IceNLP: A Natural Language Processing Toolkit for Icelandic

Hrafn Loftsson[1], *Eiríkur Rögnvaldsson*[2]

[1]Department of Computer Science, Reykjavik University, Reykjavik, Iceland
[2]Department of Icelandic, University of Iceland, Reykjavik, Iceland
hrafn@ru.is, eirikur@hi.is

Abstract

Icelandic is a morphologically complex language, for which language technology resources are scarce. Only a few years ago, it could be stated that language technology was practically non-existent in Iceland. In this paper, we describe the development of an NLP toolkit for processing the language, the challenges faced and the decisions made during development. The current version of the toolkit consists of a tokeniser/sentence segmentiser, a morphological analyser, a linguistic rule-based tagger, and a finite-state parser. The development of our toolkit is a step towards building a Basic Language Resource Toolkit (BLARK) for the Icelandic language.

Index Terms: morphological analysis, part-of-speech tagging, finite-state parsing

1. Introduction

Language technology (LT) in Iceland was practically non-existent only a few years ago. In 1999, a report was written by a commission for the Icelandic Ministry of Education, Science and Culture on the status of LT in Iceland. The following points, borrowed from [1], were amongst the conclusions:

- Far fewer language technology tools were available for Icelandic than for more widely used languages in neighbouring countries.
- Basic research in language technology in Iceland hardly existed.

Since this report was written the situation has improved to some extent (cf. [2, 3]). Moreover, in the last 2-3 years, we have reacted by starting basic research and development in the field of Natural Language Processing (NLP) for the Icelandic language. As a result, we now possess a toolkit named *IceNLP* which offers basic text analysis and processing capabilities, i.e. tokenisation/sentence segmentation, morphological analysis, (linguistic rule-based) part-of-speech (POS) tagging, and (finite-state) shallow parsing. The development of *IceNLP*, which is freely available to the research community[1], is a step towards building a Basic Language Resource Toolkit (BLARK) for the Icelandic language, i.e. "the minimal set of language resources that is necessary to do any precompetitive research and education at all" [4].

In this paper, we describe the development of *IceNLP*, the challenges faced and the decisions made during development.

In Section 2, we briefly describe some essential features of the Icelandic language and previous work on tagging Icelandic text. The development of *IceNLP* is described in Section 3, and we conclude with a discussion on future enhancements in Section 4.

[1]The toolkit can be tested by visiting `http://nlp.ru.is`.

Char #	Category/ Feature	Symbol – semantics
1	Word class	**n**–noun
2	Gender	**k**–masculine, **v**–feminine, **h**–neuter, **x**–unspecified
3	Number	**e**–singular, **f**–plural,
4	Case	**n**–nominative, **o**–accusative, **þ**–dative, **e**–genitive
5	Article	**g**–with suffixed article
6	Proper noun	**m**–person, **ö**–place, **s**–other

Table 1: The semantics of the noun tags.

2. The Icelandic language and previous work

Icelandic is one of the Nordic languages. It has a basic subject-verb-object (SVO) word order, which is, nevertheless, relatively free. The language is morphologically rich, mainly due to inflectional complexity. Feature government and agreement plays an important role in Icelandic, e.g. the words of a noun phrase agree in gender, number and case, a preposition governs the case of the following noun phrase, a finite verb agrees with the subject in person and number, an adjectival verb complement agrees with the subject in gender and number, etc. A thorough description of the language can for example be found in [5].

As discussed in the introduction, basic research in LT for Icelandic has only taken its first steps the last few years. One might say, however, that the foundation for NLP in Iceland was built around 1990, with the construction of the *IFD* corpus, the only currently available POS tagged corpus for Icelandic [6]. This corpus is balanced and consists of about 590k tokens. The tagset used is large, i.e. it contains about 660 morpho-syntactic tags, and thus reflects the morphological complexity of the language. In this tagset, each character in a tag has a particular function. Table 1 shows the semantics of the noun tags.

To illustrate, consider the sentence *fallegu hestarnir stukku* (beautiful horses-the jumped). The corresponding tag for *fallegu* is *lkfnvf*, denoting adjective (*l*), masculine (*k*), plural (*f*), nominative (*n*), weak declension (*v*), positive (*f*); the tag for *hestarnir* is *nkfng*, denoting noun (*n*), masculine (*k*), plural (*f*), nominative (*n*), with suffixed definite article (*g*); and the tag for *stukku* is *sfg3fþ* denoting verb (*s*), indicative mood (*f*), active voice (*g*), 3^{rd} person (*3*), plural (*f*), and past tense (*þ*). Note the agreement in gender, number and case between the adjective and the noun, and the agreement in number between the subject and the verb (the verb also agrees with the subject in person, but since all nouns are 3^{rd} person by default, the person feature is not overtly expressed in this case).

In 2002-2004, Helgadóttir performed an Icelandic tagging experiment (ITE) [2], using state-of-the-art data-driven taggers, and the *IFD* corpus for training and testing. The highest accuracy, 90.4%, was obtained by the statistical *TnT* tagger [7]. This accuracy is considerably lower than the one achieved for related languages, e.g. Swedish where 93.6% accuracy was obtained in an experiment using the same taggers, a tagset consisting of 139 tags, and only 100k tokens of training data [8]. This difference in accuracy can be explained by the large Icelandic tagset, as well as by the fact that Icelandic is considerably more complicated, morphologically, than Swedish.

This overview shows that before we started our work on *IceNLP* in 2004, some groundwork had been carried out in POS tagging, but no parser had been published for the language, however.

3. IceNLP

Our NLP toolkit, *IceNLP*, consists of the following sequentially applied modules:

1. **Preprocessor**
   (a) Sentence segmentation
   (b) Tokenisation
2. **POS tagger – *IceTagger***
   (a) Morphological analyser – *IceMorphy*
      i. Lexicon lookup
      ii. Unknown word guessing
      iii. Tag profile gap filling
   (b) Disambiguation
      i. Local rules
      ii. Heuristics
3. **Finite-state parser – *IceParser***
   (a) Phrase structure module
   (b) Syntactic functions module

The first part uses well known techniques to split the input text into sentences and each sentence into individual tokens [9]. The POS tagger processes one tokenised sentence at a time and returns the contextually appropriate tag for each word. The parser assumes tagged input and returns a shallow phrase structure and labels indicating syntactic functions. The system is written as a collection of Java classes and comprises about 19,000 lines of code.

3.1. The linguistic rule-based tagger

When we started developing *IceNLP*, the first major challenge was to improve the tagging accuracy of Icelandic text. We concluded that data sparseness was to blame for the relatively low tagging accuracy in the ITE, i.e. the size of the tagset is large in relation to the size of the training corpus.

We hypothesised that higher tagging accuracy could be obtained by developing a linguistic rule-based tagger. However, we did not have the luxury of spending an enormous effort on development. For example, we were not in a position to develop a linguistic rule-based tagger using the framework of Constraint Grammar (CG) [10] – experience has shown that the effort needed to develop such a system can be measured in man-years (e.g. [11],[12]). Indeed, a prerequisite for the effectiveness of a CG system is the construction of an extensive lexicon and a morphological analyser, both of which demand large resources. On the other hand, CG systems, which both assign POS tags and labels for syntactic functions, have been shown to obtain very high accuracy [12]. The main challenge for us was therefore to develop a high accuracy linguistic rule-based tagger without spending years in development.

Hence, instead of using existing purpose-built software, we decided to develop our tagger, which we call *IceTagger*, from scratch. *IceTagger* consists of three main components: morphological analysis, local rules for initial disambiguation, and heuristics for further disambiguation.

The morphological analyser, *IceMorphy*, looks up each word in a lexicon and returns the *tag profile* (the set of possible tags) if the word is found. If the word is not found, a rule-based component (the unknown word guesser) is used to guess the tag profile, based on morphological/compound analysis and/or ending analysis. Since the lexicon used is not extensive (as discussed below), it has a number of *tag profile gaps*. Such a gap occurs when a particular word, listed in the lexicon, has some missing tags in its tag profile. The missing tag(s) might simply not have been encountered during the derivation of the lexicon. *IceMorphy* is able to fill in the gaps for words belonging to particular morphological classes.

The local rules remove illegitimate tags from words based on a local context. In contrast, the heuristics, can refer to a word which is not in the nearest neighbourhood, when disambiguating a particular word. The purpose of the heuristics is to perform grammatical function analysis, guess prepositional phrases, and use the acquired knowledge to force feature agreement where appropriate. If a word is still ambiguous after the application of the local rules and the heuristics, the default heuristic is simply to choose the most frequent tag according to frequency information derived from the *IFD* corpus.

IceTagger differs from a CG tagger in mainly two ways:

- Its main lexicon is automatically derived from the *IFD* corpus (the lexicon contains about 60,000 word forms), and is thus far from extensive. This means that *IceMorphy* mainly uses information derived from the *IFD* corpus. Moreover, the main lexicon has a number of *tag profile gaps* as discussed above.
- The number of constraints (local rules) are only about 175 (instead of in the thousands as is common in CG taggers), but, in addition, the tagger uses general heuristics as an aid in the disambiguation phase.

The emphasis on using heuristics during disambiguation is well suited for a language like Icelandic, in which feature agreement plays an important role. Instead of writing a large number of local rules, we have written a number of heuristics which guess syntactic functions (like subjects, objects and complements) of verbs and use the information to force feature agreement where appropriate [13].

Evaluation (using the *IFD* corpus) shows that *IceTagger* achieves 91.5% accuracy, which is equivalent to 11.5% error reduction rate when compared to the accuracy of the *TnT* tagger [14].

One of the advantages of building a tool like *IceTagger* is that we have complete control over the source, which can lead to an easier integration with other tools, for which the source is also available. For example, in order to improve the accuracy further, we have made *IceTagger* call a trigram tagger, *TriTagger*, (our re-implementation in Java of the *TnT* tagger) for words not fully disambiguated. Evaluation of the resulting tagger, which runs like a single tagger, shows an accuracy of

91.8% [15]. Additionally, we have used *IceTagger* in combination methods with four data-driven taggers, resulting in an accuracy of 93.5% [15].

Our tagging system as a whole, including the preprocessor, *IceMorphy* and *IceTagger* (excluding *TriTagger*) was developed in 7 man months, which can be considered short development time for a linguistic rule-based system. For a detailed description of *IceTagger* and *IceMorphy*, the reader is referred to [13, 14].

3.2. The finite-state parser

The second major challenge was parsing. Currently, no Icelandic treebank exists, and therefore developing a data-driven parser was not an option. Since full parsing methods are in most cases data-driven, we decided to develop a shallow parser. We selected the finite-state parsing mechanism, because of its success for various languages, its efficiency and robustness. More specifically, we decided to use the incremental finite-state approach, in which a parser comprises a sequence of finite-state transducers, which add syntactic information into the text in an incremental manner [16].

The Xerox Finite-State Tool (XFST) [17] is often used to develop finite-state parsers. The XFST includes extensions to the standard regular expression calculus, which simplify the creation of finite-state transducers for syntactic processing. However, our finite-state parser *IceParser* is implemented in Java and the (freely available) lexical analyser generator tool JFlex (http://jflex.de/). Each transducer is written in a separate file, which is compiled into Java code by JFlex. The resulting Java code is a deterministic finite-state automaton, along with actions to execute for each recognised pattern. The reason for not using the XFST for implementation is that the Java implementation allows for an easier integration of the parser with the other Java modules of the *IceNLP* tool.

The input to *IceParser* is text tagged with the *IFD* tagset. It produces output according to a shallow annotation scheme, specifically designed for this project [18]. The scheme consists of descriptions for annotation of both constituent structure and syntactic functions. Furthermore, we have developed a *grammar definition corpus*, a corpus consisting of 214 sentences (annotated using our scheme), representing the major syntactic constructions in Icelandic.

Accordingly, the parser comprises two main modules: a phrase structure module and a syntactic functions module. Here, we briefly describe these modules – consult [19] for a more detailed description.

The purpose of the phrase structure module is to add brackets and labels to input sentences to indicate constituent structure. This module consists of 14 transducers, which annotate phrases like AdvPs, APs, NPs, PPs, VPs and multiword expression phrases. The syntactic annotation is performed in a bottom-up fashion, i.e. the deepest constituents are analysed first. For example, AdvPs are marked before APs, which are in turn marked before NPs.

The purpose of the syntactic functions module is to add tags to denote grammatical functions. The input to the first transducer in this module is the output of the last transducer in the phrase structure module. This module consists of 8 transducers, which annotate syntactic functions like subjects, objects, verb complements and genitive qualifiers.

The transducers include numerous patterns, written to account for the relatively free word order of Icelandic. Apart from relying on category information (like the *word class*) in the POS tags, the patterns only use the grammatical *case* feature. The reason for not fully using the morphological information available in the POS tags, is that we want our parser to be used as a grammar checking tool, among other things. If the parser, for example, uses feature agreement to a great extent to mark phrases then it will not be possible for the grammar checking tool to point out feature agreement errors inside phrases. This is because the corresponding words would not have been recognised as one phrase by the parser, due to the lack of feature agreement!

Evaluation of *IceParser* (using a hand-annotated *gold standard* of 509 randomly selected sentences) shows that, for the case of perfect tagging (i.e. POS tags taken from the *IFD* corpus), the F-measure for constituent structure and syntactic functions is 96.7% and 84.3%, respectively. These results are the first parsing results published for the Icelandic language, but a comparison to related languages indicate that our parser performs well [19].

When *IceTagger* is used to tag the sentences in the *gold standard*, before *IceParser* is run, the overall F-measure for constituent structure drops from 96.7% to 91.9%, which is equivalent to about 5.0% reduction in accuracy. Similarly, the overall F-measure for syntactic functions drops from 84.3% to 75.3%, which is equivalent to about 10.7% reduction in accuracy. Thus, the accuracy of the syntactic functions module is more sensitive to tagging errors than the constituent module. This can be explained by the fact that the syntactic functions component relies to a much higher extent on the case feature, which is often responsible for the errors made by the tagger.

The advantage of implementing *IceParser* in a tool like JFlex is having full control of the source, which makes both optimisation and intergration easier. This has, for example, enabled us to implement a very efficient version of the parser (as discussed in [19] and demonstrated in Section 3.3). The disadvantage, however, is that the time spent on development is most likely longer than it would have taken by using the XFST. The whole parsing project, including the development of the shallow syntactic annotation scheme, the grammar definition corpus, evaluation and development of *IceParser* took a little more than one man-year.

3.3. Illustration

To illustrate the functionality of *IceNLP*, consider the input text: *Hann er mjög góður kennari* (he is (a) very good teacher), for which *IceTagger* returns the word-tag pairs:

Hann fpken er sfg3en mjög aa góður lkensf kennari nken

The POS tags denote a personal pronoun, a finite verb, an adverb, an adjective, and a noun, respectively.

The tagged text is then fed into *IceParser*, which returns:

*{*SUBJ> [NP Hann fpken NP] *SUBJ>}*
[VPb er sfg3en VPb]
*{*COMP< [NP [AP [AdvP mjög aa AdvP] góður lkensf AP] kennari nken NP] *COMP<}*

This output indicates that the noun phrase, consisting of the pronoun *Hann*, is a subject (*SUBJ>) of the finite verb *er* – the ">" in *SUBJ> indicates that the verb appears to the right of the subject. Furthermore, the noun phrase *mjög góður kennari*, which includes an adjective phrase including an adverb phrase, is a complement of the verb *er*, which is situated to the left of

the complement.

IceNLP tokenises, tags and parses the above sentence in only 4 msec. Our *gold standard* of 509 sentences (8281 tokens) is processed by *IceNLP* in less than 3 sec.

4. Conclusions and future work

We have described the development of *IceNLP*, an NLP toolkit for processing the Icelandic language, the challenges faced and the decisions made during development. The current version of the toolkit consists of a tokeniser/sentence segmentiser, a morphological analyser, a linguistic rule-based tagger, and a finite-state parser. The development of our toolkit is a step towards the goal of building a BLARK for the Icelandic language.

We have argued that the advantage of building a tagger from scratch, and a parser using a lexical analyser generator, is that we have full control of the source. In the future, we would like to use this advantage to further improve individual components of our toolkit.

We have demonstrated that the parsing accuracy is sensitive to the correctness of the POS tags in the input text. It is therefore important to try to improve the tagging accuracy. One of the BLARK components for Icelandic is the *Morphological Description of Icelandic* [3], a large resource currently containing about 250,000 lemmas and 5.5 million word forms. This resource could be used to generate a large lexicon for the purpose of POS tagging. Intuitively, using a larger (and more comprehensive) lexicon should result in higher accuracy, because of fewer occurrences of unknown words and fewer tag profile gaps. However, a larger lexicon could result in higher average ambiguity rate, which generally reduces the tagging accuracy. Thus, experimenting with different lexicon sizes is an interesting project.

Of the various morphological features available in the rich POS tags, the transducers of our finite-state parser only use the case feature in their patterns. This indicates that a smaller version of the tagset should be designed and the taggers (both *IceTagger* and the data-driven taggers) re-evaluated using this new tagset. When designing the smaller tagset, the main decision is what features of the large tagset can be left out without to much loss of information. Additionally, we would like to use the same large tagset, but develop a version of the parser which uses the features available in the POS tags to a greater extent.

5. Acknowledgements

The parsing part described in this paper was partly supported by the Icelandic Research Fund through the grant "Shallow parsing of Icelandic text".

6. References

[1] Arnalds, A., "Language Technology in Iceland", in H. Holmboe (ed.), *Nordisk Sprogteknologi 2003*. Museum Tusculanums Forlag, Copenhagen, 2003.

[2] Helgadóttir, S., "Testing Data-Driven Learning Algorithms for PoS Tagging of Icelandic", in H. Holmboe (ed.), *Nordisk Sprogteknologi 2004*. Museum Tusculanums Forlag, Copenhagen, 2004.

[3] Bjarnadóttir, K., "Modern Icelandic Inflections", in H. Holmboe (ed.), *Nordisk Sprogteknologi 2005*. Museum Tusculanums Forlag, Copenhagen, 2005.

[4] Krauwer, S., Maegaard, B., Choukri, K., and Jørgensen, L.-D., "BLARK for Arabic", Center for Sprogteknologi, NEMLAR project, 2004, `http://www.nemlar.org/Publications/BLARK-final_190906.pdf`.

[5] Þráinsson, H., "Icelandic", In E. König and J. Auwera (eds.), *The Germanic Languages*, Routledge, London, 1994.

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

[7] Brants, T., "TnT: A statistical part-of-speech tagger", in *Proceedings of the* 6^{th} *Conference on Applied natural language processing*, Seattle, WA, 2000.

[8] Megyesi, B., Data-driven Syntactic Analysis: Methods and Applications for Swedish, PhD dissertation, KTH, Stockholm, 2002.

[9] Palmer, D., "Tokenisation and Sentence Segmentation", in R. Dale, H. Moisl and H. Somers (eds.), *Handbook of Natural Language Processing*. Marcel Dekker, New York, 2000.

[10] Karlsson, F., Voutilainen, A., Heikkilä, J., and Anttila, A., *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Mouton de Gruyter, Berlin, 1995.

[11] Hagen, K., Johannessen, J., and Nøklestad, A., "A Constraint-Based Tagger for Norwegian", In C.-E. Lindberg and S.-N. Lund (eds.), 17^{th} *Scandinavian Conference of Linguistics. Odense Working Papers in Language and Communication*, 19, 31–48, University of Southern Denmark, Odense, 2000.

[12] Samuelsson, C. and Voutilainen, A, "Comparing a Linguistic and a Stochastic Tagger, in *Proceedings of the* 8^{th} *Conference of the European Chapter of the ACL (EACL)*, Madrid, 1997.

[13] Loftsson, H., "Tagging a Morphologically Complex Language Using Heuristics", in T. Salakoski, F. Ginter, S. Pyysalo and T. Pahikkala (eds.), *Advances in Natural Language Processing,* 5^{th} *International Conference on NLP, FinTAL 2006, Proceedings*, Turku, 2006.

[14] Loftsson, H., "Tagging Icelandic text using a linguistic and a statistical tagger", in *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the ACL (NAACL-HLT 2007)*, Rochester, NY, 2007.

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

[16] Aït-Mokhtar, S. and Chanod, J-P., "Incremental Finite-State Parsing", in *Proceedings of Applied Natural Language Processing*, Washington DC, 1997.

[17] Karttunen, L., Chanod, J-P., Grefenstette, G., and Schiller, A., "Regular expressions for language engineering", *Natural Language Engineering*, 2(4):305–328, 1996.

[18] Loftsson, H. and Rögnvaldsson, E., "A shallow syntactic annotation scheme for Icelandic text", Technical Report RUTR-SSE06004, Department of Computer Science, Reykjavik University, Reykjavik, 2006.

[19] Loftsson, H. and Rögnvaldsson, E., "IceParser: An Incremental Finite-State Parser for Icelandic", in *Proceedings of NoDaLiDa 2007*, Tartu, 2007.