the two: `[NP nn][NP ne|neg]`. A standard regular expression for this pattern would be `[NP nn][NP neg?]`, and often several general patterns can be unified in one regular expression. This has been done for several of the identified pattern categories and shows considerable reduction in the number of patterns. The original number of patterns including genitive constructions was 384, but through the simplifying and merging process they have been reduced to 71 patterns. As stated above, the patterns may be refined after the final testing.

4.4. Extraction of relations

Since we are presenting work in progress, no final evaluation data is available yet. However, we have extracted relations from SubMIM based on patterns marked as *superordinate, coordinated nouns*, and *genitive construction*. Only one pattern was used to extract superordinates or hypernyms, an Icelandic equivalent to one of the patterns introduced by (Hearst, 1992): `NP (,NP)*, and|or other NP` (in Icelandic nine different morphological forms of *other* can be used in this pattern). All in all 369 hypernyms were extracted, which is a rather low number. Cederberg and Widdows (2003) extracted 513

hypo-/hypernym pairs from approximately the first 430,000 words from the BNC, using the six patterns reported by (Hearst, 1998). Finding more patterns in Icelandic indicating hypernymy would possibly increase the number of extracted hypernyms. The results are however reliable in that most word pairs express some kind of a sub-/superordinate relation, although how many are really taxonomic hypo-/hypernym pairs still needs to be evaluated.

Genitive constructions can express many different relations between the involved nouns. The main interest for the extraction of semantic relations is the part-whole relation often expressed by a genitive construction (Berland and Charniak, 1999; Girju and Badulescu, 2006; Pantel and Pennacchiotti, 2008). The problem is that it is impossible to judge from the lexical-syntactic pattern alone whether it expresses a part-whole relation or not. Berland and Charniak (1999) extract parts of a given seed word representing the whole, e.g. *car*. They then filter out words having the suffixes *ness, ing* and *ity*, since those tend to express qualities rather than parts. Finally they use a probability measure to rank their results, which gave them 55% accuracy with the top 50 words. Another approach was taken by (Girju and Badulescu, 2006), where they used WordNet to ontologize the extracted word pairs. With a training set and a learning algorithm, information on the ontological category of each word in the pair was used to deduce the likelihood of a new part-whole relation between the pair.

For the moment we call the general relation expressed by a genitive construction the relation of properties. Even without refining this relation, it can give valuable information. The results already reveal polysemy of terms, not necessarily accounted for in dictionaries: *cod* as "fish" (*the stomach of the fish*) or as a "product" (*the market price of the fish*); *house* as a "building" (*the roof of the house*), as a "property" (*the running of the house*), as a "theatre" (*the house consultant [house=theatre]*), or as a "restaurant/pub" (*the house band*). It is also possible to categorize relations including the action nouns registered as such in existing lexicographic resources, following the filtering of *ness, ing,* and *ity* proposed by Berland and Charniak (1999). The next step in processing this material will involve validation by human assessors and by statistical testing, which could improve results.

Noun-coordination information has been used to collect co-hyponyms (Roark and Charniak, 1998; Caraballo, 1999) and Cederberg and Widdows (2003) use it to extend results from pattern-based hypernym extraction. Then a hyponym is used as a seed word to extract potential co-hyponyms, since coordinated nouns often belong to the same hierarchy level in a hypernym hierarchy. We will discuss the potential use and problems of this pattern in section 6.2.

5. Semantic relatedness and clustering

We adopt a broad conception of semantic relatedness, by which words that belong to the same semantic domain or topic are semantically related; this broad definition is in line with (Manning and Schütze, 1999, p. 296), though they use the term "semantic similarity". A thorough discussion of semantic similarity and semantic relatedness can be found in (Zesch and Gurevych, 2009) and (Turney, 2006). An established way to compute semantic relatedness is to use the cosine similarity measure between two vectors. The vectors can be built by counting cooccurrences of words of interest - in our case nouns - with the most frequent content-bearing words of the language (Manning and Schütze, 1999; Cederberg and Widdows, 2003) or for example according to their cooccurrence with verbs in certain grammatical functions such as subject or direct object (Weeds, 2003; Cimiano, 2006). We intend to exploit both methods and first experiments have been conducted with the former method, based on cooccurrences with frequent content-bearing words. We have used the tagged and lemmatized version of SubMIM, though, for languages without access to such a resource, it is also possible to perform this co-occurrence analysis on a clean corpus (Bullinaria, 2008). The resulting cooccurrence matrix from SubMIM has about 11,300 rows representing nouns, including proper nouns, and 900 columns representing frequent content-bearing words. The content-bearing words were obtained from a word frequency list for Icelandic. This list was filtered for stop words and compared with the most frequent words in SubMIM. Several words from the common frequency list were not high frequency in SubMIM and were exchanged with corpus specific high frequency words. The top 100 words from the resulting list of 1,000 high frequency content-bearing words were then deleted (see (Manning and Schütze, 1999, p. 302)), leaving a list of 900 words used for the cooccurrence analysis. With the correction of the automatic lemmatization and a larger corpus we expect both a larger number of results and an improvement in the computation of semantic relatedness. Nevertheless we used these results in a preliminary experiment on clustering nouns from SubMIM with respect to semantic relatedness. To reduce noise in the clustering data, very frequent and very rare words were eliminated (cf.(Dhillon and Modha, 2001)). Excluding words occurring in collocation with more than 15% of the 900 content-bearing words and words that are not counted more often than 18 times[10] (18 is 2% of the column size of the matrix), the nouns to be clustered were reduced to 7,871 nouns. The first choice in exploring new data with clustering is often the *k*-means algorithm, an elementary but very popular approximation method (Duda et al., 2001, p. 526). In this algorithm a set of initial cluster centers is randomly defined. The data elements are then assigned to the closest center, according to some distance measure - here the cosine similarity measure - and then the center of each cluster is recomputed. The procedure of assigning data elements and recomputing centers is then repeated until some stopping criterion is reached.

The first observation made after running *k*-means on our data was that just like the Euclidean distance measure most often used in *k*-means (Manning and Schütze, 1999, p. 516), clustering using cosine similarity results in singelton clusters. At the same time, several clusters were very large and as one would expect, they normally have poor cluster quality. Cluster quality was computed by summing up the similarity values of all members of a cluster with

[10] With more data this threshold should be set higher, normally higher frequency is needed for lexical statistical analysis (Kilgarriff and Grefenstette, 2003).

Extracted word pair	Pattern	Similarity
líkami - fruma ('body' - 'cell')	genitive construction	0.7435
námskrá - grunnskóli ('curriculum' - 'elementary school')	genitive construction	0.5326
morð - glæpur ('murder' - 'crime')	superordinate	0.5165
þröstur - fugl ('thrush' - 'bird')	superordinate	0.4923
þorskur - botnfiskur ('cod' - 'bottom dweller')	superordinate	0.4921
þróun - líftæknifyrirtæki ('development' - 'biotechnology company')	superordinate	0.4529
frumskógur - málari ('jungle' - 'painter')	genitive construction	0.2676

Table 1: Results from a pattern-based method validated with a semantic relatedness measure

its mean vector and dividing by number of members, thus getting the average similarity value for the cluster (Dhillon and Modha, 2001). To force the algorithm to make clusters of reasonable size, i.e. not too small and not too large, an elementary validation process was implemented. It examines a finished *k*-means partition and deletes all clusters that have less than four members. If a cluster has more than some MAX members, it is split in two clusters. The validation process thus can change the initial *k* number of clusters. After the validation *k*-means is run again. This is repeated until no change is made to the partition in the validation process. Results on SubMIM using MAX=200 and initial number of clusters *k*=32 show 60 clusters containing from 23 to 184 words. We found that 46 of the 60 clusters can be characterized by a subsuming concept, whereas 14 cannot. These concepts have different ontological status so that a one-to-one mapping in an ontology is not possible. There are traditional scientific domains like BIOCHEMISTRY (*hormone, secretion, metabolism*), BIOLOGY, and METEOROLOGY, domains from public discourse like FINANCES (*privatisation, tax environment, monopoly*) POLITICS, and GLOBALISATION, concrete things like HOUSE (*bathroom, master bedroom, laundry room*), VEHICLE, as well as domains containing mostly proper nouns like FOOTBALLERS (*lampard, gerrard, thierry*), MUSIC/MUSICIANS, and PROPER NOUNS in general.

6. Combination of results

In order to improve results gained from different extraction methods, it is reasonable to combine them and so be able to extend and/or validate results. A word pair extracted with one method can be supported or not supported through results from another method. As shown by Cimiano et al. (2005), different pattern-based methods with different resources can give better results than just using one resource and Cederberg and Widdows (2003) use latent semantic analysis and noun coordination information to improve results of automatic hyponymy extraction. Pantel and Pennacchiotti (2008) extend their pattern-based method with a measure of pattern and instance reliability. With hybrid methods like this it should be possible to reduce the human validation effort which will be necessary at some point during the building of the semantic database.

6.1. Validation

Many studies on extraction of semantic relations from English text use WordNet to validate the results, e.g. (Pantel and Pennacchiotti, 2008). Neither a WordNet-like resource nor a semantically annotated corpus is available for Icelandic, so some kind of cross-validation between the extraction methods will be used for validation. Like Cederberg and Widdows (2003) we use semantic relatedness values to verify results from pattern extraction. Although we still need to correct and extend data used for the computation of semantic relatedness (see section 5), it seems to be a valuable measure on extracted hypernyms and word pairs from genetive constructions, as shown in table 1. But since semantic relatedness in our sense means belonging to the same topic or domain, incorrectly extracted taxonomic relations like hypernymy can still get high similarity values, as shown for *development - biotechnology company* in table 1. Cederberg and Widdows (2003) achieved a 30% reduction in error using this kind of semantic relatedness to verify extracted hypernyms, precision improved from 40% to 58%. It is a matter of further evaluation to see if we are able to improve our results in this way, despite these findings.

In order to be able to evaluate the results systematically, more data on relatedness is needed since most of the extracted word pairs are not found in the present similarity matrix. For very low frequency words we may not get this data, but with the use of a larger corpus a considerable extension of similarity measures should be possible.

6.2. Extension

As Cederberg and Widdows (2003) showed, it is possible to extend results of hypernym extraction by extracting coordinated nouns of a hyponym. After extracting e.g. *cloves* as a hyponym of *spice* from *...sugar, honey, grape, must, cloves and other spices...* the hyponyms of *spice* can be extended with *nutmeg, cinnamon,* and *coriander* by extract-

ing . . . *nutmeg or cinnamon, cloves or coriander*. They also point out that a seed word like *cloves* has to be chosen carefully, so that different meanings of a seed word don't lead to extraction of words not related to the hypernym in question. We experienced this problem in our tests where we wanted to extract further co-hyponyms of *fugl* 'bird' as a hyponym of *dýralíf* 'animal life'. The word *fugl* can also mean *birdie* (from the domain of golf) and extracted co-hyponyms included *forgjöf* 'handicap' and *par* 'par'.

Another problem concerns the level of the hypernym. One result of the hypernym extraction was *þorskur* 'cod' IS-A *botnfiskur* 'bottom dweller'. This is correct, and so was the extraction of co-hyponyms including various sorts of fish. However, the hypernym here is too narrow: not all of the "co-fishes" are bottom dwellers so that they need to be subsumed under the broader hypernym *fish*.

One question that needs to be further investigated is if certain kind of properties expressing semantic features can be used to extend results. As an example, can a word known to have the property *beginning* be extended to having the property *end*? In examining some extracted words having the property *beginning*, differences regarding modality became apparent. While *a century, an aria* and *a book* all have a certain beginning and an end, there is no necessary or predefined end to *a marriage, a town*, or *the world*, just a potential one.

7. Conclusions and future work

We have addressed several methods for the extraction of semantic relations and given some provisional results in applying these methods to Icelandic. Our current work is based on a small corpus, while further corpus development is taking place. The next steps include thorough testing and evaluation of these methods as well as the implementation of further methods. Together, these should yield a considerable amount of lexical-semantic information about Icelandic nouns. We then face the challenge of combining this information with existing lexical resources, as well as the relations already extracted from the Icelandic dictionary, to build the basis of a semantic database for Icelandic. Throughout this process, we are also guided by the long-term aim of mapping this Icelandic database to WordNet (see e.g. (da Silva et al., 2008)), which has practical ramifications for organization of the resource.

We hope that our efforts will benefit not only the development of Icelandic LT but also other less-resourced languages, by identifying effective methods for addressing the sparse data problem and by contributing necessary open source tools.

8. References

Matthew Berland and Eugene Charniak. 1999. Finding Parts in Very Large Corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 57–64.

Lars Borin and Markus Forsberg. 2009. All in the Family: A Comparison of SALDO and WordNet. In Bolette Sandford Pedersen, Anna Braasch, Sanni Nimb, and Ruth Vatvedt Fjeld, editors, *Proceedings of the NODALIDA 2009 Workshop Wordnets and other Lexical Semantic Resources - between Lexical Semantics, Lexicography, Terminology and Formal Ontologies*, volume 7 of *NEALT Proceedings Series*, pages 7–12, Odense, Denmark.

Jordan Boyd-Graber, Christiane Fellbaum, Daniel Osherson, and Ropert Schapire. 2006. Adding Dense, Weighted Connections to WordNet. In Petr Sojka, K.-S. Choi, Christiane Fellbaum, and P. Vossen, editors, *Proceedings of the GWC*, pages 29–35.

John A. Bullinaria. 2008. Semantic Categorization Using Simple Word Co-occurrence Statistics. In M. Baroni, S. Evert, and A. Lenci, editors, *Proceedings of the ESSLLI Workshop on Distributional Lexical Semantics*, pages 1–8, Hamburg, Germany. ESSLLI.

Sharon Caraballo. 1999. Automatic Construction of a Hypernym-Labeled Noun Hierarchy from Text. In *Proceedings of ACL*, pages 120–126.

Scott Cederberg and Dominic Widdows. 2003. Using LSA and Noun Coordination Information to Improve the Precision and Recall of Automatic Hyponymy Extraction. In *Proceedings of the International Conference on Natural Language Learning (CoNLL)*, pages 111–118.

Philipp Cimiano, Aleksander Pivk, Lars Schmidt-Thieme, and Steffen Staab. 2005. Learning Taxonomic Relations from Heterogenous Evidence. In Paul Buitelaar et al., editor, *Ontology Learning from Text: Methods, Evaluation and Applications*, volume 123 of *Frontiers in Artificial Intelligence*.

Philipp Cimiano. 2006. *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*. Springer.

Bento Carlos Dias da Silva, Ariani Di Felippo, and Maria das Graças Volpe Nunes. 2008. The Automatic Mapping of Princeton WordNet Lexical-Conceptual Relations onto the Brazilian Portugese WordNet Database. In Nicoletta Calzolari et al., editor, *Proceedings of LREC2008*, pages 1535–1541.

Inderjit Dhillon and Dharmendra S. Modha. 2001. Concept Decompositions for Large Sparse Text Data using Clustering. *Machine Learning*, 42(1):143–175.

Richard O. Duda, Peter E. Hart, and David G. Stork. 2001. *Pattern Classification*. John Wiley, New York, Chichester, etc.

Christiane Fellbaum, editor. 1998. *WordNet. An Electronic Lexical Database*. MIT Press, Cambridge Mass., London.

Ana Fernández-Montraveta, Gloria Vázquez, and Christiane Fellbaum. 2008. The Spanish Version of WordNet 3.0. In Angelika Storrer, Alexander Geyken, Alexander Siebert, and Kay-Michael Würzner, editors, *Text Resources and Lexical Knowledge*, pages 175–182. Mouton de Gruyter.

Roxana Girju and Adriana Badulescu. 2006. Automatic Discovery of Part-Whole Relations. *Computational Linguistics*, 32(1):83–134.

Erla Hallsteinsdóttir, Thomas Eckart, Chris Biemann, Uwe Quasthoff, and Matthias Richter. 2007. Íslenskur Orðasjóður - Building a Large Icelandic Corpus. In *Proceedings of NODALIDA-07*, Tartu, Estonia.

Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of COLING-92*, pages 539–545, Nantes.

Marti A. Hearst. 1998. Automated Discovery of WordNet Relations. In Christiane Fellbaum, editor, *WordNet. An Electronic Lexical Database*. MIT Press, Cambridge Mass., London.

Sigrún Helgadóttir. 2004. Mörkuð íslensk málheild [A Tagged Icelandic Corpus]. In *Samspil tungu og tækni*, pages 65–71. Ministry of Education, Science and Culture, Reykjavík.

Anton Karl Ingason, Sigrún Helgadóttir, Hrafn Loftsson, and Eiríkur Rögnvaldsson. 2008. A Mixed Method Lemmatization Algorithm Using a Hierarchy of Linguistic Identities (HOLI). In Bengt Nordström and Aarne Ranta, editors, *Advances in Natural Language Processing*, volume 5221 of *Lecture Notes in Computer Science*, pages 205–216, Berlin. Springer.

Adam Kilgarriff and Gregory Grefenstette. 2003. Web as Corpus. *Computational Linguistics*, 29(3):1–15.

Hrafn Loftsson and Eiríkur Rögnvaldsson. 2007. IceParser: An Incremental Finite-State Parser for Icelandic. In Joakim Nivre, Heiki-Jaan Kaalep, Kadri Muischnek, and Mare Koit, editors, *Proceedings of the 16th Nordic Conference on Computational Linguistics (NODALIDA)*, pages 128–135, Tartu, Estonia.

Hrafn Loftsson. 2008. Tagging Icelandic Text: A Linguistic Rule-Based Approach. *Nordic Journal of Linguistics*, 31(1):47–72.

Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge Mass., London.

Jane Morris and Graeme Hirst. 2004. Non-classical Semantic Relations. In *Workshop on Computational Lexical Semantics, Human Language Technology Conference of the North American Chapter of the ACL*, pages 46–51, Boston, MA.

Anna Björk Nikulásdóttir and Matthew Whelpton. 2009. Automatic Extraction of Semantic Relations for Less-Resourced Languages. In Bolette Sandford Pedersen, Anna Braasch, Sanni Nimb, and Ruth Vatvedt Fjeld, editors, *Proceedings of the NODALIDA 2009 Workshop Wordnets and other Lexical Semantic Resources - between Lexical Semantics, Lexicography, Terminology and Formal Ontologies*, volume 7 of *NEALT Proceedings Series*, pages 1–6, Odense, Denmark.

Anna Björk Nikulásdóttir. 2007. Autmatische Extrahierung von semantischen Relationen aus einem einsprachigen isländischen Wörterbuch [Automatic Extraction of Semantic Relations from a monolingual Icelandic Dictionary]. Master's thesis, University of Heidelberg.

Patrick Pantel and Marco Pennacchiotti. 2008. Automatically Harvesting and Ontologizing Semantic Relations. In Paul Buitelaar and Philipp Cimiano, editors, *Ontology Learning and Population: Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*. IOS Press.

Bolette Sandford Pedersen, Sanni Nimb, Jörg Asmussen, Nicolai Hartvig Sörensen, Lars Trap-Jensen, and Henrik Lorentzen. 2009. DanNet: the Challenge of Compiling a Wordnet for Danish by Reusing a Monolingual Dictionary. *Language Resources and Evaluation*, 43:269–299.

Jan Pomikálek, Pavel Rychlý, and Adam Kilgarriff. 2009. Scaling to Billion-plus Word Corpora. In *Advances in Computational Linguistics. Special Issue of Research in Computing Science*, volume 41, Mexico City.

Brian Roark and Eugene Charniak. 1998. Noun-phrase Co-occurrence Statistics for Semi-automatic Lexicon Construction. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*.

Eiríkur Rögnvaldsson, Hrafn Loftsson, Kristín Bjarnadóttir, Sigrún Helgadóttir, Anna Björk Nikulásdóttir, Matthew Whelpton, and Anton Karl Ingason. 2009. Icelandic Language Resources and Technology: Status and Prospects. In Rickard Domeij, Kimmo Koskenniemi, Steven Krauwer, Bente Maegaard, Eiríkur Rögnvaldsson, and Koenraad de Smedt, editors, *Proceedings of the NODALIDA 2009 Workshop Nordic Perspectives on the CLARIN Infrastructure of Language Resources*, volume 5 of *NEALT Proceedings Series*, pages 27–32, Odense, Denmark.

Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. 2005. Automatic Extraction of Semantic Relationships for WordNet by means of Pattern Learning from Wikipedia. In A. Montoyo R. Munos and E. Métais, editors, *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB 2005)*, volume 3513 of *Lecture Notes in Computer Science*, pages 67–79, Alicante, Spaine, June. Springer.

Peter D. Turney. 2006. Similarity of Semantic Relations. *Computational Linguistics*, 32(3):379–416.

Julie Weeds. 2003. *Measures and Applications of Lexical Distributional Similarity*. Ph.D. thesis, University of Sussex.

Torsten Zesch and Iryna Gurevych. 2009. Wisdom of Crowds versus Wisdom of Linguistics - Measuring the Semantic Relatedness of Words. *Natural Language Engineering*, 16(1):25–59.

Nganasan – Computational Resources of a Language on the Verge of Extinction

István Endrédy[1], László Fejes[2], Attila Novák[1], Beatrix Oszkó[2], Gábor Prószéky[1], Sándor Szeverényi[2], Zsuzsa Várnai[2], Beáta Wagner-Nagy[2]

[1] MorphoLogic
5, Kardhegy u., 1116 Budapest, Hungary
{endredy, novak, proszeky}@morphologic.hu

[2] Department of Finno-Ugric and Historical Linguistics, Linguistics Institute, Hungarian Academy of Sciences
33, Benczúr u., 1063 Budapest, Hungary
{fejes, oszko, szeverenyi, varnai, wnbea}@nytud.hu

Abstract

This paper describes the creation and dissemination of computational resources for Nganasan: annotated corpora, morphological analyzer, morphological generator and the development of a website where all of them are available for a wider public. The morphology and especially the phonology of the language are so complex that the implementation of the morphological tools was a real challenge.

1. Introduction

Nganasan belongs to the Northern branch of Samoyedic languages: it is an endangered language spoken in Northern Siberia, in Russia. It is a language on the verge of extinction, namely, it is spoken by less than 500 people most of whom are middle aged or older, and due to the Russian minority policy Russian is the language of teaching in schools for Nganasans. Therefore, it has been an urgent scientific task to provide documentation for the language. Similar work has been done earlier for Sami languages as well (http://giellatekno.uit.no/english.html).

2. Nganasan Root and Suffix Dictionaries

The Nganasan side of the Russian–Nganasan dictionary of Kost'erkina et al. (2001) has been elaborated and converted to the phonemic transcription made up of Roman characters. In the course of building morphological tools further roots have been added to the system, which currently contains approximately 4200 roots. The team also has provided category labels for each item, which was missing from the source, e.g. harmonic features of nouns that cannot be seen on the surface, features of verbal aspect, or irregularity.

During the preparation of the root dictionary, we also started to describe the suffixes of Nganasan in a formal manner. The first step of this was the creation of a list of the suffixes that contained the underlying phonological form of each suffix together with its category label plus a feature that indicates which morphological root form the suffix can attach to. We used the following model to describe the language: we hypothesize that there are three allomorphs for each root morpheme (out of which two or all three might have the same form), and suffixes are sorted into three groups depending on which root allomorph they attach to. Some suffixes display ambiguous behavior: they can attach to two of the root allomorphs. There are some vowel symbols in the underlying phonological representation that mark vowels that vary according to the vowel harmony rules of Nganasan: in the case of suffixes the quality of these vowels depends on the harmonic features of the root they are attached to. The first suffix list we compiled contained additional information for derivational suffixes: we gave the category label for the root it attaches to and the category label for the derived form as well.

3. The Complexity of Nganasan Morpho-phonology

This language displays many special phonological and morpho-phonological features, including the phenomena of vowel harmony and two types of consonant gradation. Nganasan gradation does not depend on the morphological make-up of the word: the only factor at play is syllable structure. Syllable boundaries and morph boundaries hardly ever coincide. In the case of short suffixes (made-up of 1 segment), it is possible that even non-adjacent morphs belong to the same syllable. There are additional factors that are needed for the description of gradation: (i) whether the syllable in question is closed (ii) whether the previous syllable is closed (iii) the length of the vowel in the previous syllable (iv) whether the syllable in question is odd or even numbered in the word. Gradation also combines with other alternations in the language: vowel harmony, degemination, root alternations and suffix alternations (as a result of which a one-syllable long suffix can easily have fourteen different allomorphs).

To illustrate the complexity of the above outlined system let us look at the allomorphs of a verbal suffix (Narrative Mood, Nominative–Accusative). Let us see the underlying

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 41–44

representation of the morpheme *hA$_2$nhV*. It has twelve allomorphs: *banghu, bjanghy, bambu, bjamby, bahu, bjahy, hwanghu, hjanghy, hwambu, hjamby, hwahu, hjahy.* These allomorphs are regularly produced from the underlying representation, which undergoes the following phonological processes. The harmonic vowel A_2 surfaces as *a* or *ja* as a result of root dependent roundness harmony, and *a* diphthongizes to *wa* when it follows an *h.* Roots are sorted into lexical classes depending on their harmonic features. This feature must be marked in the lexicon as it is totally arbitrary. Some roots may belong to more than one class, as they display vacillating behavior. The harmonic vowel *V* can surface as *u, y, ü* or *i,* its behavior being regulated by roundness and frontness harmonies (in the suffix being discussed it can only surface as *u* or *y,* however, as there is a back vowel (*a, ja, wa*) in the previous syllable in every case). The consonant *h* appears as *h* in strong grade and as *b* in weak grades. The consonant cluster *nh* surfaces as (i) *ngh* in strong grade or if it undergoes the so-called "nunnation effect"[1], as (ii) *h* in rhythmical weak grade, as (iii) *mb* in syllabic weak grade. (A nasal consonant assimilates in place of articulation to the following consonant, and it disappears in rhythmical weak grade unless there is an immediately preceding nasal on the consonantal tier: this latter phenomenon is called nunnation.) An obstruent in the onset position is in strong grade (i) in even-numbered open syllables and (ii) if it is preceded by a non-nasal Coda consonant. Otherwise, it is in rhythmical weak grade (i) if preceded by a long vowel or (ii) if it is in odd-numbered syllable. Otherwise, it is in syllabic weak grade in even-numbered closed syllables.

We created morpheme inventories by defining adjacency classes using the program *lexc* (Beesley-Karttunen 2003). The program *xfst* serves to describe a sequenced phonological rule-system by a set of context dependent re-write rules broadly used by generative phonologists. The program set composes the rules and the lexicon and the emerging full morpho-phonological description of the language is a two-level finite-state translating automaton, which can be used both for analysis and generation. Using the *xfst* formalism, we could create a full description of Nganasan. The calculus implemented by the program makes it possible to ignore irrelevant symbols (such as morpheme boundaries in the case of gradation) in the environment-description of a re-write rule; therefore environments encompassing non-adjacent morphemes can also be listed. As the program automatically eliminates intermediate levels of representation created by individual rules, generation and analysis can be performed efficiently.

4. Corpus and Other Tools for Testing of the Nganasan Description

We have elaborated the fairy tale collection of Labanauskas (2001). It consists of 58 texts and more than 17000 running words. Our corpus contains other texts collected by the members of our research group as well: they consist of 4400 words altogether. Unfortunately, almost all of the elements of the collection use inconsistent encoding system, thus their normalization was one of the first tasks. Then we made frequency statistics using the corpus. Word form statistics served as input of the morphological analyzer showing various parses (in the box below the entry in question) according to the recent status of our description of Nganasan morphology[2] (Figure 1).

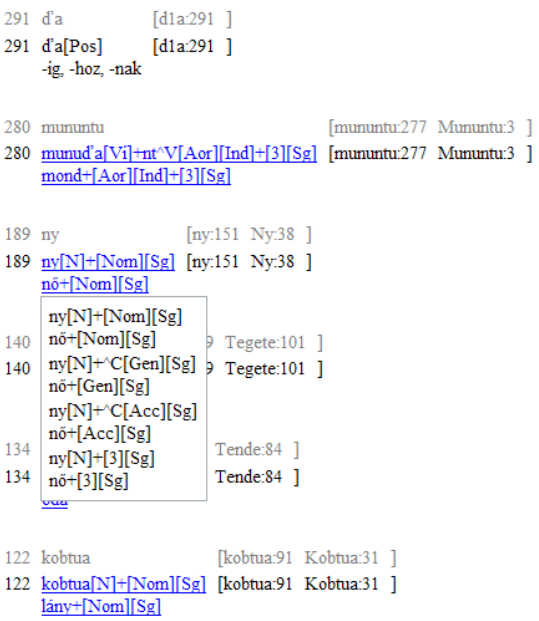

Figure 1: Output of the Nganasan morphological analyzer

The present version of the analyzer leaves 3.7% of the words of the above mentioned corpus unanalyzed. In the case of another 7.9% of the words, analysis is only successful with a version of the analyzer in which some phonological constraints are relaxed.

Although morphological analyzers can be used to rapidly analyze huge amounts of text, they cannot be used alone to create morpho-syntactically annotated corpora, because there is always a great degree of morphological ambiguity in the texts. In addition, corpora always contain a number of out-of-vocabulary word forms that the morphological analyzer is not able to recognize. Usually, some kind of morphological guessing may be used to solve this latter problem, but that usually leads to a disambiguation problem again: that of the possible guessed analyses. The morphological annotation needs to be disambiguated. Although there are standard (statistical) techniques of automatic disambiguated morpho-syntactic (part-of-speech) tagging, these tagging tools must always be trained on manually disambiguated texts. And in fact for the automatic tagging to be of an acceptable accuracy, a much larger amount of manually tagged training data is needed

[1] If there is a nasal in the previous syllable, weakening of the nasal+obstruent cluster optionally blocked (and it surface in strong grade).

[2] In the present version glosses are in Hungarian only.

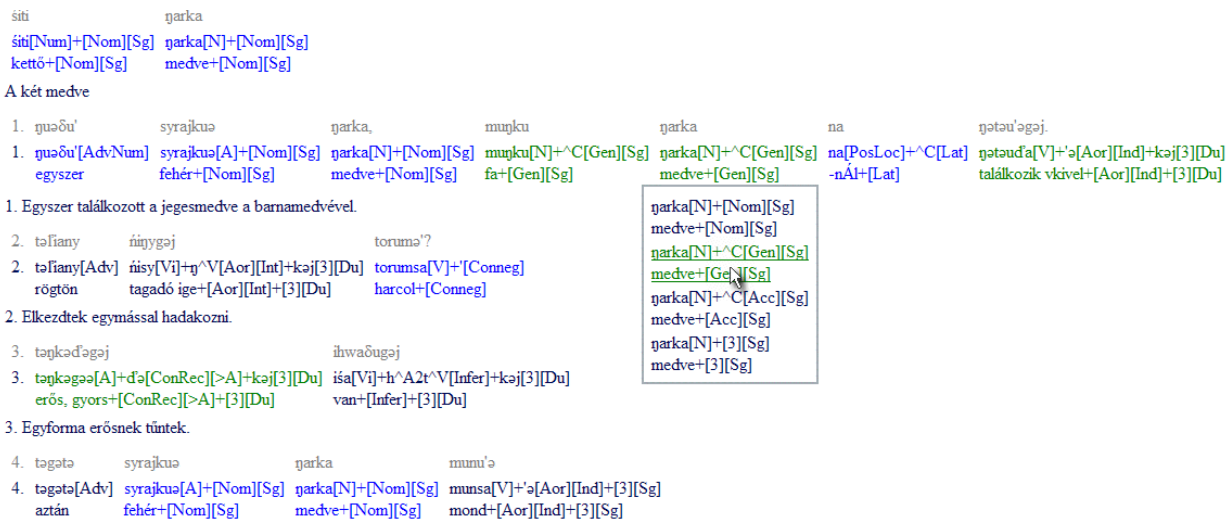

Figure 2: Screen based output of the morphological analysis of Nganasan texts

than available (and even then there will be tagging errors). Another problem with standard part-of-speech taggers is that they do not identify the lemma of words (only the part of speech tag), which is only half of the annotation that we would like to have. Moreover, the word form and the part

of speech tag do not always identify the lemma unambiguously, because the paradigms of different lemmas quite often partially overlap at the same paradigm slots. In those cases the lemma cannot be identified fully automatically from the part of speech tagged text. Thus manual disambiguation is inevitable (for at least a subset of the corpus). So a tool is needed that makes the manual disambiguation task as efficient as possible.

We have created a tool that can be used for the morpho-syntactic annotation and manual disambiguation of corpora. In order to make the use of this tool efficient, we implemented it as a web application so that it can be concurrently used by linguists/native speakers remotely. It can of course also be installed on and used locally form a local web server.

After tokenizing and morphologically analyzing the text uploaded to the web server, the tool presents individual sentences to the user along with their context clearly indicating ambiguous and unanalyzed words, with the possibility of manually adding analyses of unknown words, removing bogus nonsense analyses (regular expressions can be used to override whole classes of unwanted analyses). The program uses statistical methods to initially rank analyses so that the automatically top ranked analyses of ambiguous words rarely need to be manually overridden. The program learns from the decisions of the user. Initial ranking of the analysis candidates can be based on the output of a tagger, the accuracy of which can be incrementally enhanced by adding more and more texts to its training set. In addition to annotating words with their lemmas and morpho-syntactic tags, the tool can be configured to add glosses in various languages. When, after making the needed adjustments, the top ranked analysis and glossing candidates are all deemed correct, the user can accept the sentence as correctly analyzed. Manually overridden ranking is always recorded as such. For each disambiguated sentence, the user id of the annotator is logged. Manual correction of typos in the original text is also possible. The user can also mark sentences as problematic. If an update of the database of the morphological analyzer is needed, the corpus can be reanalyzed using the recompiled analyzer without the already disambiguated and accepted sentences being affected.

A morphological analyzer is not enough for checking the adequacy of the inflectional paradigms. Namely, one cannot detect that a possible alternative form of a certain word form is missing from the word's sample paradigm with the help of analyzer only. A morphological generator is also a very useful tool to track down problems when word forms in the corpus remained unanalyzed. With the help of the generator we could create the form that was adequate according to the grammar. In many cases this strategy led us to the source of the problem in our system.

5. Features of the Web-based System

The Nganasan analyzer and generator have been combined with a web page, and it runs on the web server of MorphoLogic: http://www.morphologic.hu/urali/index.php?lang=english.

As the title of the page suggests, in the future we plan to add further tools for other Uralic languages (Prószéky–Novák 2005).

We have developed an ergonomic way to show the potential analyses of ambiguous words. The parses of the word forms of the input sequence are interlinear. It means that a single analysis is shown on the screen for each input word, but further segmentations can be seen in a pop-up window if the mouse cursor is over the word form in question. The user can chose any of the offered analyses to

replace the original output on the screen (Figure 2). The word form generator is also given in the form of a web service (Figure 3).

We have also developed a soft keyboard the help of which authentic Nganasan texts can be inputted without installing new drivers to the system (Figure 4).

Figure 3: Web-based Nganasan word form generator

Figure 4: Soft keyboard for inputting Nganasan

6. Conclusion

We have developed a computational toolset – morphological analyzer, morphological generator, dictionaries and test corpus – for Nganasan. The language has not yet been morphologically described as thoroughly as with the help of these tools: details of the description which often remain vague in non-computational grammars unavoidably had to be made explicit in the computationally implemented grammar. Many gaps, uncertainties and inconsistencies were detected and in many cases we could correct our grammars and dictionaries. With the help of a corpus we built, the adequacy of the implemented description was very thoroughly tested. It is very important to note here that many questions which remained open should induce further field research. The tools we developed can be used to annotate corpora to facilitate research on other aspects of Nganasan.

7. References

Beesley, K.R., Karttunen, L. (2003) *Finite State Morphology.* CSLI Publications. Stanford: Stanford University.

Kost'erkina, N.T.. Momd'e, A.Č., Ždanova , T. Yu. (2001) *Slovar' nganasansko–russkij i russko-nganasanskij.* Sankt-Pet'erburg : Prosvesčen'ije.

Labanauskas, K.I. (2001). *Nganasanskaya folklornaya khrestomatiya (Нганасанская фольклорная хрестоматия).* Dud'inka: Таймырский окружный центр народного творчества..

Novák, A. (2008) Language Resources for Uralic Minority Languages. *Proceedings of the SALTMIL Workshop at LREC-2008: Collaboration: Interoperability between People in the Creation of Language Resources for Less-resourced Languages*. Marrakech: ACL. pp.27–32.

Prószéky, G., Novák, A. (2005). Computational Morphologies for Small Uralic Languages. In: Arppe Antti et al. (eds.) *Inquiries into Word, Constraints and Contexts (Festschrift in the Honour of Kimmo Koskenniemi on his 60th Birthday).* Stanford: CSLI Publications, Stanford University, pp. 150-157.

Wagner-Nagy, B. (ed.) (2002). *Chrestomathia Nganasanica.* SUA Supplementum 10. Szeged–Budapest: SZTE Finnugor Tanszék – MTA Nyelvtudományi Intézet

Developing a Large-Scale Lexicon for a Less-Resourced Language: General Methodology and Preliminary Experiments on Sorani Kurdish

Géraldine Walther[1] **& Benoît Sagot**[2]

1. LLF, Université Paris 7, 30 rue du Château des Rentiers, 75013 Paris, France
2. Alpage, INRIA Paris–Rocquencourt & Université Paris 7, Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France
geraldine.walther@linguist.jussieu.fr, benoit.sagot@inria.fr

Abstract

In this paper, we describe a general methodology for developing a large-scale lexicon for a less-resourced language, i.e., a language for which raw internet-based corpora and general-purpose grammars are virtually the only existing resources. We apply this methodology to the development of a morphological lexicon for Sorani Kurdish, an Iranian language mostly spoken in northern Iraq and north-western Iran. Although preliminary, our results demonstrate the relevance of this methodology.

1. Introduction

Building large scale language resources for languages where there are only few linguistic resources and even less, if any, NLP resources available constitutes a challenge for NLP resource development. In this work, we aim at building a methodology which will allow us to develop new language resources for less-resourced languages from scratch. We will especially concentrate on the development of lexical resources, for the benefit they offer as such and as a starting point in the development other NLP resources and tools.

We first describe our methodology for building new language resources for resource-scarce languages (Section 3.). It uses solely raw on-line corpora and a few (basic) linguistic sources, such as simple reference grammars. In section 4., we illustrate this methodology with the description of SoraLex, a new, if preliminary, morphological lexicon for Sorani Kurdish destined to be enriched and completed with further syntactic information.[1]

2. Related work

In the past years, a large variety of approaches have been described aiming at developing morphological linguistic resources, in particular for less-resources languages. All of them try to benefit as much as possible from the limited amount of available data and information. Some approaches do not even rely on any prior linguistic knowledge, and fall in the paradigm of the unsupervised learning of a language's morphology (Goldsmith, 2001; Baroni, 2003; Creutz and Lagus, 2005). In such approaches, a raw corpus (usually in the form of a list of words) serves as the only input of the system, which automatically produces either a segmentation for each word into its morphemes, or even a full set of inflectional paradigms, associated with a set of lemmas (Snover and Brent, 2001). These techniques are useful for various purposes, including providing linguistic insights which are independent from the grammatical tradition of the considered language, if any. However, given the complexity and richness of morphological studies accessible for a very large range of languages, we agree with Forsberg et al. (2006) that it is time- and precision-wise counter-productive to try and automatically reproduce all this complexity instead of formalising morphological analyses available through linguistic literature.

In that regard, our approach is closer to most large-scale morphological resource development efforts (Ide and Véronis, 1994; Zanchetta and Baroni, 2005; Sagot, 2010), that also rely on explicit or implicit formalised morphological descriptions embedded in or compiled into part-of-speech (POS) taggers, lemmatisers and/or morphological analysers. However, we do not want to mandatorily rely on a lemmatiser or even on a POS tagger, as we aim at dealing with languages for which such tools do not yet exist. In further stages of the lexicon development, it shall of course become possible to POS-annotate a corpus of increasing size and therefore to train a POS-tagger, that shall give us access to acquisition techniques such as described by Molinero et al. (2009). However, we first need techniques that are able to automatically extract new lexical entries (i.e., lemmas and their associated inflection class), starting from a raw corpus and a formalised morphological description.

Several algorithms have been designed to extract new lemmas from such a limited amount of information. They have been applied to several languages such as Russian (Oliver et al., 2003), French verbs (Clément et al., 2004), German nouns (Perera and Witte, 2005), Slovak (Sagot, 2005), French verbs, nouns and adjectives (Forsberg et al., 2006) and Polish (Sagot, 2007). These techniques differ from one another in various aspects, such as the soundness of the underlying probabilistic model and/or heuristics, the richness of the manually described linguistic clues that are exploited (constraints on possible stems for each inflectional class, derivation patterns…), the use of Google for checking the "existence" of a form, or the possibility to benefit from (probabilised since uncertain) part-of-speech information when it becomes available.

In this work, we try to combine some of these ideas and techniques so as to define an efficient methodology for

[1] As we shall explain below, we call a *morphological lexicon* a set of entries of the form *(lemma,inflection class)* and the associated formalised description of the inflection classes. This allows for building, e.g., inflection and lemmatisation tools and a full-form lexicon (see below).

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 45–52

the development of a morphological lexicon for resource-scarce languages, and apply it to Sorani Kurdish.

3. A methodology for developing basic language resources from scratch

3.1. Constructing the morphological architecture

The most basic and yet most needed step in our language resource development is the construction of a morphological lexicon. A morphological lexicon associates a lemma and a morphosyntactic tag with each known wordform (form, in short).[2] However, building a morphological lexicon of a given language cannot be efficiently done without sufficient insight into this language's morphology. One needs to have at least access to a basic set of lexical entries and their morphosyntactic features in order to define the lemmas and the possible morphosyntactic tags of a given form. Our methodology therefore requests a preliminary study of the language's morphological specificities. These can however be extracted quite easily from simple linguistic descriptions of the language.

A summary linguistic study of the language's morphology should allow for the definition of a list of parts-of-speech together with their inflectional properties. From there, the linguistic descriptive features can be converted into an NLP tool-accessible language.

We chose to use the *Alexina* framework (Sagot, 2010) as a baseline for our lexical resource development. One asset of this framework lies in covering both the morphological and the syntactic level (e.g., valency) of a given lexicon — which shall be useful in further stages of the lexical resource development. *Alexina* offers an opportunity for representing lexical information in a complete, efficient and readable way (Sagot, 2005; Sagot, 2007; Sagot, 2010). Moreover it is compatible with the LMF standard[3] (Francopoulo et al., 2006).[4]

The *Alexina* model is based on a representation that separates the description of a lexicon from its use:

- The intensional lexicon factorises the lexical information by associating each lemma with a morphological class (previously defined in a formalised morphological description) and deep syntactic information; it is used for lexical resource development;
- The extensional lexicon, which is generated automatically by *compiling* the intensional lexicon, associates each inflected form with a detailed structure that represents all its morphological and syntactic information; it is directly used by NLP tools such as parsers.

Within this model, the necessary tasks for developing an intended new resource therefore consist in elaborating a formalised description of the targeted language's morphology, converting this description into the *Alexina* morphological language (Sagot, 2007) and finding possible lexical entries that can be associated with the inflection tables defined within the chosen *Alexina* model.

In the *Alexina* formalism, inflection is modelled as the affixation of a prefix and a suffix around a stem, while *sandhi* phenomena may occur at morpheme boundaries, sometimes conditioned by stem properties.[5] The formalism, which shares some widespread ideas with the DATR formalism (Evans and Gazdar, 1990) , relies on the following scheme:

- The core of a morphological description is a set of inflection classes which can (partly or completely) inherit from one another,
- Each inflection class defines a set of forms, each one of them being defined by a morphological tag and by a prefix and a suffix that, together with the stem, constitute the morpheme-like sequence *prefix_stem_suffix*;
- *Sandhi* phenomena allow to link the surface form to the underlying *prefix_stem* and *stem_suffix* sequences by applying regular transformations;
- Forms can be controlled by tests over the stem (e.g., a given rule can apply only if a given regular expression matches the stem and/or if another one does not match the stem);
- Forms can be controlled by "variants" of the inflection classes (e.g., forms can be selected by one or more flags which complement the name of the class).

Tables 2 and 1 in Section 4.3. illustrate this model by showing respectively a few *sandhi* rules and an excerpt of a verbal inflection class.

Translating a morphological description into the *Alexina* morphological language requires making choices about what will have to be treated as a dependent affix (prefixed or suffixed to the to-be-determined stems), an independent though typographically joined form or a typographically autonomous form. For that reason, the first descriptive task of the resource development consists in identifying the different affixes that can combine with possible stems. These identified affixes are used for constructing the inflectional tables associated with each of the previously defined inflectable parts-of-speech.

The second task consists in *somehow* gathering possible lexical entries for each part-of-speech (see Section 3.2.).

[2] Of course, a same form may receive more than one *(lemma,morphosyntactic tag)* pair.

[3] The Lexical Markup Framework ISO/TC37 standard.

[4] A fair number of lexical resources are already being developed within the *Alexina* framework, such as the Le*fff* for French (Sagot, 2010), the Le*ffe* for Spanish and other resources for Galician, Polish, Slovak, Persian and English. This fact should ensure the workability of new *Alexina* lexicons. It may also pave the way for future cross-language NLP applications.

[5] A *sandhi* — the term comes from traditional Sanskrit grammars — is a transformation of a given phonological/typographic sequence due to its encountering another specific sequence. The term *sandhi* is however nowadays used mainly although not always in order to refer to transformations occurring at morpheme boundaries. For example, in French, when the suffix *-ons* (1st person plural) is juxtaposed to the stem *mang-* (*to eat*), a *sandhi* phenomenon occurs that causes the insertion of a *e*, thus producing the form *mangeons* (*(we) eat*).

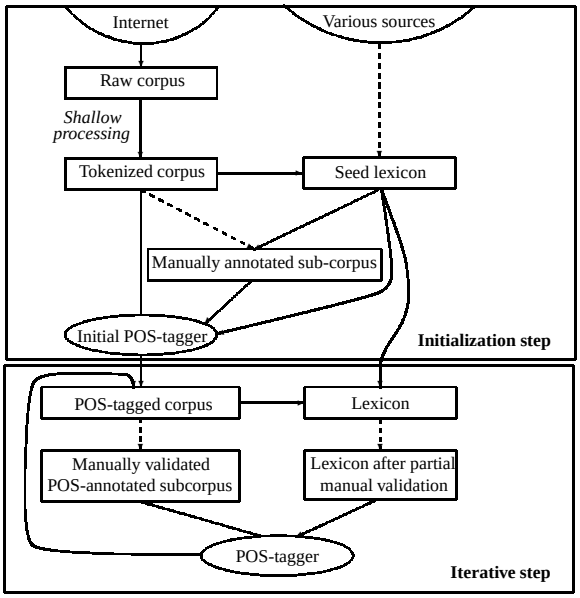

Figure 1: Overview of our lexical resource development methodology (dashed lines denote manual or semi-automatic steps, plain lines automatic steps)

3.2. Initialisation step: building a seed lexicon

Since an *Alexina* morphological lexicon consists of both a morphological description of a language and a set of lexical entries sorted according to their parts-of-speech, the next step in the development of the intended new lexical resource consists in finding possible candidates for the different word classes. This corresponds to the "initialisation step" in Figure 1.

To do so, one can either manually list a certain amount of lemmas associated with their inflectional class — that is, if there exists a resource listing such candidates — or, if such resources fail to be available, use the previously elaborated morphological description to infer the possible entries from obvious stems having combined with the identified affixes. All that is needed with this second method is a relatively large raw corpus. First we have to tokenise the raw corpus in order to extract a list of possible combinations of stems and affixes.[6] The types of possible combinations offer relatively accurate evidence for the classification of the inflectable words. This however only works with languages that display sufficiently rich inflectional classes and with those words which in fact combine with established affixes. For other cases, manual listing seems unavoidable.

After listing the lexical entries, we should be in possession of a small seed lexicon which will constitute the baseline for the development of all further large-scale resources.

3.3. Iterative step: enriching the lexicon and developing further resources

The "iterative step" of our methodology consists in using the newly built morphological seed lexicon to create other NLP tools which conversely allow to further develop the lexicon. The lexicon and these tools therefore benefit from each others' improvement.

Together with a limited-size manually POS-annotated corpus,[7] the information within the seed lexicon allows for building a specific lexicon-aware POS-tagger such as MElt (Denis and Sagot, 2009) for the newly-to-be-endued language. Once trained, MElt will be able to generate POS-tagged corpora for the targeted language, hence paving the way for the automatic extraction of candidate lexical entries thanks to simple techniques such as those described by Molinero et al. (2009). Of course, the newly suggested entries will require some (partial) manual validation, yet, since validating lexical entries is much less time-consuming than validating tagged corpora (Denis and Sagot, 2009) or, even more so, manually annotating raw corpora, this method does undoubtedly provide a much faster means for developing large-scale lexical resources from scratch.

A further expansion of the obtained resource would be the addition of the syntactic level of the lexicon, for which *Alexina* is already fully equipped. This step will require some more specific linguistic analysis and formalisation of the language's syntactic features, yet the necessary study of those features will conversely benefit from the existence of new POS-tagged corpora. Once the syntactic module of the new lexicon completed, it will also become possible to develop other NLP tools, such as parsers, for this language.

In brief, using the newly trained MElt-based POS-tagger will rapidly provide us with vast POS-tagged and morphologically annotated corpora, which will help improving the morphosyntactic lexicon, the underlying linguistic descriptions, and all other derived NLP tools. Thus, at any stage of our resource development, the interplay of the different modules enables an automatic iterative enrichment of each one of them.

4. A real case-study: SoraLex

We tested the above described methodology by building SoraLex, a morphological lexicon for Sorani Kurdish. For now, SoraLex only takes into account the morphological level of the intended lexicon, but it is destined to later be completed for syntactic information as well.

Sorani Kurdish is a resource-scarce language for which the only NLP resource available on the Internet seems to be raw text; as opposed to numerous other languages, there appear to be no usable on-line NLP tools accessible. We were therefore forced to build the whole development of our resource solely on some raw on-line corpora and the few existing linguistic descriptions of the language.[8] Since the first known description of the Kurdish language by Maurizio Garzoni (1787), only few other grammars have been published, none of them adopting a contemporarily formalised approach. For our description, we have been relying mainly on the descriptions of (McCarus, 1958; MacKenzie, 1961; Blau, 2000; Thackston, 2006).

Using solely those sources, we were yet able to build a preliminary version of the SoraLex morphological lexicon.

[6] We shall see in section 4. that the definition of the tokens sometimes requires a set of preliminary word vs. affix definitions.

[7] Say, a few hundreds of sentences.

[8] Although most of these descriptions are available on-line in the form of PDF documents, they can obviously not be considered as NLP resources and be used as such.

4.1. Sorani Kurdish language in brief

The Kurdish languages belong to the western branch of the Indo-Iranian languages. Kurdish speakers are mostly to be found in central and western Turkey, southern Iraq and Syria and western Iran. Yet a great number of Kurdish speakers also dwell in the neighbouring territories as well as spread all over the globe wherever the Kurdish diaspora has fancied to scatter them.[9]

Kurdish is composed of several dialects, of which the two major groups are the northern *Kurmanji*[10] written with extended Latin characters and the central/southern *Sorani*[11] written within a modified version of the Arabic script. Kurmanji and Sorani both possess a standardised form,[12] which, in the case of Sorani, has been largely shaped through the influence of the Kurdish Academy in Baghdad in the 1970s.

Standard Sorani Kurdish tends towards the Sulaymaniyah dialect of the north-eastern Iraqi province of As-Sulaymaniyah counting about one million Kurdish speakers.

Both in the remainder of this paper and in the SoraLex lexicon, we use an extension of the bijective transliteration employed for developing the PerLex lexicon for the Persian language (Sagot and Walther, 2010).[13]

4.2. The major morphological features

Sorani grammars (like those of the employed reference grammars) generally list the following parts-of-speech: nouns, verbs, pronouns and several *particles*.[14]

In our morphological description, we distinguish proper nouns, determiners, conjunctions, complementisers and prepositions in addition to the above mentioned classes. Though not yet explicitely linguistically motivated, our choices are preliminarily derived from usual classes within typological approaches. While most of these parts-of-speech correspond to their usual definition, the particle-class requires a closer look. Among the particles, we have counted the several pre- and postverbs (MacKenzie, 1961), adverbial suffixes and the second elements of Sorani circumpositions (*-ewe* and *-da*) (Thackston, 2006).

Concerning inflectional morphology, Sorani Kurdish, like most Indo-European languages, displays two major inflectional classes, the nominal class (including nouns, proper nouns, pronouns and adjectives) and the verbal class. In our approach towards the construction of a new lexical resource for Sorani Kurdish, those two classes have been endowed with a complete morphological description which has afterwards been adapted to the *Alexina* morphological language.

Concerning nominal inflection, the following elements have been included as affixal elements: the indefinite marker *-ěk*, the singular and plural definite marker *-eke*, and *-ekan*, the enclitic particle *-y* for marking modified nouns, called Ezafe, the enclitic pronominal person markers *-m*, *-t*, *-y*, *-man*, *-tan* and *-yan*, the demonstrative circumfixal demonstratives (Thackston, 2006) composed of the close *em-* and distant *ew-* respectively combined with the suffix *-e* and the focus particle *-yš*. As opposed to a certain amount of other Kurdish languages, Sorani Kurdish has lost any kind of case opposition between direct and oblique nominal forms, nor does it display any inflection for gender.

Other affixal elements linked to the nominal part of the Sorani Kurdish inflectional system are the comparative *-tar* and superlative *-taryn* attaching to adjectives only.

These different affixal markers can combine with each other, thereby creating rather complex morphological inflection pattern.

Still, future work shall aim at further defining the status of the Ezafe, the indefinite and definite markers and the enclitic personal suffixes, since their morphosyntactic properties clearly indicate a rather ambiguous status of these elements.

Concerning the verbal class, Sorani Kurdish resembles most Iranian languages in the fact that it possesses only a very limited amount of verbal lexemes (around 300). Most verbal meanings known from the more extensively described Indo-European languages are expressed through complex verbal predicates build from a light verbal head and a predicative element which can be either a noun or an adjective, or even an adposition or a pre- or postverb (MacKenzie, 1961; Blau, 2000).

The construction of Sorani verb forms obeys the following rules. Most descriptions concur in stating the existence of two distinct verbal stems, one (SI) for the present tense forms, one (SII) for the past tense forms.[15] For now, we also adopt this approach to Sorani verb morphology.

Sorani verb forms consist of the combination of a given stem with a set of pre- and suffixes, such as in the following representation:

Modal/Temporal Prefix(es) - Stem - Personal Suffix(es).

However, number of other elements may be inserted between the affixes and the stem. Enclitic pronominal person markers, for example, often appear between the modal prefix and the stem. Those specific difficulties yet have to be taken into account for SoraLex.

Sorani Kurdish also displays three sets of personal suffixes, the first being used with present verb forms derived from SI, the second with most past tense verb forms from SII

[9] In Europe, for example, a significant part of the important Turkish community is in fact of Kurdish origin.

[10] About 50% of the Kurdish speakers.

[11] About 25% of the Kurdish speakers.

[12] Established orthographic rules, standard uses, available normalised on-line corpora like newspapers and other websites.

[13] The use of a transliteration has at least two motivations. First, it allows for an easier development (e.g., text editors are not always very left-to-right-script-friendly, and lexicographers are not always familiar with the arabic script). Second, we use a latin-2 transliteration, which is compatible with NLP tools that require 8-bit encodings.

[14] Yet those lists appear to be incomplete and do not make the linguistic choices underlying the classification explicit. This part of Sorani linguistic description yet needs to be done.

[15] However this statement is not followed by (Bonami and Samvelian, 2008) who suggest the existence of three distinct stems, one for the present tenses, one for the past tenses and one for the passive forms. Our reading of the data contained within the reference grammars also gave us the impression that the question of the number of stems still needs to be solved. Depending on the question's outcome, the here presented morphological lexicon might yet expect some substantial changes.

and the third, being identical to the enclitic present forms of the verb *bwwn* 'to be', with the remaining verb forms.
Yet the above-mentioned enclitic pronominal person markers may also function as agent-verb agreement markers for transitive verbs in the past tense. For those verbs, the normal personal suffixes function as patient markers. The interplay between the normal personal suffixes and the patient markers being a particularly complex phenomenon in Sorani Kurdish, we decided not to take into account the role of the pronominal person markers within the inflectional verb paradigm at this stage of our resource development and to wait for further linguistic insight. Concerning the pronominal person markers, linguistic motivation for treating them either within the morphological or the syntactic level of our *Alexina* lexicon might result in a substantial modifications in our morphological formalisation.
Having in mind the above sketched linguistically motivated morphological description, we have built a preliminary version of the morphological module of SoraLex.

4.3. An *Alexina* morphological description of Sorani Kurdish

As explained above, the first step of the development of the morphological module of SoraLex consisted in converting our morphological description gathered within the reference grammars of Sorani Kurdish so as to make them usable within the *Alexina* framework. Examples thereof are illustrated by the intransitive verb inflection class shown in Table 1 and in the noun inflection class shown in Table 5 together with a few sandhi rules in Table 2.[16]

```
<table name="v1intrans" canonical_tag="Inf"
                stems="..*[aywdt]">
    <form suffix="n" tag="Inf"/>
    <alt>
        <form suffix="ww" tag="PastPart" var="c"/>
        <form suffix="w" tag="PastPart" var="v"/>
    </alt>
...
    <form prefix="de" suffix="①m" tag="1sgPreInd"/>
    <form prefix="de" suffix="①y" tag="2sgPreInd"/>
    <form prefix="de" suffix="①ě" tag="3sgPreInd"/>
    <form prefix="de" suffix="①yn" tag="1plPreInd"/>
    <form prefix="de" suffix="①n" tag="2plPreInd"/>
    <form prefix="de" suffix="①n" tag="3plPreInd"/>
...
```

Table 1: Excerpts of the inflection class for Sorani Kurdish regular intransitive verbs in our *Alexina* morphological description.

Let us take the examples of the verbs *čwwn* 'to go' and *parastn* 'to protect'. *Čwwn* belongs to the so called regular intransitive verbs shown in Table 1 which form their present stems by simply dropping their final vowel, whereas *parastn* counts as an irregular (transitive) verb, showing notably a case of vowel alternation between SI and

[16]These tables are of course only excerpts of the full inflection tables contained within our *Alexina* description.

```
<sandhi source="ww_①" target="_"/>
<sandhi source="parast_①" target="parěz_"/>
```

Table 2: A few *sandhi* rules from our *Alexina* description of Sorani Kurdish morphology, used to model the alternations between stems (the "_" models a morpheme boundary)

Canonical form	Inflection class	SI	SII
čwwn	**v1intrans**	*č-*	*čww-*
parastn	**v2trans**	*parěz-*	*parast-*

Table 3: Two verbal entries with their corresponding stems

SII. Their respective present and past stems are shown in Table 3.
Table 1 shows how the canonical form for intransitive verbs, the infinitive, is formed by adding the suffix *-n* (suffix="**n**") to the default stem SII. In fact, this also applies to transitive verbs. The past participle forms are similarly formed by adding either *-ww* or *-w*, depending on whether the stem ends respectively in a consonant (var="**c**") or a vowel (var="**v**"), which is specified as a variant of the inflection class ("**v1intrans:v**" in the case of *čww-*, "**v2trans:c**" in the case of *parast-*). The present indicative forms make use of the sandhi phenomena shown in Table 2. Whenever the default stem (i.e., SII) encounters the symbol ① in an inflection table, the appropriate sandhi is triggered and the corresponding SI is generated. This results in the inflected forms shown in Table 4.
The case of nouns, illustrated in Table 6, is simpler, since nouns do not show stem alternations. Depending on the ending of their stems, they may only take certain forms of the following suffixes. As above, this constraint is modelled as inflection class variants (var="**c**" for stems ending in consonants and var="**v**" for stems ending in vowels).
Moreover, the "rads" and "except" constraints allow for further constraining the possible stems on which a suffix may attach: rads="**.*[eěao]**" allows for the suffix to attach on any stem ending in *e*, *ě*, *a* or *o*, while except="**.*[eěao]**" allows for the attaching of a given suffix to any stem except those ending in *e*, *ě*, *a* or *o*.
Table 6 shows an excerpt of the inflected forms for the nouns *dost* 'friend' and *dě* 'village', ending respectively in a consonant and in a vowel, as generated by the inflection class shown in Table 5.

4.4. Creation of a raw corpus and a seed lexicon

As mentioned above, the only other source of information we exploited is a raw corpus of Sorani Kurdish. We extracted such a corpus from the blog[17] of the programme *Ruwange* broadcasted by the Belgium-based Kurdish channel *Roj TV*. This blog allows for the automatic recursive retrieval of its pages, which we performed with the standard tool `wget`. We extracted all textual sections from the HTML files, removed all markup, filtered out lines that did not have the appearance of valid Sorani text (character set, spacing characteristics...) and segmented

[17]`http://ruwange.blogspot.com/`

Inflected form	***čwwn***	***parastn***
Inf	*čww_n*	*parast_n*
PastPart	*čww_w*	*parast_ww*
1sgPreInd	*č_m*	*parěz_m*
2sgPreInd	*č_y*	*parěz_y*
3sgPreInd	*č_ě*	*parěz_ě*
1plPreInd	*č_yn*	*parěz_yn*
2plPreInd	*č_n*	*parěz_n*
3plPreInd	*č_n*	*parěz_n*

Table 4: Several inflected forms for the verbal entries in Table 3

it automatically into sentences based on final punctuation marks. Then we normalised[18] and transliterated all characters. We tokenised the corpus,[19] resulting in 590,568 token occurrences and 62,993 unique tokens. The most frequent tokens are the preposition *le*, the conjunction *w* and the preposition *be*.

With the help of this frequency list and the grammars listed above, we manually created a set of closed-class entries (29 conjunctions and complementisers, 22 punctuation marks, 10 determiners, 49 prepositions, 26 pronouns, 38 numerals, 10 particles). We also built a lexicon of 68 verb lemmas, which already covers almost 25% of the full set of Sorani Kurdish verbs.

In order to extract nouns, adjectives and adverbs from our corpus in a more systematic way, we decided to start with a simple technique, based on our knowledge of Sorani Kurdish morphology. We designed a regular expression[20] covering a large range of possible nominal and adjectival suffixes, such that the removal of these suffixes provides a nominal or adjectival candidate stem, i.e., in Sorani Kurdish, a lemma. In order to rank the obtained lemmas, we take advantage of the following hypotheses. First, the longer a suffix, the more probable it is correctly identified, and therefore its removal provides a valid nominal or adjectival lemma. Second, the more different suffixes have been identified on a given stem/lemma, the more confident we are in its correctness. Therefore, we assigned to each suffix a weight equal to its length, and weighted each candidate lemma by the sum of the weights of all (unique) suffixes it has been encountered with. This resulted in a list of 1,009 candidate lemmas with a weight of 10 or more, for

[18]Sorani Kurdish, as Urdu, has the following property. The isolated and final forms of the Arabic letter *hâ* constitute one letter (pronounced *e*), whereas the initial and medial forms of the same Arabic letter constitute *another* letter (pronounced *h*), for which a different Unicode encoding is available. In many electronic texts, such as the blog we used as a corpus, these letters are written using only the *hâ*, and differentiate both letters using the *zero-width non joiner* character that prevents a character from being joined to its follower. We had to normalise this in order to get two different Unicode encodings for these two different letters.

[19]For this task we used a simple tokeniser, that only recognises numbers, URLs, email addresses and a few other very surfacic phenomena. It then identifies all punctuation marks as individual tokens, as well as all remaining sequences of non-whitespace characters.

[20]`((y[eě]|ě)(k(an|e)?)?)?(y?š)?([mty](an)?)?y?$`

```
<table name="N1" rads="..*">
    <form suffix="" tag="Abs"/>
    <alt>
        <form suffix="ěk" tag="SingIndef"
              rads=".*" var="c"/>
        <form suffix="ě" tag="SingIndefFam"
              rads=".*" var="c"/>
        <form suffix="yěk" tag="SingIndef"
              rads=".*" var="v"/>
        <form suffix="yě" tag="SingIndefFam"
              rads=".*" var="v"/>
        <form suffix="yek" tag="SingIndefFam"
              rads=".*" var="v"/>
        <form suffix="ye" tag="SingIndefFam"
              rads=".*" var="v"/>
    </alt>
...
    <alt>
        <form suffix="an" tag="PlIndef"
              rads=".*" var="c"/>
        <form suffix="yan" tag="PlIndef"
              rads=".*" var="v"/>
    </alt>
...
    </alt>
        <form suffix="eke" tag="SingDef"
              except=".*[eěao]" var="c"/>
        <form suffix="eke" tag="SingDef"
              except=".*[eěao]" var="v"/>
        <form suffix="ke" tag="SingDef"
              rads=".*[eěao]" var="v"/>
    </alt>
...
    <alt>
        <form suffix="ekan" tag="PlDef"
              except=".*[eěao]" var="c"/>
        <form suffix="ekan" tag="PlDef"
              except=".*[eěao]" var="v"/>
        <form suffix="kan" tag="PlDef"
              rads=".*[eěao]" var="v"/>
    </alt>
...
```

Table 5: Excerpts of the inflection class for Sorani Kurdish nouns in our *Alexina* morphological description

Inflected form	***dost***	***dě***
SingIndef	*dost_ěk*	*dě_yěk*
SingIndefFam	*dost_ě*	*dě_yě* *dě_yek* *dě_ye*
PlIndef	*dost_an*	*dě_yan*
SingDef	*dost_ke*	*dě_ke*
PlDef	*dost_ekan*	*dě_kan*

Table 6: Several inflected forms for the nouns *dě* 'village' and *dost* 'friend'

which we performed a partial manual validation.

In order to build additional open-class candidates, we also applied our implementation of the algorithm described in (Sagot, 2005). This algorithm is based on the list of unknown and open-class tokens associated with their frequencies. On our corpus, and taking into account

the already existing entries, we obtained 4,104 candidate lemmas, ordered according to a weight that takes into account both the likelihood of each lemma as computed by the algorithm and the number of occurrences of its inflected forms. We manually validated a limited amount of these candidates. A web-based interface already developed and used for other lexical development projects shall allow for an efficient large-scale manual validation of these candidate entries, and therefore improve the coverage of SoraLex in the near future.

Finally, we used the Sorani Kurdish Wikipedia[21] for collecting proper nouns. Those were found through the titles of Wikipedia articles indicating either a city, a country or a person *category*. We collected and normalised the titles of these articles as well as those of all the articles redirecting towards them. We were thereby able to build a lexicon for proper nouns consisting in person, country and city names. These tasks resulted in a set of (only) 131 proper noun lemmas. Person names have been assigned the class of invariable lemmas, whereas countries and cities received an inflectional noun class that doesn't allow for the formation of plural forms.

Using these manual and semi-automatic techniques, we obtained a seed lexicon for Sorani Kurdish. This lexicon contains 17,600 extensional (form-level) entries corresponding to 13,315 different forms from 468 intensional (lemma-level) entries. This lexicon covers 48.4% of all token occurrences in our raw corpus.

5. Conclusion

In this paper, we introduced a three-step methodology for developing morphological lexicons for resource-scarce languages, i.e., languages for which raw corpora and linguistic studies are basically the only available sources of information. First, we argued for the relevance of a careful linguistic study allowing for the manual development of a formalised description of the language's morphology. In a second step, the initialisation step, we suggest employing both existing and novel techniques that use such a description for constructing semi-automatically a *seed* lexicon from a raw corpus of the language. Coupled with a (small) manually annotated corpus, this seed lexicon helps training a preliminary version of a lexicon-aware part-of-speech tagger such as MElt (Denis and Sagot, 2009), which enables to generate a large POS-tagged corpus. Such a corpus is in turn useful for efficiently improving the coverage of the lexicon (Molinero et al., 2009), and therefore the quality of the tagger, thus defining a virtuous iterative process.

We illustrate this methodology by reporting the first steps towards the development of a large-scale morphological lexicon for Sorani Kurdish within the *Alexina* framework. We are currently about to move from the initialisation step to the iterative step. Apart from following our methodology, we aim at exploring other complementary approaches. In particular, we plan to develop techniques for extracting relevant information from existing lexical resources available for closely related languages. Ongoing work in this direction has given satisfying results for the Galician language starting from resources for Spanish, and we intend to benefit from the ongoing initiative around the PerLex lexicon for Persian (Sagot and Walther, 2010) so as to try and gather complementary information.[22]

On the longer term, we intend to develop a first set of NLP tools for Sorani Kurdish based on SoraLex and existing technologies already adapted to Persian language based on PerLex. This includes, among others, advanced tokenisation and segmentation modules, named entity recognisers and spelling correctors.

SoraLex, as all *Alexina* lexicons, is available under a free software license (LGPL-LR) on the web-page of the *Alexina* project.[23]

6. References

Marco Baroni. 2003. Distribution-driven morpheme discovery: A computational/experimental study. In Geert Booij and Jaap van Marle, editors, *Yearbook of Morphology 2003*, pages 213–248. Dordrecht: Springer.

Joyce Blau. 2000. *Méthode de kurde sorani*. L'Harmattan, Paris, France.

Olivier Bonami and Pollet Samvelian. 2008. Sorani kurdish person markers and the typology of agreement. In *13th International Morphology Meeting*, Vienna, Austria.

Lionel Clément, Benoît Sagot, and Bernard Lang. 2004. Morphology based automatic acquisition of large-coverage lexica. In *Proceedings of the 4th Language Resources and Evaluation Conference (LREC'04)*, pages 1841–1844, Lisbon, Portugal.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05)*, pages 106–113, Espoo, Finland.

Pascal Denis and Benoît Sagot. 2009. Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art POS tagging with less human effort. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation (PACLIC 2009)*, Hong Kong.

Roger Evans and Gerald Gazdar. 1990. The DATR Papers: February 1990. Technical Report CSRP 139, University of Sussex, Brighton, UK.

Markus Forsberg, Harald Hammarström, and Aarne Ranta. 2006. Morphological lexicon extraction from raw text data. In *Proceedings of FinTAL 2006, LNAI 4139*, pages 488–499, Turku, Finland. Springer-Verlag.

Gil Francopoulo, Monte George, Nicoletta Calzolari, Monica Monachini, Nuria Bel, Mandy Pet, and Claudia Soria. 2006. Lexical Markup Framework (LMF).

[21] Available at the following address: `http://ckb.wikipedia.org`. We used the dump of March 26th, 2010.

[22] This idea could also be used in order to develop a lexicon for Kurmanji Kurdish from SoraLex, and/or to benefit from existing limited-size lexical resources for this language.

[23] `http://alexina.gforge.inria.fr/`

In *Proceedings of the 5th Language Resources and Evaluation Conference (LREC'06)*, Genoa, Italy.

Maurizio Garzoni. 1787. *Grammatica e Vocabulario della Lingua Kurda*. Sacra Congregazione di Propaganda Fide, Rome, Italy.

John A. Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198.

Nancy Ide and Jean Véronis. 1994. MULTEXT: Multilingual text tools and corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING'94)*, Kyoto, Japan.

David N. MacKenzie. 1961. *Kurdish dialect studies*, volume 1 of *London Oriental Series*. Oxford University Press, London, UK.

Ernest N. McCarus. 1958. *A Kurdish Grammar: descriptive analysis of the Kurdish of Sulaimaniya, Iraq*. Ph.D. thesis, American Council of Learned Societies, New-York, USA.

Miguel Ángel Molinero, Benoît Sagot, and Lionel Nicolas. 2009. A morphological and syntactic wide-coverage lexicon for Spanish: The Le*ff*e. In *Proceedings of the 7th conference on Recent Advances in Natural Language Processing (RANLP 2009)*, Borovets, Bulgaria.

Antoni Oliver, Irene Castellón, and Lluís Màrquez. 2003. Use of Internet for augmenting coverage in a lexical acquisition system from raw corpora: application to Russian. In *Proceedings of the RANLP'03 International Workshop on Information Extraction for Slavonic and Other Central and Eastern European Languages (IESL'03)*, Borovets, Bulgaria.

Praharshana Perera and René Witte. 2005. A self-learning context-aware lemmatizer for German. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, pages 636–643, Vancouver, Canada.

Benoît Sagot and Géraldine Walther. 2010. A morphological lexicon for the Persian language. In *Proceedings of the 7th Language Resources and Evaluation Conference (LREC'10)*, Valetta, Malta. To appear.

Benoît Sagot. 2005. Automatic acquisition of a Slovak lexicon from a raw corpus. In *Lecture Notes in Artificial Intelligence 3658 ((c) Springer-Verlag), Proceedings of TSD'05*, pages 156–163, Karlovy Vary, Czech Republic.

Benoît Sagot. 2007. Building a morphosyntactic lexicon and a pre-syntactic processing chain for Polish. In *Proceedings of the 3rd Language & Technology Conference (LTC'05)*, pages 423–427, Poznań, Poland.

Benoît Sagot. 2010. The Le*fff*, a freely available, accurate and large-coverage lexicon for French. In *Proceedings of the 7th Language Resources and Evaluation Conference (LREC'10)*, Valetta, Malta. To appear.

Matthew G. Snover and Michael R. Brent. 2001. A bayesian model for morpheme and paradigm identification. In *Proceedings of the 39th annual meeting of the ACL*, pages 490–498, Toulouse, France.

Wheeler M. Thackston. 2006. Sorani kurdish: A reference grammar with selected readings. www.fas.harvard.edu/~iranian/Sorani/sorani_1_grammar.pdf.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! a free corpus-based morphological resource for the Italian language. In *Proceedings of Corpus Linguistics 2005*, Birmingham, UK. University of Birmingham.

Developing a PoS-tagged corpus using existing tools

Hrafn Loftsson*, Jökull H. Yngvason*, Sigrún Helgadóttir†, Eiríkur Rögnvaldsson‡

*School of Computer Science, Reykjavik University, Iceland
†The Árni Magnússon Institute for Icelandic Studies, Iceland
‡Department of Icelandic, University of Iceland, Iceland
{hrafn,jokull06}@ru.is, {sigruhel,eirikur}@hi.is

Abstract

In this paper, we describe the development of a new tagged corpus of Icelandic, consisting of about 1 million tokens. The goal is to use the corpus, among other things, as a new gold standard for training and testing PoS taggers. We describe the individual phases of the corpus construction, i.e. text selection and cleaning, sentence segmentation and tokenisation, PoS tagging with a combination method, error detection, and error correction. Furthermore, we discuss what problems have emerged, highlight which software tools have been found to be useful, and identify which tools are re-usable across different languages. Our preliminary evaluation results show that the error detection programs are effective and that our tagger combination method is crucial with regard to the amount of hand-correction that must be carried out in future work. We believe that our work will be of help to those wishing to develop similar resources for less-resourced languages.

1. Introduction

Language Technology (LT) for the Icelandic language has only existed for about a decade (Rögnvaldsson et al., 2009). In 2000, the Icelandic Frequency Dictionary (*IFD*) corpus (Pind et al., 1991) was the only LT resource. Since then, various resources have been developed, e.g. part-of-speech (PoS) taggers (Helgadóttir, 2005; Loftsson, 2008), a finite-state parser (Loftsson and Rögnvaldsson, 2007a), a lemmatizer (Ingason et al., 2008), and a morphological database (Bjarnadóttir, 2005). Icelandic is thus no longer a less-resourced language by any reasonable definition.

Before the work presented in this paper, the *IFD* corpus has been used to train and test PoS taggers (programs which automatically tag each word in running text with morphosyntactic information) on Icelandic text. The *IFD* corpus consists of about 590k tokens and all text fragments in the corpus were published for the first time in 1980–1989. In the tagset used, each character in a tag has a particular function. The first character denotes the word class and the remaining characters (up to 5) denote various morphological properties, for example, gender, number and case. The size of the *IFD* tagset is about 700 tags. The corpus was tagged with a special program that used grammatical rules and frequency information, derived from the manual tagging of 54k tokens (Briem, 1989). The automatic tagging was then hand-corrected line by line.

There are at least three problems associated with the use of the *IFD* corpus for developing PoS taggers. First, the corpus is relatively small in relation to the size of the tagset. Second, the underlying text has a strong literary bias. Third, the corpus text is 20–30 years old. Consequently, data-driven taggers, which have been trained on the corpus, may run into data sparseness problems, their accuracies may not be high enough when tagging different genres, and they may have difficulties when encountering recent linguistic phenomena which only occur in informal texts.

In this paper, we describe a work in progress, the development of a new tagged Icelandic corpus consisting of about 1 million tokens. It is intended that the corpus serve as a more suitable gold standard for developing PoS taggers than the *IFD* corpus. The new corpus will be tagged using the *IFD* tagset, although with some very minor modifications. The development of the corpus, henceforth referred to as the *GOLD*, consists of the following phases:

1. Text selection
2. Text cleaning
3. Sentence segmentation and tokenisation
4. PoS tagging
5. Error detection
6. Error correction
7. Evaluation

The emphasis in this work is on utilising existing tools and on automating the development process to as great an extent as possible. Phase 2 is semi-automatic and phases 3-5 are completely automatic. Our development platform is Ubuntu Linux and our software, which we intend to make open source, is written in shell scripts, Perl and Python.

In the remainder of this paper, we describe each of the above phases, discuss what problems have emerged, highlight which software tools have been found to be useful, and identify which tools are re-usable across different languages. Furthermore, we describe our preliminary evaluation results which show that the error detection programs are effective, and that our tagger combination method is crucial with regard to the amount of hand-corrections that inevitably must be carried out in order to make our *GOLD* a reliable standard.

2. Text selection

From 2004, work has been going on at the Árni Magnússon Institute for Icelandic Studies (AMI) to compile a tagged corpus of approximately 25 million tokens of texts (henceforth referred to as the *MIM* corpus) from different genres

Kepa Sarasola, Francis M. Tyers, Mikel L. Forcada (eds.)
7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010, Valetta, Malta, 23 May 2010, p. 53–60

of contemporary Icelandic, i.e. texts written from the year 2000 and onwards.
Each word will be accompanied by a PoS tag and a lemma and each text will have bibliographic information attached.
Emphasis has been placed on collecting written texts, both from printed sources and from texts found on the web. Just over 2% of the texts in the corpus will be transcribed spoken texts collected in connection with other projects (parliamentary speeches, conversations and interviews). Only texts that were available digitally were collected.
Permission has been sought from copyright owners of all the texts used. All copyright owners have signed a special declaration and agree that their material may be used free of licensing charges. In turn, AMI agrees that only 80% of each published texts are included and that copies of *MIM* are only made available under the terms of a standard licence agreement. The licence agreement will be modelled after the BNC User Licence[1].
MIM will be made available in two ways. Firstly, the corpus will be searchable on the website of the institute and, secondly, those that wish to use it in their own computers for language research or in LT can obtain a copy by signing the licence agreement. The corpus will be made available in TEI-conformant XML format (Burnard and Bauman, 2008).
The texts for the *GOLD* were selected from texts collected for *MIM*[2]. The texts were selected so as to reflect as far as possible the proportion of different types of text in *MIM*.
There are 13 different text types in the *GOLD*. In Table 1 these text types are listed together with the number of tokens of each type. As can be seen from the table this classification reflects to a great extent the origin of the texts. Texts were randomly sampled from textfiles from these different text types to the extent possible. None of the text samples in the *GOLD* is longer than 5,000 words. It should be noted that since permission has been granted by copyright owners of all the texts in the *MIM* the same applies to the text samples in the *GOLD*.

3. Text cleaning

The text samples for the *GOLD* were sampled after the *MIM* texts had been cleaned and prepared for tokenisation and tagging. In this section, we therefore give a short account of the process of cleaning the texts for *MIM*.
The texts obtained for *MIM* came in various formats. Text from most published books came as pdf-files. It is possible to extract text from pdf-files by using special programs: we mainly used a program developed by a member of our team. Some pdf-files are, however, rather difficult to handle and as a last resort we used optical character recognition software that is used for extracting text from scanned paper documents (ABBY FineReader: `http://finereader.abbyy.com/`). Some texts came in Word-documents which are easy to convert to text. Many texts were sampled directly from the Web so there was no need to change the format.
Text files varied greatly in quality. The cleanest text was obtained from the newspaper *Morgunblaðið*, taken directly from their database, classified by content. The text was sampled so as to reflect seasonal variation in topics under discussion. Their files, however, contained metadata that could be removed automatically. Some material was delivered as XML-files and we wrote a special program to extract clean text from those.
The text files obtained were either encoded using UTF-8 or ISO-8859-1 character encoding. It was decided that all texts in *MIM* should be converted to UTF-8 – hence all the texts in the *GOLD* use UTF-8 character encoding.
Texts from printed books and periodicals usually come with hyphenation. It was therefore necessary to run the texts through a program that joined the two parts of a word that had been split between lines. Various other measures had to be taken, either with automatic or semi-automatic means. We removed manually long quotations in a foreign language, long quotations from Old Icelandic texts and from new texts that we did not have permission to use, as well as footnotes, tables of content, indexes, reference lists, poems, tables and pictures.
Some texts were particularly difficult to handle and had to be fixed manually. This was particularly true for texts from the newspaper *Fréttablaðið* that were obtained as pdf-files. The text that was extracted from the files had to be rearranged to a certain extent.
As a final text cleaning step, we needed to carry out the following. Headings that do not end with an end-of-sentence marker (like a period, an exclamation mark or a question mark), but are only separated from the following text with a line-break, are common to many of the text types. For example:

```
Lokaorð
Bókmenntaverk verða að lögmálum ...
```

Here "Lokaorð" 'Epilogue' is the heading for the text below starting with "Bókmenntaverk".
The sentence segmentiser that we use (see the next section) does not consider such strings as being separate sentences, because in general a sentence can span multiple lines (with line-breaks in-between).
To handle this, we have written a program which searches for lines not ending with an end-of-sentence marker and whose following line starts with an upper-case letter. Such a pattern is a candidate for a heading followed by a new sentence. The program displays these occurrences to a user who then decides whether it is a correct candidate or not. If the candidate is correct, the program writes an additional line-break after the heading into the text file.

4. Sentence segmentation and tokenisation

Sentence segmentation and (word) tokenisation are often neglected, yet very important, pre-processing tasks. The former identifies where one sentence ends and another one

[1] `http://www.natcorp.ox.ac.uk/`. The crucial point in the licence agreement is that the Licencee can use his results freely but may not publish in print or electronic form or exploit commercially any extracts from the corpus other than those permitted under the fair dealings provision of copyright law.

[2] Note that, in contrast to *GOLD*, the tagging of *MIM* will not be manually checked and corrected.

begins, whereas the latter splits (for each sentence) a sequence of characters into meaningful linguistic units, like words, numbers and punctuation marks.

Developing a good sentence segmentiser/tokeniser is not a trivial task. For example, in the case of sentence segmentation, a period can serve as an end-of-sentence marker, as a decimal point, as a part of an abbreviation, etc. In the case of tokenisation, various things need to be accounted for, even when processing space-delimited languages. The most common of these is probably deciding when a punctuation character or other non-alphanumeric character should be part of the preceding character sequence or not. For example, a good tokeniser needs to be able to recognise occurrences of abbreviations, because otherwise they will be broken up into individual parts. Surprisingly little has been published regarding these important NLP tasks, but a good coverage of sentence segmentation and tokenisation can be found in (Grefenstette and Tapanainen, 1994; Palmer, 2000).

In our project, we rely on a sentence segmentiser and a tokeniser which are part of the *IceNLP* toolkit (Loftsson and Rögnvaldsson, 2007b). This toolkit is open source[3] and therefore we have been able to extend its functionality immediately when we have encountered new pre-processing errors (for example, by adding new abbreviations to the list of known abbreviations). Note that even though these tools were originally developed for processing Icelandic they should be applicable (or at least easily adjusted) to other related languages.

To a user, the segmentiser and the tokeniser is a single program which accepts an input file having a particular format and writes individual tokens to an output file in a given format. In our case, the input format is "free" (as opposed to one sentence per line) and the output format is one token per line with an empty line between sentences.

Obviously, the accuracy of the tokenisation is very dependent on the quality of the input. A typical first example occurs when the tokeniser splits a single word into two words, due to an erroneous additional space in the input. Consider the phrase "langan fangelsis dóm" 'long prison sentence'. In Icelandic "fangelsisdóm" is a single word, but since the tokeniser uses white space as a delimiter it has no chance of tokenising this correctly.

As a second example, note that a missing space (as opposed to an additional space) can also cause problems. Consider the string "c.Markmið" for which the correct string should have been "c. Markmið" 'c. Goal'. In this case, the tokeniser returns a single token for the string, because, generally, it allows a period to be a part of a token. However, a period is usually not a part of a string containing alpha characters unless the string is an abbreviation! Therefore, we have written a post-processing utility, which runs after the tokenisation, and fixes errors of this type (given a list of known abbreviations).

5. PoS tagging

Our PoS tagging phase consists of two parts. First, we tag the text with five individual taggers and then we apply a combination method to improve the tagging accuracy.

[3] http://icenlp.sourceforge.net

5.1. Individual taggers

The individual taggers that we use are (listed in descending order of accuracy when tagging Icelandic text): *IceTagger* (Loftsson, 2008; Loftsson et al., 2009), *Bidir* (Dredze and Wallenberg, 2008), *TnT* (Brants, 2000), *fnTBL* (Ngai and Florian, 2001), and *MXPOST* (Ratnaparkhi, 1996).

IceTagger is a linguistic rule-based tagger, specifically developed for tagging Icelandic text. It is a part of the *IceNLP* toolkit and thus open source. The other four taggers are data-driven, i.e. they learn a tagging model from a pre-tagged corpus. We obtained the bidirectional tagger *Bidir* from its developers, but *TnT*, *fnTBL* and *MXPOST* are downloadable from the web.

As discussed in Section 1, the *IFD* corpus has been used for training the data-driven taggers as well as developing *IceTagger*. The average tagging accuracy of the individual taggers used in the current project, measured using ten-fold cross-validation (and all the 700 tags in the tagset) against the *IFD* corpus, varies from around 89% to 92.5% (Helgadóttir, 2005; Loftsson, 2006; Loftsson et al., 2009).

All the taggers expect tokenised input. With the exception of *MXPOST*, the input format is one token per line with an empty line between sentences. *MXPOST* wants one sentence per line, and therefore we need to run a program which converts between these formats, both before and after tagging with *MXPOST*.

The *TnT* tagger is the only tagger that does not handle text in UTF-8 encoding. Thus, we needed to write a script which maps specific non-ASCII characters, like certain types of single quotes, to a character sequence that does not appear in the texts. For example, we map the single quotes ‘ and ’ to the characters strings BEGINSINGLEQ and ENDSINGLEQ, and then change the resulting file from UTF-8 encoding to ISO-8859-1 before *TnT* is run. When the tagger is finished we change the file back to UTF-8 and map the character strings back to the original quotes.

Once the tagging with the individual taggers has been carried out, we apply a few fixes to their output. For example, we make sure that the tag for every punctuation character is equivalent to the character itself, that number constants are always tagged with the same tag, and that abbreviations are always tagged in the same way.

5.2. Combined tagging

Finally, we apply a tagger combination method to the resulting output. Tagger combination methods are a means of correcting for the biases of individual taggers, and they are especially suitable when tagging a corpus, i.e. when effectiveness (accuracy) is more important than efficiency (running time). It has been shown that combining taggers will often result in a higher tagging accuracy than is achieved by individual taggers (Brill and Wu, 1998; van Halteren et al., 2001; Sjöbergh, 2003; Loftsson, 2006). The reason is that different taggers tend to produce different errors, and the differences can often be exploited to yield better results.

We use *CombiTagger* (Henrich et al., 2009), an open source system[4], for carrying out the combination automatically. We feed the output of the individual taggers into *Combi-*

[4] http://combitagger.sourceforge.net

Text type	Tokens	% of all tokens	Error candidates	True positives (%)	Tags corrected	% of tokens	Evaluation sample	Accuracy (%)
Newspaper 1[a]	251,814	24.9	1,781	78.8	479	0.6	1,526	92.3
Books	237,204	23.5	1,663	77.7	1,438	0.6	1,247	95.1
Blogs	135,350	13.4	1,036	85.0	1,083	0.8	720	90.0
Newspaper 2[b]	94,749	9.4	750	79.2	724	0.8	1,021	87.6
www.visindavefur.is[c]	92,218	9.1	682	87.5	828	0.9	970	92.8
Websites	65,177	6.5	430	70.5	386	0.6	694	94.0
Laws	41,319	4.1	259	84.9	254	0.6	434	94.0
School essays	34,372	3.4	213	85.0	200	0.6	359	94.2
Written-to-be-spoken	19,348	1.9	142	85.2	151	0.8	202	92.1
Adjudications	12,880	1.3	101	96.0	148	1.1	134	88.1
Radio news scripts[d]	11,198	1.1	68	73.5	58	0.5	117	92.3
Web media	8,522	0.8	40	62.5	29	0.3	89	95.5
E-mail	5,513	0.5	54	88.9	59	1.1	58	89.7
Total:	1,009,664	100.0	7,200	80.9	5,837	0.7	7,571	92.3

Table 1: Information about the various text types in the new gold standard

[a]The newspaper *Morgunblaðið*. Only 500 error candidates out of 1,781 have been inspected, yet.
[b]The newspaper *Fréttablaðið*.
[c]A website operated by the University of Iceland where the public can post questions on any subject.
[d]The Icelandic National Broadcasting Service.

Tagger through the command line (a GUI interface is also available) and write the combined tagging result to a new file. The default combination method (which we indeed use) is simple voting (majority voting)[5], where each tagger gets an equal vote when voting for a tag and the tag with the highest number of votes is selected.
In *CombiTagger*, the resolution of ties depends on the exact order of the tagger files fed into the program. For example, if there is a voting tie between two tagger groups[6] A and B then the tag proposed by group A is selected if one of its tagger's output has been loaded into *CombiTagger* before some output from group B. The order that we use is therefore descending order of accuracy as listed at the beginning of this section.
The whole process of tagging with the five taggers, applying fixes, and running *CombiTagger* is of course dependent on the size of the text being processed. To give an indication of the running time involved, it took 17 minutes (running on a Dell Precision M4300 2 Duo CPU, 2.20 GHz) processing the text type "Newspaper 2" which consists of 94,749 tokens (see Table 1), of which the *Bidir* tagger took close to 12 minutes and *CombiTagger* only 8 seconds. Thus, the whole tagging task processes about 93 tokens per second.

5.3. Discussion

We have not been able to find papers in the literature about corpus construction projects that follow our tagging method as described above, i.e. in which more than one tagger trained on the same corpus C_1 is used to tag another corpus C_2 (using the same tagset), followed by applying a tagger combination method. This is interesting because, as mentioned above, various researchers have demonstrated the effectiveness of applying combination tagging (Brill and Wu, 1998; van Halteren et al., 2001; Sjöbergh, 2003).
There may be several reasons for this. First, when one is confronted with the task of constructing the first tagged corpus for a language L, no data-driven tagger exists for L! Therefore, in order to apply a combination method for that task, one needs access to more than one tagger which is not data-driven, i.e. linguistic rule-based taggers, and, moreover, the taggers need to use the same tagset (if not, then a mapping between the tagsets needs to be possible). Linguistic rule-based taggers are, however, infrequent, let alone more than one such for a language L.
Second, if a corpus C_1 already exists for L, tagged with tagset T_1, then researchers may be reluctant to construct a new corpus C_2 for L, tagged with the same tagset. Indeed, projects have been carried out in which C_2 is tagged with a new tagset T_2 but, nevertheless, using taggers that have been trained on C_1. For example, in both (Zavrel and Daelemans, 2000) and (de Does and van der Voort van der Kleij, 2002), a stacking method was used to construct C_2, because T_2 was different from T_1 and/or the individual taggers used different tagsets. Stacking is a machine learning method which combines classifiers (taggers) by applying a classification algorithm using as features the tags chosen by the individual taggers. However, in this method, some amount of training data is still necessary, i.e. hand-annotated data that show which tag (from T_2) is correct for the combined classifier given the tags from the individual taggers.

6. Error detection

The output of the tagging phase is a single file consisting of tokens and the respective tags selected by *CombiTagger*.

[5]Other combination methods are possible, e.g. weighted voting or some user supplied voting algorithms.
[6]We use the term *tagger group* to denote a group of two or more taggers that agree on a particular tag.

Clearly, various tagging errors exist at this point, but, fortunately, some of the errors are systematic and can thus be detected automatically.

In a morphologically complex language like Icelandic, feature agreement, for example inside noun phrases, plays an important role. Therefore, of the total number of tagging errors existing in an Icelandic corpus, feature agreement errors are likely to be prevalent.

We use the noun phrase (NP), prepositional phrase (PP) and verb phrase (VP) error detection programs described by Loftsson (2009). In order to use these programs, a tagged corpus needs to be converted to one sentence per line and then parsed by *IceParser*, a finite-state parser which marks constituent structure and syntactic functions (Loftsson and Rögnvaldsson, 2007a)[7]. The output of the error detection programs is one error candidate per line. The parser and the error detection programs are not language independent. However, both are components of *IceNLP* and may thus be changed to work for other languages.

Let us consider two examples of error candidates. The first one is found by the noun phrase error detection program:

```
[NP [AP raunverulegt lhensf AP]
verðmæti nheo NP]
```

The above demonstrates a disagreement in case inside a noun phrase. The substring "[NP" denotes the beginning of a noun phrase, whereas "[AP" denotes the beginning of an adjective phrase (the AP is contained within the NP). The words in the phrase are "raunverulegt verðmæti" 'real value' and the corresponding PoS tags follow each word. The PoS tag "lhensf" denotes adjective (*l*), neuter (*h*), singular (*e*), nominative case (*n*), strong declension (*s*), and positive form (*f*). The PoS tag "nheo" denotes noun (*n*), neuter (*h*), singular (*e*) and accusative case (*o*). Thus, the noun is marked with accusative case but the adjective with nominative case.

The second example of an error candidate is found by the verb phrase error detection program:

```
{*SUBJ> [NP Ég fp1en NP] *SUBJ>}
[VP þekkti sfg3eþ VP]
```

This demonstrates a disagreement in person between the subject "Ég" 'I' and the main verb "þekkti" 'knew' (the substring "{*SUBJ>" denotes the beginning of the subject). The PoS tag "fp1en" denotes pronoun (*f*), personal (*p*), 1^{st} person (*1*), singular (*e*), nominative case (*n*). The PoS tag "sfg3eþ" denotes verb (*s*), indicative mood (*f*), active voice (*g*), 3^{rd} person (*3*), singular (*e*), past tense (*þ*). Thus, the subject is marked as 1^{st} person whereas the verb is marked as 3^{rd} person.

Both these error candidates signal a true error, but some of the candidates are *false positives* due to incorrect constituent marking by *IceParser*[8].

[7] When the PoS tags fed into *IceParser* are error free (hand-annotated), the F-measure of the parser for constituent structure is 96.7%. On the other hand, when PoS tags are produced by a tagger like *IceTagger* (thus containing some errors), the F-measure drops down to 91.9% (Loftsson and Rögnvaldsson, 2007a).

[8] Note that the set of false positives can be used to improve the parser!

7. Error correction

We have written a program which inspects each error candidate and finds the line number in the tagged file where the first token associated with the error candidate occurs (in the first example above, the word "raunverulegt" occurs in line number 36,527 in the tagged file). The program outputs each error candidate along with the corresponding line number.

Once the error detection programs and the program for generating line numbers have been run, we load both the error candidates along with the line numbers and the tagged file into a spreadsheet. At that point, we start inspecting each error candidate, find its instance in the tagged file and correct the error if needed. Note that each error candidate can result in a correction of more than one tag.

Obviously, the *GOLD* contains errors other than the ones pointed to by the error detection programs. One frequent tagging error occurs in the tag for the first word of a sentence. The reason is that each sentence in the *IFD* corpus, the training corpus for the individual taggers, starts with a lower case letter (except in the case of proper nouns)! Therefore, all the taggers (except *IceTagger* which is not data-driven) very often tag a word at the beginning of a sentence with a proper noun tag. Indeed, we have written a program which points to likely errors at the beginning of a sentence, but we have not yet been able to inspect those candidates.

Finally, note that, before a corpus is published as a reliable gold standard, it has to be read, gradually line by line, and all tagging errors corrected. Clearly, the error correction that we have already carried out will speed up that process. For the line-by-line inspection, we intend to use some corpus correction software, e.g. *Posedit* which is open source[9]. In this final step, it is important to correct tokenisation errors as well.

8. Evaluation

In this section, we present two kinds of evaluations. First with regard to error detection and, second, regarding tagging accuracy.

8.1. Error detection

The total number of tokens in our *GOLD* is 1,009,664. Running on the whole corpus, the error detection programs output 7,200 error candidates, which is 0.7% of the total number of tokens. As previously stated, the development of the *GOLD* is a work still in progress. We have not yet finished inspecting all the error candidates, but at the time of writing we have inspected 5,919 of the 7,200 candidates (82.2%)[10]. This has resulted in 5,837 error corrections (corrections of PoS tags) in texts containing 837,334 tokens, i.e. we have had to correct 0.7% of the tokens based on the error candidates already inspected.

[9] `http://elearning.unistrapg.it/corpora/posedit.html`

[10] For "Newspaper 1", we have only inspected 500 out of the 1,781 error candidates. As a result, we have made 479 corrections of tokens in texts containing 79,484 tokens, i.e. we have had to correct 0.6% of the tokens for "Newspaper 1" – see Table 1.

Information about the number of error candidates and the ratio of *true positives* for each text type can be seen in Table 1. The weighted average ratio of true positives is 80.9%. When applying the same error detection programs on the *IFD* corpus, Loftsson (2009) found 30.1% of the error candidates (448 out of 1489) to be true positives. The large difference can be explained by the fact that the *IFD* corpus has been corrected line by line whereas our *GOLD* has not – yet.

8.2. Tagging accuracy

In order to estimate the tagging accuracy of the individual text types, we performed the following. For each text type, we sampled every 100^{th} word (for "Newspaper 1", "Books" and "Blogs" we have only finished sampling 50-60% of the texts), i.e. 1% of the corresponding text. For each sampled word, we manually checked whether its tag was correct or not. A tag is correct if the whole tagstring (consisting of up to 6 letters) is correct.

The results can be found in the last two columns in Table 1. The 95% confidence limits for the estimated accuracy are acceptable for the largest samples (400 tokens or more, e.g. for Websites: $\pm 1.92\%$). The smallest samples (sample size less than 200) are, however, too small and need to be enlarged to give more reliable results. However, the resulting tagging accuracy achieved is very encouraging. For many of the text types, "Books", "Websites", "Laws" and "School essays", the tagging accuracy is $\geq 94\%$. Note that these texts contain continuous texts of good quality.

For "Books" the accuracy is above 95% which seems very good compared to the best tagging result of 93.5% using the *IFD* corpus (whose text is of similar type as our "Books") obtained by Loftsson (2006) when applying a simple voting method using five taggers. The main difference between our combination and the one used by Loftsson is twofold. First, we extend the dictionaries of *IceTagger* and *TnT* with part of the data from the Morphological Database of Icelandic Inflections (MDII) (Bjarnadóttir, 2005)[11]. By using data from the MDII, the ratio of unknown words in *IceTagger* and *TnT* is significantly lower than in the other three taggers. Note that due to the different unknown word ratio, we do not present tagging accuracy for unknown words and known words separately in Table 1. Second, we use the *Bidir* tagger in the combination instead of the *MBT* tagger (Daelemans et al., 1996) – the former is significantly more accurate than the latter when tagging Icelandic text.

The accuracy of four of the text types is $\leq$ 90%, i.e. "Blogs", "Newspaper 2", "Adjudications" and"E-mail"). For e-mails and blogs, this is not surprising, because the structure of sentences in these texts is sometimes unconventional and usually informal, and they often contain high frequency of foreign words and unconventional spelling. The relatively low accuracy of the adjudications texts can be explained by the fact the the word order is often "stilted", which the underlying tagging models sometimes have difficulties with.

For the "Newspaper 2" texts (*Fréttablaðið*), the taggers often have difficulties with foreign words (e.g. proper nouns), abbreviations, headlines, etc., and, moreover, these texts contain classified ads which can be difficult to tag for the individual taggers. Furthermore, as mentioned in Section 3, these texts were particularly difficult to handle and had to be fixed manually.

The tagging accuracy for the "Newspaper 1" text is much better compared to "Newspaper 2". As mentioned in Section 3, the "Newspaper 1" text was taken directly from the database of the publisher, classified by content and was therefore relatively clean. No classified ads were contained in the text. It is therefore not surprising that the tagging accuracy is better than for newspaper text that had to be extracted from pdf-files.

8.2.1. Error examples

The tagging errors found during our estimation of tagging accuracy are of various kinds. Most of them do not seem to be systematic, and hence we have not been able to write programs to correct them automatically. Below we give three examples of output from *CombiTagger* showing different kinds of errors found in the text type "Books". The first two columns show the word and the tag, respectively; in the third column we show an English gloss.

First, consider the sentence fragment:

```
það             fphen    it
virðist         sfm3en   seems
falla           sng      fit
vel             aa       well
að              c        to
staðalmynd      nven     stereotype-the
samfélagsins    nheeg    society's-the
```

This fragment contains two errors because "að staðalmynd" should be tagged as "að *aþ* staðalmynd *nveþ*", i.e. "að" as a preposition governing the dative case (instead of a conjunction (*c*)), and "staðalmynd" as a noun (*n*), feminine (*v*), singular (*e*) and dative case (*þ*) (instead of nominative case (*n*)). Only one of the five taggers, the *fnTBL* tagger, tags these two words correctly.

The second example demonstrates a long-distance dependency which is often difficult for taggers to handle correctly:

```
þær             fpvfn    they
fóru            sfg3fþ   went
að              cn       to
mennta          sng      educate
sig             fpkeo    themselves
```

Here "þær" is correctly tagged as a pronoun (*f*), personal (*p*), feminine (*v*), plural (*f*), nominative case (*n*), but the reflexive pronoun "sig", is incorrectly marked as masculine (denoted by the third letter (*k*) in the tag) and singular (denoted by the fourth letter (*e*)) instead of feminine and plural, because it refers to the word "þær". Only one of the five taggers, *IceTagger* in this case, tags the word "sig" correctly.

The last example demonstrates an incorrectly tagged wordclass:

```
Sá              faken    that
```

[11] This database is accessible from http://bin.arnastofnun.is/

æsti	sfg3eþ	upset (man)
verður	sfg3en	becomes
stöðugt	aa	consistently
ágengari	lvenvm	(more) agressive

Here "æsti" is tagged as a verb (the first letter in the tag *sfg3eþ* denotes a verb) but should be tagged as an adjective. None of the taggers tags this word correctly because this particular word form does not exist as an adjective in the dictionaries used by the taggers – it only exists as a verb.

9. Conclusion

In this paper, we have described the development of a new corpus, *GOLD*, of Icelandic text. *GOLD* consists of about 1 million tokens and will be used as a gold standard for training and testing PoS taggers. We have described the individual phases of the corpus development, text selection and text cleaning, sentence segmentation and tokenisation, PoS tagging, error detection and error correction, and, finally, evaluation results.

We have identified which tools have been of help during the development and which tools are usable across different languages. We believe that our work will be of help to researchers wishing to develop similar resources for less-resourced languages.

Our evaluation of tagging accuracy indicates that the error detection programs are effective and that the extra effort of applying five taggers and a combination method is crucial with regard to the amount of hand-correction that inevitably must be made in order to use *GOLD* as a reliable gold standard in the future.

Finally, since the methods applied in the construction of *GOLD* have been successful, we intend to use the same methods when tagging *MIM*, the corpus of 25 million tokens of modern Icelandic texts. The only difference is that we do not foresee a line-by-line inspection of the tagging for *MIM*!

Acknowledgments

The work in this paper was partly supported by both the Icelandic Student Innovation Fund and the Icelandic Research Fund, grant 090662012.

10. References

K. Bjarnadóttir. 2005. Modern Icelandic Inflections. In H. Holmboe, editor, *Nordisk Sprogteknologi 2005*. Museum Tusculanums Forlag, Copenhagen.

T. Brants. 2000. TnT: A statistical part-of-speech tagger. In *Proceedings of the* 6^{th} *Conference on Applied Natural Language Processing*, Seattle, WA, USA.

S. Briem. 1989. Automatisk morfologisk analyse af islandsk tekst. In *Papers from the Seventh Scandinavian Conference of Computational Linguistics*, Reykjavik, Iceland.

E. Brill and J. Wu. 1998. Classifier Combination for Improved Lexical Disambiguation. In *COLING-ACL '98:* 36^{th} *Annual Meeting of the Association for Computational Linguistics and* 17^{th} *International Conference on Computational Linguistics*, Montreal, Quebec, Canada.

L. Burnard and S. Bauman. 2008. Guidelines for Electronic Text Encoding and Interchange P5 edition. Text Encoding Initiative. http://www.tei-c.org/Guidelines/P5/.

W. Daelemans, J. Zavrel, P. Berck, and S. Gillis. 1996. MBT: a Memory-Based Part of Speech Tagger-Generator. In *Proceedings of the* 4^{th} *Workshop on Very Large Corpora*, Copenhagen, Denmark.

J. de Does and J. van der Voort van der Kleij. 2002. Tagging the Dutch PAROLE Corpus. In M. Theune, A. Nijholt, and H. Hondorp, editors, *Language and Computers – Computational Linguistics in the Netherlands 2001. Selected Papers from the Twelfth CLIN Meeting*. Rodopi, Amsterdam-New York.

M. Dredze and J. Wallenberg. 2008. Icelandic Data Driven Part of Speech Tagging. In *Proceedings of the* 46^{th} *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Columbus, OH, USA.

G. Grefenstette and P. Tapanainen. 1994. What is a word, What is a sentence? Problems of Tokenization. In *Proceedings of the* 3^{rd} *International Conference on Computational Lexicography*, Budapest, Hungary.

S. Helgadóttir. 2005. Testing Data-Driven Learning Algorithms for PoS Tagging of Icelandic. In H. Holmboe, editor, *Nordisk Sprogteknologi 2004*. Museum Tusculanums Forlag, Copenhagen.

V. Henrich, T. Reuter, and H. Loftsson. 2009. CombiTagger: A System for Developing Combined Taggers. In *Proceedings of the* 22^{nd} *International FLAIRS Conference, Special Track: "Applied Natural Language Processing"*, Sanibel Island, Florida, USA.

A. K. Ingason, S. Helgadóttir, H. Loftsson, and E. Rögnvaldsson. 2008. A Mixed Method Lemmatization Algorithm Using Hierachy of Linguistic Identities (HOLI). In B. Nordström and A. Ranta, editors, *Advances in Natural Language Processing,* 6^{th} *International Conference on NLP, GoTAL 2008, Proceedings*. Gothenburg, Sweden.

H. Loftsson and E. Rögnvaldsson. 2007a. IceParser: An Incremental Finite-State Parser for Icelandic. In *Proceedings of the* 16^{th} *Nordic Conference of Computational Linguistics (NoDaLiDa 2007)*, Tartu, Estonia.

H. Loftsson and E. Rögnvaldsson. 2007b. IceNLP: A Natural Language Processing Toolkit for Icelandic. In *Proceedings of Interspeech 2007, Special Session: "Speech and language technology for less-resourced languages"*, Antwerp, Belgium.

H. Loftsson, I. Kramarczyk, S. Helgadóttir, and E. Rögnvaldsson. 2009. Improving the PoS tagging accuracy of Icelandic text. In *Proceedings of the* 17^{th} *Nordic Conference of Computational Linguistics (NODALIDA-2009)*, Odense, Denmark.

H. Loftsson. 2006. Tagging Icelandic text: An experiment with integrations and combinations of taggers. *Language Resources and Evaluation*, 40(2):175–181.

H. Loftsson. 2008. Tagging Icelandic text: A linguistic rule-based approach. *Nordic Journal of Linguistics*, 31(1):47–72.

H. Loftsson. 2009. Correcting a PoS-tagged corpus using three complementary methods. In *Proceedings of the* 12^{th} *Conference of the European Chapter of the ACL (EACL 2009)*, Athens, Greece.

G. Ngai and R. Florian. 2001. Transformation-Based Learning in the Fast Lane. In *Proceedings of the* 2^{nd} *Conference of the North American Chapter of the ACL*, Pittsburgh, PA, USA.

D. Palmer. 2000. Tokenisation and Sentence Segmentation. In R. Dale, H. Moisl, and H. Somers, editors, *Handbook of Natural Language Processing*. Marcel Dekker, New York.

J. Pind, F. Magnússon, and S. Briem. 1991. *Íslensk orðtíðnibók [The Icelandic Frequency Dictionary]*. The Institute of Lexicography, University of Iceland, Reykjavik, Iceland.

A. Ratnaparkhi. 1996. A Maximum Entropy Model for Part-Of-Speech Tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PA, USA.

E. Rögnvaldsson, H. Loftsson, K. Bjarnadóttir, S. Helgadóttir, A. B. Nikulásdóttir, M. Whelpton, and A. K. Ingason. 2009. Icelandic Language Resources and Technology: Status and Prospects. In R. Domeij, K. Koskenniemi, S. Krauwer, B. Maegaard, E. Rögnvaldsson, and K. de Smedt, editors, *Proceedings of the NODALIDA 2009 Workshop Nordic Perspectives on the CLARIN Infrastructure of Language Resources*. Odense, Denmark.

J. Sjöbergh. 2003. Combining POS-taggers for improved accuracy on Swedish text. In *Proceedings of NoDaLiDa 2003*, Reykjavik, Iceland.

H. van Halteren, J. Zavrel, and W. Daelemans. 2001. Improving Accuracy in Wordclass Tagging through Combination of Machine Learning Systems. *Computational Linguistics*, 27(2):199–230.

J. Zavrel and W. Daelemans. 2000. Bootstrapping a Tagged Corpus through Combination of Existing Heterogeneous Taggers. In *Proceedings of the* 2^{nd} *International Conference on Language Resources and Evaluation (LREC 2000)*, Athens, Greece.