Tagging a Morphologically Complex Language Using an Averaged Perceptron Tagger: The Case of Icelandic

Hrafn Loftsson[1], *Robert Östling*[2]

(1) School of Computer Science, Reykjavik University, Iceland
(2) Department of Linguistics, Stockholm University, Sweden

hrafn@ru.is, robert@ling.su.se

ABSTRACT

In this paper, we experiment with using Stagger, an open-source implementation of an Averaged Perceptron tagger, to tag Icelandic, a morphologically complex language. By adding language-specific linguistic features and using IceMorphy, an unknown word guesser, we obtain state-of-the-art tagging accuracy of 92.82%. Furthermore, by adding data from a morphological database, and word embeddings induced from an unannotated corpus, the accuracy increases to 93.84%. This is equivalent to an error reduction of 5.5%, compared to the previously best tagger for Icelandic, consisting of linguistic rules and a Hidden Markov Model.

KEYWORDS: Averaged Perceptron, Part-of-Speech Tagging, Morphological Database, Linguistic Features, Word Embeddings.

1 Introduction

Part-of-Speech (PoS) tagging is the task of assigning labels, denoting word classes and morphosyntactic features, to words in running text.

The state-of-the-art tagging accuracy for English, using supervised methods, is above 97%, e.g. (Collins, 2002; Toutanova et al., 2003; Giménez and Màrquez, 2004; Shen et al., 2007), and newest results using semi-supervised methods are close to 97.5% (Spoustová et al., 2009; Søgaard, 2011).

English is a morphologically simple language and most work on tagging English has used the 48 tags in the Penn TreeBank tagset (Marcus et al., 1993). Many morphologically complex languages use much larger tagsets, which encode detailed morphosyntactic features, in addition to the basic PoS categories. For example, 1100 tags appear in the Prague Dependency Treebank 2.0 (Spoustová et al., 2009), over 1000 tags appear in a Polish tagset (Radziszewski, 2013), a tagset for Bulgarian contains 680 tags (Georgiev et al., 2012), and a reduced version of an Icelandic tagset contains 565 tags (Loftsson et al., 2011).

The state-of-the-art for many morphologically complex languages is well below the 97+% figures for English (Spoustová et al., 2009; Loftsson et al., 2011; Radziszewski, 2013). A notable exception is the work of Georgiev et al. (2012) for Bulgarian, in which about 98% accuracy was achieved, using bidirectional sequence classification (Shen et al., 2007), in combination with linguistic rules and a morphological lexicon (which included *all* the words in the test set, resulting in 0% unknown word rate).

In this paper, we experiment with using Stagger (Östling, 2012), an open-source implementation of the Averaged Perceptron tagger by Collins (2002), to tag Icelandic. Our motivation for applying Stagger to Icelandic is threefold. First, the training time in Stagger is relatively short compared to other data-driven methods like (Lafferty et al., 2001; Giménez and Màrquez, 2004; Shen et al., 2007), when using a large tagset. Second, Stagger has been shown to obtain state-of-the-art results for Swedish (Östling, 2012), a closely related language to Icelandic. Third, Stagger is implemented in Java and thus allows easy integration to components in the Java-based IceNLP toolkit (Loftsson and Rögnvaldsson, 2007).

By training and testing on the Icelandic Frequency Dictionary (IFD) corpus (Pind et al., 1991), using a tagset of 565 tags, we obtain state-of-the-art tagging accuracy for Icelandic: *i)* 92.82% by adding language-specific linguistic features to the base feature set of Stagger, and using IceMorphy (Loftsson, 2008) as an unknown word guesser; and *ii)* 93.84%, by adding data from a full form database of Icelandic inflections (Bjarnadóttir, 2012), and word embeddings generated with a neural network language model (Collobert and Weston, 2008).

It is notable that our best tagging results using Stagger (93.84%) beat the previous best results for Icelandic (93.48%), obtained by using a hybrid tagger based on a linguistic rule-based method and a Hidden Markov Model (Loftsson et al., 2011). The increase in accuracy is equivalent to an error reduction of 5.5%.

This paper is structured as follows. In Section 2, we describe Stagger. We discuss previous work in tagging Icelandic in Section 3. Our development and evaluation work is described in Section 4. Error analysis is carried out in Section 5, and we conclude in Section 6.

2 Stagger

Stagger was originally developed for Swedish, where it achieves the state-of-the-art accuracy of 96.57%, using a tagset of size 150 (Östling, 2012). In this section, we give an overview of Stagger, i.e. the underlying Averaged Perceptron algorithm, the base feature set, PoS filters, and word embeddings.

2.1 Averaged Perceptron

The Averaged Perceptron algorithm of Collins (2002) uses a discriminative, feature-rich model that can be trained efficiently. Recent research also shows that the algorithm, given a good search method, can be used for PoS tagging with state-of-the-art accuracy (Shen et al., 2007; Tsuruoka et al., 2011).

Features are modeled using *feature functions* of the form $\phi(h_i, t_i)$ for a history h_i and a tag t_i, in the way pioneered by Maximum Entropy models (Berger et al., 1996; Ratnaparkhi, 1996). The history h_i is a complex object modeling different aspects of the sequence being tagged. It may contain previously assigned tags in the sequence to be annotated, as well as other contextual features such as the form of the current word, or whether the current sentence ends with a question mark. Intuitively, the job of the training algorithm is to find out which feature functions are good indicators that a certain tag t_i is associated with a certain history h_i.

A model consists of feature functions ϕ_s, each paired with a *feature weight* α_s which is to be estimated during training. The scoring function is defined over entire sequences, which in a PoS tagging task typically means sentences. For a sequence of words w of length n in a model with d feature functions, the scoring function is defined as:

$$score(w, t) = \sum_{i=1}^{n} \sum_{s=1}^{d} \alpha_s \phi_s(h_i, t_i)$$

The highest scoring sequence of tags:

$$\bar{t} = \arg\max_t score(w, t)$$

can be computed, for example, using the Viterbi algorithm or (as in in our case) a beam search of width 8.

Training the model is done in an error-driven fashion: tagging each sequence in the training data with the current model, and adding to the feature weights the difference between the corresponding feature function for the correct tagging, and the model's tagging.

Algorithm 1 shows one iteration of the perceptron training algorithm over the training set T of sequences. The model is initialized to $\alpha_s = 0$ for all s. Collins (2002) shows that rather than using the estimated model parameters α_s directly when tagging data outside the training set, both tagging accuracy and the speed of convergence can be improved by using values of α_s averaged during the training process. In our case, we average the weights after every 4096 training instances (sentences), which seems to strike a good balance between efficiency and accuracy.

Algorithm 1 Perceptron training iteration.

for all $\tilde{h}, \tilde{t} \in T$ **do**
  $\bar{t} \leftarrow \arg\max_t score(w, t)$
  for $i \leftarrow 1..n$ **do**
    for $s \leftarrow 1..d$ **do**
      $\alpha_s \leftarrow \alpha_s + \phi_s(\tilde{h}_i, \tilde{t}_i) - \phi_s(\bar{h}_i, \bar{t}_i)$
    end for
  end for
end for

2.2 Features

Stagger uses a basic feature set similar to that of Ratnaparkhi (1996). All of the basic features are binary, i.e. with the value 0 or 1.

Table 1 shows the templates on which the basic features used in Stagger are based. One instance of the first template may be:

$$\phi_s(h,t) = \begin{cases} 1 & \text{if } t = \text{c} \wedge w_i = \text{og} \wedge i \neq n \\ 0 & \text{otherwise} \end{cases}$$

or in other words, 1 if the current tag is "c" (conjunction), the current word (w_i) is "og" 'and', and the current word is not the last in the sentence.

History-independent features
$t_i = x, w_i = y, i = n$
$t_i = x, w_i = y, i = 1, c(i)$
$t_i = x, w_{i-1} = y, w_i = z$
$t_i = x, w_i = y, w_{i+1} = z$
$t_i = x, w_{i-1} = y, w_i = z, w_{i+1} = u$
$t_i = x, w_{i-2} = y, w_{i-1} = z, w_i = u$
$t_i = x, w_i = y, w_{i+1} = z, w_{i+2} = u$
$t_i = x, w_{i+\{-2,-1,1,2\}} = y$
$t_i = x, prefix_{\{1,2,3,4\}}(w_i) = y, i = 1, c(i)$
$t_i = x, suffix_{\{1,2,3,4,5\}}(w_i) = y, i = 1, c(i)$
$t_i = x, k(i), \text{"-"} \in w_i$
$t_i = x, k(i), k(i+1)$
History-dependent features
$t_i = x, t_{i-1} = y$
$t_{i-2} = x, t_{i-1} = y, t_i = z$
$t_i = x, t_{i-1} = y, w_i = z$
$t_i = x, t_{i-1} = y, w_i = z, w_{i+1} = u$

Table 1: Templates for the basic features of Stagger. t_i is the tag at position i in the sequence (of length n). w_i is the lower-cased word at position i. $k(i)$ is the type of token i (e.g. digits, Latin letters, symbol). $c(i)$ is the capitalization of token i (upper, lower, N/A). $prefix/suffix_k(w_i)$ is the k-letter prefix/suffix of word w_i. x, y, z, u are constants, which in any given feature function has a fixed value.

The features can be divided into two categories: those who describe previously assigned tags in the sequence (history-dependent features), and those who do not. Making this distinction can lead to a large speed increase, since the history-independent feature functions only have to be evaluated once for each tag and word sequence, while the history-dependent ones must also be evaluated for every history.

In addition to the basic feature set detailed above, we also performed an experiment adding Collobert and Weston (2008) embeddings through feature functions of the following form:

$$\phi_{x,j}(h_i, t_i) = \begin{cases} \text{CW}(w_i)_j & \text{if } t_i = x \\ 0 & \text{if } t_i \neq x \end{cases}$$

where $\text{CW}(w_i)_j$ is the j:th dimension of word w_i's Collobert and Weston embedding. The embeddings are described in Section 2.4.

2.3 PoS filter

Ratnaparkhi (1996) observed that both accuracy and speed can be improved by not considering *all* tags for every word. We use the following basic method for determining which tags to consider for a given word. If the word w is known, i.e. occurs in the training data, only the tags found during training for w are considered (we refer to this set of possible tags as the tag profile for w). Other words are limited to manually created tags from the set of open word classes.

There is, however, one complication: during training, all words are known. Using the tag filtering approach exactly as described above during training would give unrealistically good accuracy on the training data, but since the perceptron algorithm learns from its errors, this tends to lead to *decreased* accuracy on other data. To prevent this, words occurring between 1 and 3 times in the training data may be assigned tags from the union of the set of tags they occur with in the training data, and the set of open word class tags.

In Section 4.3.1, we show how the PoS filter can be further improved using a morphological analyzer, and in Section 4.3.2 using a lexicon.

2.4 Collobert and Weston embeddings

Neural network language models have been used to create word embeddings, vectors in $\mathbb{R}^d$ that represent syntactic and semantic properties of words based on their distributional properties (Collobert and Weston, 2008; Collobert et al., 2011). Collobert and Weston (C&W) embeddings have been shown to improve accuracy when used as features in named entity recognition and shallow parsing (Turian et al., 2010) as well as PoS tagging (Östling, 2012). This amounts to a type of semi-supervised learning, since properties of words learned from an unannotated text corpus are used to benefit a supervised learning task. In the case of PoS tagging, C&W embeddings are useful because words of the same word class tend to receive similar embeddings due to the similar syntactic contexts in which they occur.

Stagger can optionally be provided with word embeddings, for use during training.

3 Icelandic and Previous Tagging Work

The Icelandic language is morphologically rich, mainly due to inflectional complexity. Nouns can appear in 16 inflectional forms, depending on number, case and the presence or absence of

a suffixed definite article. Adjectives can have as many as 120 inflectional forms, depending on gender, number, case, strong or weak declension, and degree. Verbs conjugate in person, number, tense, mood and voice, and can have as many as 107 inflectional forms (Bjarnadóttir, 2012).

From a syntactic point of view, Icelandic has a basic subject-verb-object (SVO) word order, but, in fact, the word order is fairly flexible, because morphological endings carry a substantial amount of syntactic information. However, the word order in Icelandic is not as flexible as is allowed in some Slavic languages. As pointed out by Dredze and Wallenberg (2008a), this combination of morphological complexity and syntactic constraints makes Icelandic a good test case for data-driven tagging methods.

3.1 Resources

All PoS taggers for Icelandic (see Section 3.2) have been trained/developed and tested using the 10 pairs of training and test sets of the Icelandic Frequency Dictionary (IFD)[1] (Pind et al., 1991), a balanced corpus of about 590,000 tokens. All 100 text fragments in the IFD were published for the first time in 1980–1989. The corpus comprises five categories of texts, i.e. Icelandic fiction, translated fiction, biographies and memoirs, non-fiction and books for children and youngsters. The tagset used in the compilation of the IFD has become the standard tagset for tagging Icelandic. It contains about 700 possible tags, of which 639 appear in the IFD. Thus, the tagset mirrors the morphological complexity of the language.

The PoS tags are character strings, in which each character has a particular function. The first character denotes the *word class*. For each word class there is a predefined number of additional characters (at most six), which describe morphological features. To illustrate, consider the word form "fiskur" 'fish'. The corresponding tag is *nken*, denoting noun (*n*), masculine (*k*), singular (*e*), and nominative (*n*) case. Table 2 shows an example of the declension for the lemma "fiskur" 'fish', in singular and plural (without the definite article) and the corresponding PoS tags.

	Singular		Plural	
Case	Form	Tag	Form	Tag
Nominative	fiskur	nken	fiskar	nkfn
Accusative	fisk	nkeo	fiska	nkfo
Dative	fiski	nkeþ	fiskum	nkfþ
Genitive	fisks	nkee	fiska	nkfe

Table 2: The declension for the noun lemma "fiskur" 'fish'.

The other main resource used in the development of Icelandic taggers is the Database of Icelandic Inflection[1] (Bjarnadóttir, 2012). Its Icelandic abbreviation is *BÍN* and henceforth we use that term. BÍN contains about 270,000 paradigms, with about 5.8 million inflectional forms.

3.2 Previous tagging work

A few years ago, no PoS tagger existed for tagging Icelandic. Now, however, various PoS taggers have been developed, in particular data-driven taggers (Helgadóttir, 2005; Dredze and Wallenberg, 2008b; Rögnvaldsson and Helgadóttir, 2011), a rule-based tagger (Loftsson, 2008), and a hybrid tagger (Loftsson et al., 2009).

[1]Available at `http://www.malfong.is`

In recent work on tagging Icelandic, the tagset has been reduced, by removing named-entity classification for proper nouns and labeling all number constants with a single tag – resulting in 565 tags appearing in the changed version of the IFD. Moreover, the newest evaluation has been carried out on a corrected version of the IFD corpus (Loftsson et al., 2011). Nevertheless, the PoS (morphosyntactic) tagging of Icelandic texts has turned out to be a challenging task. The reason is, for example, that the tagset is large in relation to the size of the available training corpus, and the tagset makes very fine distinctions.

Table 3 shows the accuracy of three taggers, TriTagger, IceTagger, and HMM+Ice+HMM[2], used by Loftsson et al. (2011), when tagging the (corrected version of the) IFD corpus (565 tags) using 10-fold cross-validation. Tagging accuracy is shown both with and without using data from BÍN. Loftsson et al. (2011) used the 10^{th} fold of the IFD for development, and the figures are thus based on the average of the first nine folds. Data from BÍN was used to extend the dictionaries used by the taggers, thereby reducing the unknown word rate (UWR) from 6.8% (when not using data from BÍN) down to 1.1%. According to Loftsson et al. (2011), when adding data from BÍN, only "hard" unknown words remain – mostly proper nouns and foreign words.

Without data from BÍN			
Tagger	Unknown	Known	All
TriTagger	72.98	92.18	90.86
IceTagger	77.02	93.07	91.98
HMM+Ice+HMM	77.47	93.84	**92.73**
With data from BÍN			
Tagger	Unknown	Known	All
TriTagger	65.84	92.22	91.93
IceTagger	63.47	93.11	92.78
HMM+Ice+HMM	60.50	93.85	**93.48**

Table 3: Average tagging accuracy (%) of three taggers when tagging the IFD corpus using 10-fold cross-validation. Average unknown word rate (UWR) in testing is 6.8% when not using data from BÍN, compared to 1.1% when using data from BÍN.

Below, we briefly describe the three taggers used by Loftsson et al. (2011). The first tagger, TriTagger, is a Hidden Markov Model (HMM) tagger, a re-implementation of the well-known TnT tagger (Brants, 2000).

The second tagger, IceTagger (Loftsson, 2008), is a linguistic rule-based tagger, but it derives its dictionaries from a training corpus. It is reductionistic in nature, i.e. it removes inappropriate tags from the tag profile for a specific word in a given context. An important part of IceTagger is its unknown word guesser, IceMorphy. It guesses the tag profile for unknown words by applying morphological analysis and ending analysis. In addition, IceMorphy can fill in the *tag profile gaps*[3] in the dictionary for words belonging to certain morphological classes (Loftsson, 2008).

The third tagger, HMM+Ice+HMM (Loftsson et al., 2009), is a hybrid tagger, comprising both IceTagger and TriTagger. It works as follows. First, TriTaggger (the HMM) performs initial

[2]These taggers are part of the open-source IceNLP toolkit, available at `http://icenlp.sourceforge.net`

[3]A tag profile gap for a word occurs when a tag is missing from the tag profile. This occurs, for example, if not all possible tags for a given word are encountered during training.

disambiguation only with regard to the word class. Then, the rules of IceTagger are run. Finally, the HMM disambiguates words that IceTagger is not able to fully disambiguate.

Before the work described in this paper (see Section 4), the HMM+Ice+HMM tagger was the state-of-the-art tagger for Icelandic, obtaining an overall accuracy of 93.48% and 92.73%, with and without using data from BÍN, respectively (see Table 3).

One additional tagger, Bidir (Dredze and Wallenberg, 2008b,a), is relevant to our work. It is based on the bidirectional sequence classification method by Shen et al. (2007). In Bidir, the learning phase is divided into separate learning problems. First, a word class (WC) tagger classifies a word according to the word class. Then the tagger only considers and evaluates tags that are consistent with the predicted word class. Secondly, a case tagger (CT) retags the case of nouns, adjectives and pronouns, given the predicted tags from the WC tagger. This combination resulted in an accuracy of 92.06%, when using the uncorrected version of the IFD corpus and the full 639 tags. For comparison, the HMM+Ice+HMM tagger obtains an accuracy of 92.31% when tagging the IFD corpus under these same settings (Loftsson et al., 2009).

4 Development and Evaluation

In this section, we describe various experiments carried out with the goal of obtaining state-of-the-art tagging accuracy for Icelandic using Stagger.[4] Following previous work, in all the experiments we trained and tested on folds 1-9 of the IFD, whereas fold 10 was used for development. Furthermore, following Loftsson et al. (2011), we used the corrected version of the IFD corpus and the reduced tagset of 565 tags appearing in the IFD (see Section 3.2). During training, we used 12 iterations.[5]

4.1 Generic Stagger

In the first experiment, we evaluated the generic version of Stagger, with only the addition of a list of open word class tags from the IFD tagset.

When comparing the results of experiment no. 1, shown in Table 4, to the results without BÍN data in Table 3, we can see that Stagger performs better than a pure HMM model (TriTagger). Stagger obtains an accuracy of 91.29% for all words, whereas TriTagger obtains 90.86%.

On the other hand, Stagger's accuracy is lower than both IceTagger (91.98%) and HMM+Ice+HMM (92.73%). It is notable, however, that Stagger's accuracy for known words (93.11%) is higher than the corresponding accuracy in both TriTagger (92.18%) and IceTagger (93.07%). This is already a promising result, given the fact that IceTagger is developed using linguistic knowledge.

In contrast, Stagger's accuracy for unknown words (66.29%) is much lower compared to the other three taggers. In Section 4.3, we experiment with increasing the accuracy of unknown words, and with reducing the UWR.

4.2 Adding linguistic features

The base features of Stagger are language-independent. In the second experiment, we added language-specific linguistic features (LF), that use particular properties of Icelandic.

[4]The additions we made to the generic version of Stagger are available at `http://www.ling.su.se/stagger`

[5]For each fold of about 530,000 tokens, training time on an Intel i5, 2.50GHz system, with 8G RAM, is 45-90 minutes, depending on the type of experiment carried out in the subsections below.

Exp.	Tagger	Unknown words	Known words	All words	Error Δ^a	Error Δ^b
1	Stagger	66.29	93.11	91.29		
2	Stagger+LF	66.06	93.26	91.42	1.5	1.5
3	Stagger+LF+IceMorphy	77.03	93.97	**92.82**	16.3	17.7
4	Stagger+LF+IceMorphy+BÍN	61.45	94.02	**93.70**	12.3	27.7
5	Stagger+LF+IceMorphy+BÍN+WE	61.99	94.15	**93.84**	2.2	29.3

[a] Error reduction with regard to the previous experiment.
[b] Error reduction with regard to the 1^{st} experiment.

Table 4: Average tagging accuracy (%) of Stagger, for different experiments, when tagging the IFD corpus using 10-fold cross-validation. Error reduction is shown for all words. State-of-the-art results are shown in bold font. Average UWR in testing is 6.8% without using data from BÍN, compared to 1.0% when using BÍN. LF=Linguistic features; WE=Word embeddings. Differences in all-word accuracy are significant at $p < 0.001$, using McNemar's test.

Previous research on tagging Icelandic has revealed that many tagging errors are due to case confusion for nominals (Helgadóttir, 2005; Dredze and Wallenberg, 2008a; Loftsson et al., 2009). In order to more accurately select the case of nominals, Dredze and Wallenberg (2008a), in their Bidir tagger (see Section 3.2), use a separate case tagger which is essentially a second-pass PoS tagger that is only permitted to change the case and gender selections of the first-pass tagger, whose output the case tagger has access to.

We first implemented a separate case tagger following Dredze and Wallenberg (2008a), although using only their *Feature Group 1* (which they reported to be by far the most useful one), and could confirm their finding that the overall tagging accuracy is improved by the case tagger.

However, since their Feature Group 1 only uses previously assigned tags to the left of the current tag, and we use a left-to-right beam search, we were able to add the case tagger features to the PoS tagger and perform the tagging in only one pass. This led to a small improvement in accuracy over the separate case tagger, but, perhaps more importantly, eliminated much of the overhead in performance and code complexity caused by re-tagging all sentences with the case tagger.

The result of this experiment no. 2, with Stagger+LF, is shown in Table 4. We obtain an accuracy of 91.42% for all words, which corresponds to a 1.5% error reduction compared to the basic feature set. Dredze and Wallenberg (2008a) obtained an error reduction of 4.6% when adding their case tagger to their base tagger. It is not clear to us why we do not achieve similar gains, even though we use very similar features.

4.3 Handling unknown words

As mentioned in Section 4.1, Stagger's accuracy for unknown words (66.06% in experiment no. 2) is significantly lower than the corresponding accuracy of the three taggers shown in Table 3 (e.g. 77.47% by the HMM+Ice+HMM tagger). The competition in this category is indeed tough. First, TriTagger uses an effective suffix algorithm, based on probability distributions for suffixes of various lengths (Brants, 2000), which has worked well for various languages. Second, both IceTagger and HMM+Ice+HMM use IceMorphy for guessing the tag profile for unknown words.

Nevertheless, Stagger's accuracy for unknown words is higher than the corresponding accuracy in various other data-driven taggers tested by Helgadóttir (2005).

The next logical step was thus trying to increase the accuracy for unknown words and minimizing the UWR. We carried this out in two ways. First, in Section 4.3.1, by integrating IceMorphy with Stagger, and, second, in Section 4.3.2, by providing Stagger with data from BÍN.

Note that unknown word guessers (like IceMorphy) exist for various languages (Mikheev, 1997; Nakov et al., 2003; Nakagawa and Yuji, 2006), whereas large comprehensive inflectional databases like BÍN are rare.

4.3.1 Stagger with IceMorphy

As mentioned in Section 3.2, IceMorphy is an unknown word guesser and a "filler" for tag profile gaps for known words. Since both IceMorphy and Stagger are open-source, and implemented in Java, we could easily integrate the two components.

We did so, in the following manner. During testing, we look up a word w in the dictionary derived by Stagger during training. If w is known, then the tag gap filling method of IceMorphy is used to guess missing tags in the profile for w.[6] If w is unknown, then IceMorphy is used to guess the whole tag profile for w. The dictionaries used by IceMorphy are generated from the IFD corpus (Loftsson, 2008). When we evaluate each test fold F, using the integration of Stagger and IceMorphy, we thus only make IceMorphy have access to the dictionaries generated from fold F.

The results for this tagger, Stagger+LF+IceMorphy (generic Stagger, in addition to linguistic features and IceMorphy), are shown as experiment no. 3 in Table 4. We obtain a large increase in tagging accuracy for all words, from 91.42% to 92.82%, which is equivalent to an error reduction of 16.3%. By integrating IceMorphy with Stagger, accuracy for unknown words increases from 66.06% to 77.03% (error reduction of 32.3%), and for known words from 93.26% to 93.97% (error reduction of 10.5%).

4.3.2 Adding data from BÍN

As discussed in Section 3.1, BÍN is a large morphological database. It has been used by Loftsson et al. (2011) to extend the dictionaries of PoS taggers, thereby minimizing the UWR.

In our fourth experiment, we added data from BÍN to Stagger, which during training can be provided with an optional lexicon. We used exactly the same data as used by Loftsson et al. (2011), i.e. a file of about 5.3 million entries from BÍN, in which the PoS tags used in BÍN have been mapped to the IFD tags (see Loftsson et al. (2011) for details on how the file is generated). From this file, we generated the entries for the optional lexicon, i.e. a 4-tuple: <word, lemma, tag, frequency>. The first three elements of this tuple come from BÍN, whereas the last element is 0, in our case. During training, the provided lexicon is automatically extended by the training set.

Since BÍN contains a very large amount of word forms, there is no need to use IceMorphy to fill in tag profile gaps – adding the data from BÍN indeed means that most such gaps disappear. Hence, in this experiment, we tested only using IceMorphy to guess the tag profile for unknown words.

[6] We do not use tag profile gap filling for verbs, because IceMorphy overgenerates (recall is higher than precision (Loftsson, 2008)) and testing on the development set resulted in higher accuracy by excluding the verbs.

The result of this experiment no. 4 is shown as the tagger Stagger+LF+IceMorphy+BÍN in Table 4. For all words, we obtain a state-of-the-art accuracy of 93.70%, which is equivalent to 3.4% error reduction compared to the previously best result of 93.48%, obtained by the HMM+Ice+HMM tagger (see Table 3). Stagger's accuracy for known words (94.02%) is somewhat higher than the corresponding accuracy of the HMM+Ice+HMM tagger (93.85%), and for unknown words Stagger's accuracy (61.45%) also surpasses that of HMM+Ice+HMM (60.50%).

4.4 Adding word embeddings

In our final experiment, no. 5, we added C&W word embeddings (WE), discussed in Section 2.4, as features during training. We induced 48-dimensional C&W embeddings using the tool of Robert Östling[7] and 10 million sentences (ca. 170 million words) of Icelandic web texts from the Wortschatz project.[8] We limited the vocabulary to words occurring at least 100 times in the data, in total 65 426 words. The embeddings were trained for 25 billion updates, or about 150 iterations through the entire corpus, which took about ten days on a 2.66 GHz Intel Core2 system.

As shown in Table 4, we obtain an overall accuracy of 93.84% for the Stagger+LF+IceMorphy+BÍN+WE tagger, which is equivalent to 29.3% error reduction compared to the first experiment (see Table 4), and 5.5% error reduction compared to the previous best results.

5 Error analysis

We performed preliminary error analysis on the output of the Stagger+LF+IceMorphy+BÍN+WE tagger. We combined the tagging errors made on the test sets of the first nine folds.

We define an error type as a pair (x, y), where x is the predicted tag and y is the gold tag. Stagger makes 4,108 different error types. 1,796 of those, or 43.72%, appear only once. The 10 most frequent errors account for 14.60% of the total errors, as shown in Table 5. It is notable that 6 of the 10 most frequent errors are due to mistakes in case assignments (errors no. 1-4 and 9-10 in Table 5).

The two most frequent error types (aþ,ao) and (ao,aþ) differentiate between a preposition governing the accusative (*ao*) or dative case (*aþ*). The case marking in these tags "[...] do not reflect any morphological distinctions in the words they are attached to, but only indicate the effect (case government) that these words have on their complements" (Loftsson et al., 2009).

For example, for the prepositional phrase "með bók" 'with book', tagged as "ao nveo", the fourth letter (o) in the tag for the noun "bók" denotes accusative case and the corresponding case marking on the preposition "með" is really redundant. In a way, a tagger is penalized twice for such case tagging errors, because of the wrong case in the preposition in addition to the error in the complement. If we do not make a distinction between the tags "ao" and "aþ", the tagging accuracy of the Stagger+LF+IceMorphy+BÍN+WE tagger increases from 93.84% to 94.17%.

6 Conclusion

We have applied Stagger, an open-source implementation of an Averaged Perceptron tagger, to the morphologically complex Icelandic language, obtaining state-of-the-art tagging accuracy

[7] `http://www.ling.su.se/english/nlp/tools/svek`
[8] `http://corpora.uni-leipzig.de/`

No.	Error type	Rate	Cumulative rate
1	(aþ,ao)	2.97	2.97
2	(ao,aþ)	2.35	5.32
3	(nveþ,nveo)	1.49	6.81
4	(nveo,nveþ)	1.33	8.14
5	(sng,sfg3fn)	1.24	9.37
6	(ao,aa)	1.21	10.58
7	(sfg3eþ,sfg1eþ)	1.18	11.76
8	(aa,ao)	1.01	12.77
9	(nheo,nhen)	0.93	13.70
10	(nhen,nheo)	0.90	14.60

Table 5: The 10 most frequent error types and their rate of occurrence in % in the output of the Stagger+LF+IceMorphy+BÍN+WE tagger. An error type is a pair (x, y): x is the predicted tag and y is the gold tag.

when using a version of the IFD corpus in which 565 different tags appear. Our best result, the accuracy of 93.84%, is equivalent to 5.5% error reduction compared to the previous best results on the same data.

Our largest gain in accuracy was obtained by integrating Stagger with IceMorphy, an open-source unknown word guesser for Icelandic, for the purpose of guessing the tag profile for unknown words, and to fill in gaps in the tag profile for known words. Nearly as much was gained when we used data from a large morphological database to reduce the UWR. Small increases in accuracy were obtained by adding some language-specific features, and by using word embeddings induced from a large unannotated corpus.

For a morphologically complex language, where unknown word forms are particularly frequent, we conclude that simple suffix features (as used in the generic version of Stagger) can not compare to a good morphological analyzer (unknown word guesser) or a large morphological database, and that the analyzer probably represents the best time investment when improving a PoS tagger.

In future work, we would like to experiment further with semi-supervised training, and apply the lessons learned from the Icelandic case to other morphologically complex languages. For Icelandic in particular, we want to explore ways to reduce the most frequent errors based on incorrect case assignments.

References

Berger, A. L., Pietra, V. J. D., and Pietra, S. A. D. (1996). A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22:39–71.

Bjarnadóttir, K. (2012). The Database of Modern Icelandic Inflection. In *Proceedings of the workshop "Language Technology for Normalization of Less-Resourced Languages", SaLTMiL 8 – AfLaT*, LREC, Istanbul, Turkey.

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

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PA, USA.

Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask learning. In *Proceedings of the* 25^{th} *International Conference on Machine learning*, ICML, Helsinki, Finland.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Dredze, M. and Wallenberg, J. (2008a). Further Results and Analysis of Icelandic Part of Speech Tagging. Technical report, Department of Computer and Information Science, University of Pennsylvania.

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

Georgiev, G., Zhikov, V., Simov, K., Osenova, P., and Nakov, P. (2012). Feature-Rich Part-of-speech Tagging for Morphologically Complex Languages: Application to Bulgarian. In *Proceedings of the* 13^{th} *Conference of the European Chapter of the Association for Computational Linguistics*, EACL, Avignon, France.

Giménez, J. and Màrquez, L. (2004). SVMTool: A general POS tagger generator based on Support Vector Machines. In *Proceedings of the* 4^{th} *International Conference on Language Resources and Evaluation*, LREC, Lisbon, Portugal.

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

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the* 18^{th} *International Conference on Machine Learning*, ICML, Williamstown, MA, USA.

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

Loftsson, H., Helgadóttir, S., and Rögnvaldsson, E. (2011). Using a morphological database to increase the accuracy in PoS tagging. In *Proceedings of Recent Advances in Natural Language Processing*, RANLP, Hissar, Bulgaria.

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

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

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(20):313–330.

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

Nakagawa, T. and Yuji, M. (2006). Guessing parts-of-speech of unknown words using global information. In *Proceedings of the* 21^{st} *International Conference on Computational Linguistics and the* 44^{th} *Annual meeting of the Association for Computational Linguistics*, Sydney, Australia.

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

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

Radziszewski, A. (2013). A tiered CRF tagger for Polish. In Bembenik, R., Skonieczny, Ł., Rybiński, H., Kryszkiewicz, M., and Niezgódka, M., editors, *Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions*. Springer Verlag.

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

Rögnvaldsson, E. and Helgadóttir, S. (2011). Morphosyntactic Tagging of Old Icelandic Texts and Its Use in Studying Syntactic Variation and Change. In Sporleder, C., van den Bosch, A., and Zervanou, K., editors, *Language Technology for Cultural Heritage: Selected Papers from the LaTeCH Workshop Series*. Springer, Berlin.

Shen, L., Satta, G., and Joshi, A. (2007). Guided Learning for Bidirectional Sequence Classification. In *Proceedings of the* 45^{th} *Annual Meeting of the Association of Computational Linguistics*, ACL, Prague, Czech Republic.

Søgaard, A. (2011). Semi-supervised condensed nearest neighbor for part-of-speech tagging. In *Proceedings of the* 49^{th} *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL-HLT, Portland, Oregon.

Spoustová, D. j., Hajič, J., Raab, J., and Spousta, M. (2009). Semi-supervised Training for the Averaged Perceptron POS Tagger. In *Proceedings of the* 12^{th} *Conference of the European Chapter of the Association for Computational Linguistics*, EACL, Athens, Greece.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, NAACL, Edmonton, Canada.

Tsuruoka, Y., Miyao, Y., and Kazama, J. (2011). Learning with Lookahead: Can History-Based Models Rival Globally Optimized Models? In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, CoNLL, Portland, Oregon, USA.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the* 48^{th} *Annual Meeting of the Association for Computational Linguistics*, ACL, Uppsala, Sweden.

Östling, R. (2012). Stagger: A modern POS tagger for Swedish. In *Proceedings of the Swedish Language Technology Conference*, SLTC, Lund, Sweden.