Tagging Icelandic text using a linguistic and a statistical tagger

Hrafn Loftsson*
Department of Computer Science
Reykjavik University
Reykjavik IS-103, Iceland
hrafn@ru.is

Abstract

We describe our linguistic rule-based tagger *IceTagger*, and compare its tagging accuracy to the *TnT* tagger, a state-of-the-art statistical tagger, when tagging Icelandic, a morphologically complex language. Evaluation shows that the average tagging accuracy is 91.54% and 90.44%, obtained by *IceTagger* and *TnT*, respectively. When *tag profile gaps* in the lexicon, used by the *TnT* tagger, are filled with tags produced by our morphological analyser *IceMorphy*, *TnT's* tagging accuracy increases to 91.18%.

1 Introduction

In this paper, we use a *linguistic rule-based method* (LRBM) and a *data-driven method* (DDM) for tagging text in the morphologically complex Icelandic language.

We present a novel LRBM. The tagger based on this method, hereafter called *IceTagger*, uses about 175 local rules for initial disambiguation, and a set of heuristics, to force feature agreement where appropriate, for further disambiguation.

The average tagging accuracy of *IceTagger* is 91.54%, compared to 90.44% achieved by the *TnT* tagger, a state-of-the-art statistical tagger (Brants, 2000). *IceTagger* makes 11.5% less errors than *TnT*. On the other hand, when *tag profile gaps* in the lexicon, used by *TnT*, are filled with tags produced by *IceMorphy*, our morphological analyser, *TnT's* tagging accuracy increases to 91.18%. In that case, *IceTagger* makes 4.1% less errors than *TnT*.

The remainder of this paper is organised as follows: In Sect. 2, we describe the different tagging methods in more detail. Sect. 3 briefly describes the Icelandic language and the tagset. The components of *IceTagger* are described in Sect. 4, and evaluation results are presented in Sect. 5.

2 The tagging methods

DDMs use machine learning to automatically derive a language model from, usually, hand-annotated corpora. An advantage of the DDMs is their language and tagset independence property. Their disadvantage is that a tagged corpus is essential for training. Furthermore, the limited window size used for disambiguation (e.g. three words) can be responsible for some of the tagging errors.

One of the better known statistical data-driven tagger is the *TnT* tagger (written in C). The tagger uses a second order (trigram) Hidden Markov model. The probabilities of the model are estimated from a training corpus using maximum likelihood estimation. New assignments of part-of-speech (POS) to words is found by optimising the product of lexical probabilities ($p(w_i|t_j)$) and contextual probabilities ($p(t_i|t_{i-1}, t_{i-2})$) (where w_i and t_i are the i^{th} word and tag, respectively).

In contrast to DDMs, LRBMs are developed with the purpose of tagging a specific language using a particular tagset. The purpose of the rules is, usually, to remove illegitimate tags from words based on context. The advantage of LRBMs is that they do not

* The author is also affiliated with the Dept. of Computer Science, University of Sheffield, Sheffield, S1 4DP, UK.

Proceedings of NAACL HLT 2007, Companion Volume, pages 105–108,
Rochester, NY, April 2007.

rely (to the same extent as DDMs) on the existence of a tagged corpus, and rules can be written to refer to words and tags in the entire sentence. The construction of a linguistic rule-based tagger, however, has been considered a difficult and time-consuming task (Voutilainen, 1995).

One of the better known LRBMs is the Constraint Grammar (CG) framework (Karlsson et al., 1995), in which both POS and grammatical functions are tagged. The EngCG-2 tagger, developed over several years and consisting of 3,600 rules, has been shown to obtain high accuracy (Samuelsson and Voutilainen, 1997).

The development time of our LRBM (written in Java; including a tokeniser, *IceMorphy* and *IceTagger*) was 7 man-months, which can be considered a short development time for a LRBM. This is mainly due to the emphasis on using heuristics (see Sect. 4.3) for disambiguation, as opposed to writing a large number of local rules.

3 The Icelandic language and its tagset

The Icelandic language is one of the Nordic languages. The language is morphologically rich, mainly due to inflectional complexity. A thorough description of the language can, for example, be found in (Þráinsson, 1994).

The main Icelandic tagset, constructed in the compilation of the tagged corpus *Icelandic Frequency Dictionary* (*IFD*) (Pind et al., 1991), is large (about 660 tags) compared to related languages. In this tagset, each character in a tag has a particular function. Table 1 shows the semantics of the noun and the adjective tags.

To illustrate, consider the phrase "*fallegu hestarnir*" (beautiful horses). The corresponding tag for "*fallegu*" is "*lkfnvf*", denoting adjective, masculine, plural, nominative, weak declension, positive; and the tag for "*hestarnir*" is "*nkfng*" denoting noun, masculine, plural, nominative with suffixed definite article.

4 IceTagger

IceTagger consists of three main components: an unknown word guesser, local rules for initial disambiguation and heuristics for further disambiguation. Both the local rules and the heuristics have been developed using linguistic knowledge and tuned using a development corpus (described in Sect. 5).

Char #	Category/ Feature	Symbol – semantics
1	Word class	**n**–noun, **l**–adjective
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
5	Declension	**s**–strong, **v**–weak
6	Proper noun	**m**–person, **ö**–place, **s**–other
6	Degree	**f**–positive, **m**–comparative, **e**–superlative

Table 1: The semantics of the noun and the adjective tags.

4.1 The unknown word guesser

The purpose of our morphological analyser *IceMorphy*, which is used as an unknown word guesser by *IceTagger*, is to generate all appropriate tags for a given word. It uses a familiar approach to unknown word guessing, i.e. it performs morphological/compound analysis and ending analysis (Mikheev, 1997; Nakov et al., 2003). Additionally, *IceMorphy* includes an important module for handling *tag profile gaps* (for a thorough description of *IceMorphy*, consult (Loftsson, 2006a)).

A *tag profile gap* arises when a particular word, listed in a lexicon derived from a corpus, has some missing tags in its tag profile (set of possible tags). The missing tag(s) might just not have been encountered during the derivation of the lexicon (e.g. during training). For each noun, adjective or verb, of a particular morphological class, *IceMorphy* is able to fill in the gaps for the given word.

To illustrate, consider the word "*konu*" (woman), and let us assume that only the tag "*nveo*" (denoting noun, feminine, singular, accusative) is found in the lexicon. Based on the "*u*" morphological suffix and the accusative case of the tag, *IceMorphy* assumes the word belongs to a particular morphological feminine noun class, in which singular accusative, dative and genitive cases have the same word form. Con-

sequently, IceMorphy generates the correct missing tags: "*nveþ*" and "*nvee*".

4.2 Local rules

The purpose of a local rule is to eliminate inappropriate tags from words, based on a local context (a window of 5 words; two words to the left and right of the focus word). This reductionistic approach is common in rule-based taggers. It is, for example, used in the CG systems.

In principle, the local rules are unordered. The firing of a rule is, however, dependent on the order of the words in a sentence. A sentence to be tagged is scanned from left to right and all tags of each word are checked in a sequence. Depending on the word class (the first letter of the tag) of the focus word, the token is sent to the appropriate disambiguation routine, which checks a variety of disambiguation constraints applicable to the particular word class and the surrounding words. At each step, only tags for the focus word are eliminated.

The format of a local rule is: *If* <*condition*> *eliminate tag t*. A <*condition*> is a boolean expression, whose individual components can refer to lexical forms or individual characters (word class/morphological features) of tags. The following are examples of <*condition*> (L_1/R_1 and L_2/R_2 denote tokens one and two to the left/right of the focus word, F, respectively):

L_1.isOnlyWordClass(x) AND L_2.isOnlyWordClass(y)
R_1.isWordClass(x) OR R_2.isWordClass(y)
L_1.isWordClass(x) AND t.isCase(y) AND t.isGender(z)
R_1.lexeme.equals(x) AND F.isWordClass(y)

To exemplify, consider the sentence part: "*við vorum* ..." (we were ...). The word "*við*" can have the following five tags ("_" is used as a separator between tags): "*ao_aþ_fp1fn_aa_nkeo*". For illustration purposes, it is sufficient to point out that the first two tags denote prepositions governing the accusative and the dative cases, respectively. Since the following word is a verb ("*vorum*") and prepositions only precede nominals, a rule, with <condition>=R_1.isOnlyWordClass(Verb), eliminates preposition tags in this context, leaving only the tags "*fp1fn_aa_nkeo*".

The current version of our tagger has 175 local rules. The rules are written in a separate file and compiled to Java code.

4.3 The heuristics

Once local disambiguation has been carried out, each sentence is sent to a global heuristic module, consisting of a collection of algorithmic procedures. Its purpose is to perform grammatical function analysis, guess prepositional phrases (PPs) and use the acquired knowledge to force feature agreement where appropriate. We call these heuristics global because, when disambiguating a particular word, a heuristic can refer to a word which is not in the nearest neighbourhood.

The heuristics repeatedly scan each sentence and perform the following: 1) *mark PPs*, 2) *mark verbs*, 3) *mark subjects*, 4) *force subject-verb agreement*, 5) *mark objects*, 6) *force subject-object agreement*, 7) *force verb-object agreement*, 8) *force nominal agreement*, and 9) *force PP agreement*. Lastly, the default heuristic is simply to choose the most frequent tag according to frequency information derived from the *IFD* corpus. A detailed description of all the heuristics can be found in (Loftsson, 2006b).

5 Evaluation

For evaluation, we used the pairs of ten training and test corpora of the *IFD* corpus, produced by Helgadóttir (2004). We used the first nine of these test corpora for evaluation, but the tenth one was set aside and used as the development corpus for *IceTagger*.

For each test corpus (10% of the *IFD*) the corresponding training corpus (90% of the *IFD*) was used to deduce the lexicon(s) used by *TnT*, *IceTagger* and *IceMorphy*. When testing the two taggers, we thus made sure that the ratio of unknown words was (almost) the same.

The accuracy of a base tagger, which assigns each known word its most frequent tag, and the most frequent noun tag/proper noun tag to lower case/upper case unknown words, is 76.27% (see table 2).

The average tagging accuracy of *IceTagger* for all words is 91.54%, compared to 90.44% for *TnT* (see table 2). *IceTagger* makes 11.5% less errors than *TnT*[1].

In order to improve the tagging accuracy of *TnT*, we used the *tag profile gap* filling mechanism of *Ice-*

[1] *TnT* is very fast, it tags about 50,000 tokens/sec on a Dell Optiplex GX620 Pentium 4, 3.20 GHz. *IceTagger* tags about 2,700 tokens/sec.

Words	Base	TnT	TnT*	IceTagger
Unkn.	4.39%	71.68%	72.75%	75.09%
Known	81.84%	91.82%	92.53%	92.74%
All	76.27%	90.44%	91.18%	91.54%

Table 2: Average tagging accuracy of the various taggers.

Morphy in the following manner. Each record in the lexicon used by *TnT* consists of a word and the corresponding tags found in the training corpus. Additionally, to facilitate lexical probability calculations, each tag is marked by its frequency (i.e. how often the tag appeared as a label for the given word). We made *IceMorphy* generate a "filled" lexicon such that each generated missing tag was marked with the frequency 1[2]. We call the resulting tagger *TnT**. Indeed, when testing *TnT**, we obtained an overall average tagging accuracy of 91.18% (see table 2). *IceTagger* makes 4.1% less errors than *TnT**.

The development of *IceTagger/IceMorphy* took 7 man-months, but it has been worth the effort. First, *IceTagger* does make fewer errors than *TnT*, and its accuracy can probably be increased by improving its individual components. Secondly, we have used *IceTagger* in various tagger combination methods to further increase the tagging accuracy of Icelandic text (Loftsson, 2006c).

6 Conclusion

In this paper, we have compared the tagging accuracy of our linguistic rule-based tagger, *IceTagger*, to the accuracy of *TnT*, a state-of-the-art statistical tagger.

IceTagger uses only about 175 local rules, but is able to achieve high accuracy through the use of global heuristics along with automatic *tag profile gap* filling. The average tagging accuracy of *IceTagger* is 91.54%, compared to 90.44% obtained by the *TnT* tagger. On the other hand, we were able to obtain 91.18% accuracy using *TnT* along with the *tag profile gap* filling mechanism of *IceMorphy*.

In future work, we would like to improve individual components of *IceTagger* and *IceMorphy*, with the purpose of further increasing the tagging accuracy.

[2]This seems logical since the missing tags were not found in the training corpus and are, hence, infrequent.

References

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

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

H. Loftsson. 2006a. Tagging Icelandic text: A linguistic rule-based approach. Technical Report CS-06-04, Department of Computer Science, University of Sheffield.

H. Loftsson. 2006b. Tagging a Morphologically Complex Language Using Heuristics. In T. Salakoski, F. Ginter, S. Pyysalo, and T. Pahikkala, editors, *Advances in Natural Language Processing,* 5^{th} *International Conference on NLP, FinTAL 2006, Proceedings*, Turku, Finland.

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

A. Mikheev. 1997. Automatic Rule Induction for Unknown Word Guessing. *Computational Linguistics*, 21(4):543–565.

P. Nakov, Y. Bonev, G. Angelova, E. Cius, and W. Hahn. 2003. Guessing Morphological Classes of Unknown German Nouns. In *Proceedings of Recent Advances in Natural Language Processing*, Borovets, Bulgaria.

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

H. Þráinsson. 1994. Icelandic. In E. König and J. Auwera, editors, *The Germanic Languages*. Routledge, London.

C. Samuelsson and A Voutilainen. 1997. Comparing a Linguistic and a Stochastic tagger. In *Proceedings of the* 8^{th} *Conference of the European Chapter of the ACL (EACL)*, Madrid, Spain.

A. Voutilainen. 1995. A syntax-based part-of-speech analyzer. In *Proceedings of the* 7^{th} *Conference of the European Chapter of the ACL (EACL)*, Dublin, Ireland.