Northern European Journal of Language Technology, 2013, Vol. 3, Article 1, pp 1–18
DOI 10.3384/nejlt.2000-1533.1331

Stagger: an Open-Source Part of Speech Tagger for Swedish

Robert Östling

Stockholm University
Department of Linguistics
robert@ling.su.se

Abstract

This work presents Stagger, a new open-source part of speech tagger for Swedish based on the Averaged Perceptron. By using the SALDO morphological lexicon and semi-supervised learning in the form of Collobert and Weston embeddings, it reaches an accuracy of 96.4% on the standard Stockholm-Umeå Corpus dataset, making it the best single part of speech tagging system reported for Swedish. Accuracy increases to 96.6% on the latest version of the corpus, where the annotation has been revised to increase consistency. Stagger is also evaluated on a new corpus of Swedish blog posts, investigating its out-of-domain performance.

1 Introduction

Stagger is a part of speech (PoS) tagger for Swedish, based on the Averaged Perceptron (Collins, 2002). It also performs named entity recognition (NER) and lemmatization, and contains a robust tokenizer designed for Internet text. This is however separate from the PoS tagging, and will not be further discussed here. Stagger is freely available[1] under the open-source GNU General Public License (GPL). Preliminary results on Stagger were briefly presented during the 2012 Swedish Language Technology Conference (Östling, 2012).

In this work, Stagger is shown to be the most accurate single tagger for Swedish, improving upon the previously best tagger (Carlberger and Kann, 1999), as evaluated by Sjöbergh (2003a), by a 10% error reduction. Further improvement is made when using a newer version of the Stockholm-Umeå Corpus (SUC), where the annotation has been revised to be more consistent. Work carried out in parallel with this (Loftsson and Östling, 2013) has also shown a modified version of Stagger to be the most accurate PoS tagger for Icelandic.

With increasing amounts of text available through the Internet, text that may or may not adhere to traditional written language norms, PoS tagging of this so-called *user-generated content* has become an important area of research. Stagger's performance on

[1] http://www.ling.su.se/stagger

System		Accuracy (%)
Granska	(Carlberger and Kann, 1999)	96.0
TnT	(Brants, 2000)	95.9
fnTBL	(Ngai and Florian, 2001)	95.6
MXPOST	(Ratnaparkhi, 1996)	95.5
TreeTagger	(Schmid, 1994)	95.1
TiMBL	(Daelemans et al., 2001)	94.7
Stomp	(Sjöbergh, 2003b)	93.8

Table 1: Tagging accuracy for 7 systems using 10-fold cross-validation on SUC 2. Reproduced from Sjöbergh (2003a, Table 1).

this domain is evaluated using a new, manually annotated corpus of Swedish blog texts: the Stockholm Internet Corpus (SIC).

This article first reviews relevant previous work on PoS tagging, particularly for Swedish (Section 2), followed by a presentation of the methods used in Stagger (Section 3), then I present the data used in this study (Section 4), the experiments and their results (Section 5) and finally conclusions and directions for future research (Section 6).

2 Related Work

In this section, previous work on Swedish PoS tagging will be discussed, as well as the feature-rich stochastic models that recent state-of-the-art PoS taggers for well-resourced languages are based on.

2.1 Swedish PoS Tagging

Swedish is a Germanic language with moderately complex inflectional morphology. The de-facto standard tagset of Swedish from the Stockholm-Umeå Corpus (SUC) version 2.0 contains three delimiter types and 22 parts of speech, which combined with morphological features makes a total of 153 different tags (Källgren, 2006). This corpus is further described in Section 4.1.

Megyesi (2001) and Sjöbergh (2003a) evaluate and compare a number of data-driven PoS taggers for Swedish. Most of the systems evaluated are not adapted for Swedish, but use generic and fairly language-independent models that have previously been developed for and tested on languages other than Swedish. The primary exception is the Granska Tagger (Carlberger and Kann, 1999), which was designed particularly for Swedish and includes a large lexicon as well as a compound analyzer. In part because of this, Granska Tagger comes out as the most accurate of seven PoS taggers in a 10-fold cross validation evaluation on the SUC corpus (Sjöbergh, 2003a). The full results of this evaluation are shown in Table 1. Note that the specifics of the evaluation are not given. Section 5.1 further discusses and compares Sjöbergh's evaluation with the one used in this work.

The top two taggers are both based on Hidden Markov Models (HMM), followed by a transformation-based tagger (Brill, 1995). All three use relatively simple models, where context is modeled primarily with tag bigrams and trigrams.

The Maximum Entropy-based tagger of Ratnaparkhi (1996) and the memory-based of Daelemans et al. (2001) were the only feature-rich models evaluated, but neither excelled. As far as I am aware, before the present work, there have been no other experiments published with feature-rich models used for Swedish PoS tagging.

2.2 Feature-Rich PoS Tagging

Ratnaparkhi (1996) used a Maximum Entropy model (Berger et al., 1996) to integrate a large number of context features into a probabilistic model for PoS tagging. Since then, other models using similar feature modeling, such as Support Vector Machines (SVM) (Giménez and Màrquez, 2003), Conditional Random Fields (CRF) (Lafferty et al., 2001) and the Averaged Perceptron (Collins, 2002), have been used for PoS tagging and other sequence labeling tasks.

Section 3.2 describes how features are modeled in the Averaged Perceptron-based Stagger, which is fairly representative of the models just mentioned.

In recent years, several different machine learning frameworks have been used to perform PoS tagging with similar levels of accuracy when evaluated on the English Wall Street Journal dataset as used by Collins (2002), which has become a de-facto standard for evaluating and comparing PoS tagging algorithms. Toutanova et al. (2003) use a Maximum Entropy framework and Shen et al. (2007) an Averaged Perceptron-based one, in both cases moving beyond simple left-to-right search to achieve an accuracy of about 97.3% on the standard evaluation setup of Collins (2002). While recent CRF-based systems have obtained good results with reasonable training times (Lavergne et al., 2010), the Averaged Perceptron algorithm was chosen for its greater simplicity.

2.3 Semi-Supervised Learning

Data that is PoS-annotated by humans is scarce even for major languages in the world, with at most a few million words available, and considerably less for most languages. However, unannotated digital text is often available in much larger quantities, measured in billions or trillions of words. Combining the information contained in both types of data, so-called *semi-supervised learning*, has been shown to be a promising way of improving the accuracy of PoS tagging and other tasks. Several different approaches have been used for semi-supervised PoS tagging, and some prominent examples will be reviewed here.

Expectation-Maximization (EM) Elworthy (1994) explores how the Baum-Welch algorithm (Baum, 1972), an instance of the more general Expectation-Maximization (EM) method, can be used to combine unlabeled and annotated data in a HMM PoS tagger. This early attempt was however unable to improve upon the fully supervised method in most circumstances.

Suzuki and Isozaki (2008) use a CRF-based algorithm with Expectation-Maximization (EM) on unlabeled text, reaching 97.40% accuracy using the same evaluation setup as Collins (2002).

Word Representations Turian et al. (2010) explore how different types of *word representations* (vectors or clusters) induced from unlabeled text can be added as features in feature-rich supervised NLP tasks. They used neural language models (Mnih and Hinton, 2007; Collobert and Weston, 2008) and Brown et al. (1992) clusters.

Huang and Yates (2009) use word clusters induced from unlabeled text using the Baum-Welch algorithm and using Latent Semantic Analysis (LSA) to improve a CRF-based PoS tagger.

Graph-Based Methods Subramanya et al. (2010) construct a graph of similar n-grams using both labeled and unlabeled data, and then use information from this to perform semi-supervised training of a CRF model.

Bootstrapping Spoustová et al. (2009) use an ensemble of PoS taggers to tag the unlabeled data, which is then used to train an Averaged Perceptron-based tagger, i.e. a *bootstrapping* method. Søgaard (2011) achieves the highest reported accuracy on the Wall Street Journal data as used by Collins (2002), 97.50%, using a fairly complex scheme including both bootstrapping and word representations.

3 Methods

3.1 Averaged Perceptron [2]

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* (primarily binary) of the form $\phi(h_i, t_i)$ for a history h_i and a tag t_i at position i in the sequence, 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 $(t_{i-1}, t_{i-2}, \ldots)$ 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 context representation h_i.

An Averaged Perceptron model consists of a set of feature functions ϕ_s, each paired with a *feature weight* α_s which is to be estimated during training. A scoring function is defined over entire sequences, which in a PoS tagging task typically means sentences. For a sequence of length n in a model with d feature functions, the scoring function for a sentence w (w_1, w_2, $\ldots w_n$) is defined as:

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

[2] Parts 3.1–3.3 of this section are largely taken from 2.1–2.3 of Loftsson and Östling (2013)

Note that unlike e.g. Maximum Entropy models (Berger et al., 1996), the scoring function is not normalized to form a probability distribution over tags t. The highest scoring sequence of tags:

$$\tilde{t} = \arg\max_t score_w(t)$$

can be computed or approximated, for example, using the Viterbi algorithm or (as in the present work) a beam search.

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 tag sequence t and the model's predicted tag sequence $\tilde{t}$.

Algorithm 1 Averaged Perceptron training iteration.

```
for all t, w ∈ T do
    t̃ ← arg max_{t'} score_w(t')
    for i ← 1..n do
        for s ← 1..d do
            α_s ← α_s + φ_s(h_i, t_i) − φ_s(h_i, t̃_i)
        end for
    end for
end for
```

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 Stagger, weights are averaged after every 4096 training instances (sentences), which seems to strike a good balance between performance and accuracy.

3.2 Features

Stagger uses a basic set of binary features similar to that of Ratnaparkhi (1996). Table 2 shows the templates on which the basic features used in Stagger are based. One instance of the first template may be:

$$\phi_s(t_i, h_i) = \begin{cases} 1 & \text{if } t_i = \text{NNS} \wedge w_i = \text{cats} \wedge i \neq n \\ 0 & \text{otherwise} \end{cases}$$

or in other words, the value is 1 if the current tag is "NNS" (plural noun), the current token is "cats", and it is not the last token in the sentence. Otherwise the value is 0.

The features can be divided into two categories: history-dependent features that use the values of previously assigned tags $(t_{i-1}, t_{i-2}, \ldots)$ in the sequence, and history-independent features that do not. Making this distinction can lead to a large increase in performance, 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.

History-independent features
$t_i = x, w_i = y, i = n$
$t_i = x, w_i = y, i = 1, c(i)$
$t_i = x, w_{i-1} = y, w_i = z$
$t_i = x, w_i = y, w_{i+1} = z$
$t_i = x, w_{i-1} = y, w_i = z, w_{i+1} = u$
$t_i = x, w_{i-2} = y, w_{i-1} = z, w_i = u$
$t_i = x, w_i = y, w_{i+1} = z, w_{i+2} = u$
$t_i = x, w_{i+\{-2,-1,1,2\}} = y$
$t_i = x, \mathit{prefix}_{\{1,2,3,4\}}(w_i) = y, i = 1, c(i)$
$t_i = x, \mathit{suffix}_{\{1,2,3,4,5\}}(w_i) = y, i = 1, c(i)$
$t_i = x, k(i), \text{“-”} \in w_i$
$t_i = x, k(i), k(i+1)$
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 2: 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). x, y, z, u are constants, which in any given feature function has a fixed value.

In addition to the basic feature set detailed above, Stagger can also use Collobert and Weston (2008) embeddings through feature functions of the following form:

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

where $E_{w_i,j}$ is the j:th dimension of word w_i's Collobert and Weston embedding. The embeddings are described in Section 3.4.

3.3 PoS filter

Ratnaparkhi (1996) observed that both accuracy and speed can be improved by using a *tag filter*, rather than considering *all* tags for every word, only those known to be possible for a given word are considered. Stagger uses the following method for determining which tags to consider for a given word: If the word w occurs in the training data (it is a *known word*), only the tags found during training or in the lexicon entry for w are considered. Other words (*unknown words*) are limited to a manually specified list of 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, or only occurring in the lexicon, may be assigned tags from the union of the set of tags they occur with in the training data, the lexicon, and the set of open word class tags.

3.4 Collobert and Weston embeddings

Human-annotated text corpora typically do not reach much more than a million words in size, while unlabeled text corpora can reach over a trillion. *Semi-supervised* data-driven algorithms that make use of both human-annotated and unlabeled text data have been successfully applied to many tasks in Natural Language Processing (NLP), see Section 2.3 for further information on the subject.

While some semi-supervised algorithms use unlabeled data in a highly task-specific manner, Turian et al. (2010) explored how different *word representations* could successfully be used in a very general way to turn a supervised algorithm semi-supervised. The aim of word representations are to model the semantic and syntactic properties of words, so that semantically and syntactically similar words tend to have similar representations.

One way of representing words is to use what is variously referred to as *word vectors* or *word embeddings*, where real-valued vectors, typically of some tens or hundreds of dimensions, are used to to represent (or *embed*) words.

Bengio et al. (2003) construct a neural language model using word embeddings, where the word embeddings and the next word predictor were trained jointly. Although this model performs well, it requires computing a normalized probability distribution over the entire vocabulary at every training instance, which is infeasible for large corpora and vocabularies.

Collobert and Weston (2008) use a similar model, but with the primary aim of computing good word representations rather than creating a practical language model. Instead of computing a probability distribution over the entire vocabulary, their model only computes a single value representing how well a focus word "fits" into a context window. Actual text windows from a corpus provide positive examples, and the same text windows with the focus word corrupted (randomly replaced) serve as negative examples. In this way, the computational complexity of each training instance is independent of the vocabulary size, and it becomes possible to train the model on a large corpus, with a large vocabulary, to obtain high-quality word embeddings (Bengio et al., 2009; Collobert et al., 2011).

A Collobert and Weston language model consists of two parts, with the first part being the word embeddings. Each word is represented by a vector in $\mathbb{R}^d$. A vocabulary of (fixed) size v is used, so the embeddings can be represented by a $v \times d$ embedding matrix E. To obtain the network inputs from an n-gram $(w_1, w_2, \ldots, w_n)$, we concatenate the corresponding rows from the matrix E to obtain an nd-dimensional vector $(E_{w_1,*}, E_{w_2,*}, \ldots, E_{w_n,*})$.

The second part of the model consists of a neural network with nd inputs for an n-gram where each word is represented by its d-dimensional embedding, a h-neuron non-linear hidden layer, and a single linear output neuron.

Intuitively speaking, the network's output function $s(x)$ is trained to compute the

degree to which the focus word[3] f (a constant $1 \leq f \leq n$) "fits" into the context of the rest of the n-gram. Positive examples are n-grams $x = (w_1, w_2, \ldots, w_f, \ldots, w_n)$ from a text corpus, and for each x a corrupted n-gram $\tilde{x} = (w_1, w_2, \ldots, w, \ldots, w_n)$ is produced from x by replacing w_f by another word w, selected uniformly from the vocabulary.

The model is trained by repeatedly iterating over n-grams x from a corpus, for each x selecting one corruption $\tilde{x}$, and backpropagating the value of the error function $\max(0, 1 - s(x) + s(\tilde{x}))$ to update the network weights and the embedding matrix. Thus, for an n-gram x, we strive to make the value of $s(x) - s(\tilde{x})$ at least 1, for any of the possible corruptions $\tilde{x}$ of x.

Turian et al. (2010) showed that adding Collobert and Weston embeddings as features in a normal supervised setting can improve accuracy in named entity recognition (NER) and shallow parsing.

4 Data

4.1 Stockholm-Umeå Corpus

The Stockholm-Umeå Corpus (SUC) is a balanced corpus of written Swedish, compiled in the 1990s at Stockholm University and the University of Umeå (Ejerhed et al., 1992), with the updated version 2 released in 2006 (Källgren, 2006) and version 3 in 2012. The last version has previously only been presented in passing (Östling, 2012), so this section will serve as a brief introduction to SUC 3 in particular.

SUC contains about 1 million words of material where each token is annotated with lemma, PoS and named entity (NE) information. Versions 2 and 3 use a set of 153 PoS tags (of which 3 are delimiters) and 9 NE labels. It has long been recognized that annotation inconsistency is an issue in SUC (Källgren, 1996), and in the decades since the project first begun, different groups and individuals have contributed to improving the quality of the annotation: Britt Hartmann, Kenneth Wilhelmsson, the Swedish Treebank project at Uppsala University, and the present author. In total, 2 952 PoS tags have been changed between versions 2 and 3, or 0.25% of all tokens. 1 514 lemmas have been changed, in many cases due to PoS tag changes.

Although previous studies have shown that using modified versions of the SUC 2 tagset can lead to more accurate PoS tagging (Forsbom, 2008; Carlberger and Kann, 1999), we have decided to keep the SUC 2 tagset, since this has become a de-facto standard for Swedish PoS tagging.

Forsbom and Wilhelmsson (2010) previously found that a subset of the annotation changes now used in SUC 3 led to an improvement in accuracy for a data-driven PoS tagger, compared to version 2 of SUC. In Section 5.3, this experiment is repeated with Stagger and the full set of changes in SUC 3.

SUC consists of 500 text excerpts of about 2000 words each, and SUC 3 contains a division of these into a default training-development-test split to facilitate reproducible evaluations. The test set (10 files, 2% of the total corpus) was chosen to be identical to

[3]Turian et al. (2010) and Bengio et al. (2009) use $f = n$, that is, they replace the last word in each n-gram. Collobert and Weston (2008) use $f = \lfloor n/2 \rfloor$, that is, they replace the center word. Here I follow Collobert and Weston (2008) in replacing the center word, since there seems to be no good reason to discard the context after a word.

that of the Swedish Treebank project.[4] The development set (10 files, 2%) was chosen randomly among the remaining files,[5] and the remaining part (480 files, 96%) is used for training.

4.2 SALDO

SALDO (Borin and Forsberg, 2009) is a digital lexicon of words and multi-word expressions in modern Swedish, consisting of a semantic hierarchy as well as morphological descriptions. In total, there are about 115 000 lemmas and 1 800 000 inflectional forms. SALDO is thus comparable in size to the lexicon used by Carlberger and Kann (1999), and has the added advantage of being available under a permissive license allowing free usage, adaptation and re-distribution.

Although not using the SUC tagset for its morphological descriptions, SALDO is still largely compatible with it, and one can easily use the two resources together. In Stagger, the possible tags of open-class words (nouns, verbs excluding participles, adjectives and adverbs) from SALDO are added to the tag filter.

4.3 Stockholm Internet Corpus

The Stockholm Internet Corpus (SIC) consists of about 8 000 tokens of text from Swedish blogs, annotated by the present author with PoS tags and named entities using the SUC tagset, with the addition of one extra PoS tag for emoticons.[6] The corpus currently contains 16 posts by 3 different authors. We hope to extend this in the future, but due to limited resources it has not yet been possible to reach a comfortable amount of material.

Although its size is currently very limited, SIC is sufficiently large to be used in a basic experiment on out-of-domain tagging accuracy, presented in Section 5.3.

5 Experiments

In order to evaluate the accuracy of Stagger, and understand when it fails, a series of experiments were carried out. Unless explicitly stated, all accuracy figures reported refer to complete tags, including part of speech as well as morphological information.

All the experiments described in this section use a beam size of 8, which was empirically found to offer a good speed-accuracy tradeoff. With this setting, training a model using the SUC corpus takes about one hour[7] and the tagging speed is about 5000 tokens/second. Since only sentence-local context is used in the tagging process, the task is trivially parallelizable and could easily be sped up on multiprocessor systems.

[4] Files: `aa05, aa09, ba07, ea10, ea12, ja06, kk14, kk44, kl07, kn08`

[5] Files: `aa11, ae06, bb07, ec16, fh10, ha02, hf01, jg02, kk21, kl06`

[6] Freely available under a Creative Commons license:
`http://www.ling.su.se/sic`

[7] All experiments were performed using a 2.4 GHz Intel Xeon E5645 CPU.

5.1 Evaluation Setup

The default training-development-test data split of SUC 3 (see Section 4.1) offers a convenient path to reproducible evaluations, but since the test set is only about 20 000 words (ca 2% of the corpus), the uncertainty is relatively large.

Therefore, in these experiments I also use 10-fold cross validation. The split for each fold is obtained in the following manner:

- Files are sorted in alphanumerical order, and each file is numbered $0 \ldots 499$ according to its rank.
- For fold f, a file with index i belongs to:
  - the test set if $i \equiv f \mod 10$
  - otherwise, the development set if $i + 1 \equiv f \mod 10 \wedge \lfloor \frac{i}{10} \rfloor \equiv 0 \mod 5$
  - otherwise, the training set

The development set is used to determine the number of iterations for the Averaged Perceptron algorithm, and may be considered part of the training data.

By choosing every 10th file for the test set, it becomes maximally balanced, as was the goal of Sjöbergh (2003a). His evaluation however uses 475 files for training in each fold (unfortunately without specifying which), whereas mine uses only 450. Our figures should therefore be comparable, but probably differ somewhat.

5.2 Collobert and Weston Embeddings

48-dimensional C&W embeddings were induced from a 2 billion word corpus of Swedish blog posts,[8] using my *Svek* software.[9] The vocabulary size was 110 760, with a 2+2 context window, and 48 hidden neurons. The embeddings were trained for 10 billion updates.

5.3 Results

Three different configurations were evaluated:

- Stagger only
- Stagger with the SALDO lexicon
- Stagger with the SALDO lexicon and C&W embeddings as features

Each configuration was evaluated on four data sets:

- SUC 2: Cross-validation on SUC 2 (see Section 5.1)
- SUC 3: Cross-validation on SUC 3 (see Section 5.1)

[8] Freely available with randomized sentence order under a Creative Commons license: `http://www.ling.su.se/sbs`

[9] Freely available under the GNU General Public License version 3: `http://www.ling.su.se/english/nlp/tools/svek/`

Configuration	SUC 2	SUC 3	Test 3	Blogs
Stagger	95.87	96.06	96.58	91.80
Stagger+SALDO	96.32	96.51	**96.96**	**92.46**
Stagger+SALDO+C&W	**96.40**	**96.58**	**97.01**	**92.49**

Table 3: PoS tagging accuracy in percent. The difference in accuracy between boldfaced figures and the normal figures in the same column are significant (at $p < 0.001$) using McNemar's test. For instance, there is no significant difference between the 92.46 and 92.49 figures in the *Blogs* column, buth both are significantly higher than the 91.80 figure above them.

Configuration	Known word accuracy	Unknown word accuracy	Total accuracy	UWR
Stagger	97.12	83.94	96.06	7.24
Stagger+SALDO	96.99	82.40	96.51	3.23

Table 4: PoS tagging accuracy and unknown word rate (UWR) in percent. Cross-validation on the SUC 3 corpus, without C&W embedding features.

- Test 3: Training-test split of SUC 3 (see Section 4.1)
- Blogs: Trained on all of SUC 3, evaluated on SIC (see Section 4.3)

The result of the evaluation is shown in Table 3. We now turn to have a closer look at how different factors affect tagging accuracy.

Annotation Accuracy Using the basic tagger, there is an error reduction of 4.3% from version 2 of SUC to the more consistently annotated version 3. In absolute terms, this is an increase in accuracy of 0.19 percentage points, comparable to the 0.25% of tokens that were changed in version 3 (see Section 4.1).

SALDO Adding the SALDO lexicon leads to a fairly large error reduction, 11.1% in the SUC 2 cross-validation and 12.1% with SUC 3. This is not surprising, since SALDO's morphology lexicon is both large and manually constructed, with over-generation of word forms being a limited problem. In the SUC cross-validations, the unknown word rate (UWR) is 7.24% without SALDO, but is reduced to 3.23% with SALDO. Table 4 shows how the accuracy for known and unknown words vary, depending on whether or not SALDO is used. Note that the unknown word accuracy is much lower than the 92.0% reported by Carlberger and Kann (1999), who had an UWR of 6.6%. This difference is surprising, and may reflect either an implementation error or an issue with the feature set used in Stagger.

Collobert and Weston Embeddings Using C&W embedding features led to a small but statistically significant error decrease in both of the cross-validations (2.1% and 1.4% for SUC versions 2 and 3, respectively). On the smaller SUC 3 test set and blog corpus, there was no significant difference.

Count	Percentage	Word
2054	4.47%	som (that/which/who)
842	1.83%	om (if/about)
622	1.36%	så (so)
615	1.34%	en (one/a/an)
429	0.94%	för (for/too/because)
376	0.82%	till (to, prep.)
373	0.81%	på (at)
343	0.75%	att (to, inf. marker)
338	0.74%	andra (other/second)
330	0.72%	ett (one/a/an)

Table 5: 10 words most frequently mis-tagged (SUC 3 cross-validation, no lexicon or C&W features).

Count	Percentage	Word
95	14.2%	å (and/to/at)
62	9.3%	de (they/them/the/it)
58	8.7%	o (and/to/at)
22	3.3%	va (what/was)
21	3.1%	så (so)
20	3.0%	för (for/too/because)
13	1.9%	B (used as a proper noun)
12	1.8%	med (with/as well)
9	1.3%	,, (double comma)
8	1.2%	som (that/which/who)

Table 6: 10 words most frequently mis-tagged (blog corpus, model trained on SUC 3 with no lexicon or C&W features).

Blog Texts As expected, accuracy decreases when tagging the SIC corpus of blog texts. The best model has an error rate of 7.51% on the blog data, more than twice as much as when evaluated on SUC (3.43%). A handful of common words are responsible for most of this increase in errors, as will be discussed further in Section 5.4.

5.4 Error Analysis

Källgren (1996) studied PoS tagging errors in an earlier version of the SUC corpus, and found that a small number of ambiguous function words are responsible for a large portion of errors. The evaluations carried out confirm this, and additionally we see that this is an even larger problem in the blog texts.

Table 5 shows the 10 words most frequently mis-tagged in the cross-validation evaluation of SUC 3. Together, these represent 13.8% of the total tagging errors. While the overall accuracy is lower when using SUC 2, the corresponding list (not shown) contains essentially the same words.

PoS	Precision	Recall	F_1-score
HS (possessive relative pronoun)	100.00%	99.38%	99.69%
PS (possessive pronoun)	99.38%	99.58%	99.48%
VB (verb)	98.82%	99.17%	98.99%
IE (infinitive marker)	98.34%	98.98%	98.66%
PP (preposition)	97.65%	99.12%	98.38%
PN (pronoun)	98.10%	98.24%	98.17%
NN (noun)	97.90%	98.21%	98.06%
DT (determiner)	97.69%	98.33%	98.01%
KN (conjunction)	97.77%	97.52%	97.65%
RG (cardinal number)	97.39%	94.82%	96.08%
SN (subordinating conjunction)	96.79%	94.61%	95.69%
HD (relative determiner)	95.94%	93.76%	94.84%
JJ (adjective)	94.94%	94.04%	94.49%
HP (relative pronoun)	92.67%	96.32%	94.46%
AB (adverb)	94.44%	94.29%	94.36%
PM (proper noun)	93.03%	92.40%	92.71%
RO (ordinal number)	94.51%	90.33%	92.37%
IN (interjection)	93.66%	87.40%	90.42%
HA (relative adverb)	94.76%	86.22%	90.29%
PC (participle)	89.55%	90.97%	90.25%
PL (verb particle)	87.17%	83.11%	85.09%
UO (foreign word)	66.99%	39.91%	50.02%

Table 7: Precision, recall and F_1-score for each major part of speech (SUC 3 cross-validation, no lexicon or C&W features).

Table 6 shows the 10 most frequently mis-tagged words in the blog corpus evaluation, and here the result is very different from that of the SUC evaluation. 22.8% of the errors are from the word variously spelled *å* or *o*, which in the standard written language is spelled *och* (and), *att* (to, inf. marker) or *å* (at, variant of the more common *på* in some expressions). The word *de* is used for two notoriously ambiguous words, *de* (the/they, 11th most frequently mis-tagged word in SUC 3) and *det* (it/that/the, 13th most frequently mis-tagged word in SUC 3). It is also sometimes used for *dem* (them).

The blog texts also contain instances of ambiguous function words, where a sense that is normally rare in written language (and thus is not readily assigned by the tagger model) is used frequently.

In total, the 10 most frequently mis-tagged words in the blog corpus are responsible for 47.8% of all tagging errors, and cause most of the degradation in tagging accuracy on this data compared to the SUC evaluation.

Table 7 shows the precision, recall and F_1-score of each major part of speech (that is, not including morphological features) in the SUC 3 evaluation. As might be expected, towards the bottom of the table we find parts of speech that are genuinely difficult to demarcate, as experience with improving the annotation quality of the SUC corpus has demonstrated.

One of the most common clusters of mis-taggings concerns adverbs, prepositions and verb particles, these together represent a total of 9.5% of all mis-tagged tokens. The three categories share many word forms, and often stress patterns (not given in writing) or context is the only way of choosing between the different possible interpretations. Adverbs are additionally often derived from neuter-gender adjectives, and confusions between these are responsible for another 4.9% of errors.

Participles are at the center of another cluster of mis-taggings, and may be confused with past-tense verbs, adjectives and nouns. Vague criteria in the corpus annotation guidelines, such as the degree of semantic connection between a verb and a participle derived from it, make it difficult to achieve a high accuracy in the annotation of participles and related categories. Forsbom (2008) shows that significant error reductions can be achieved by merging the categories of adjectives and past participles.

There are also frequent confusions of different morphological forms within the same part of speech. Some of the most common confusions are due to zero-marked plural nouns, and adjectives with identical plural and (determined) singular forms.

6 Conclusions and Future Work

I have presented Stagger, a new PoS tagger for Swedish based on the Averaged Perceptron (Collins, 2002). By using the SALDO lexicon (Borin and Forsberg, 2009) and Collobert and Weston (2008) embeddings, Stagger reaches an accuracy of 96.4% using cross-validation on the Stockholm-Umeå Corpus version 2 (Källgren, 2006), higher than any of the single taggers in the evaluation of Sjöbergh (2003a), and on par with his voting ensemble of 6 different taggers.

Stagger does not use non-local features, but this is often necessary to decide e.g. number in nouns and adjectives with identical singular/plural forms, and adding these features is one promising direction for future work. Another fact revealed by the error analysis is that some ambiguous function words stand for a considerable fraction of tagging errors, and tracking long-range syntactic dependencies could help in disambiguating these.

Loftsson and Östling (2013) show that in Icelandic, closely related to Swedish but with a more complex morphology, it is beneficial to add a morphological analyzer to handle words not in the training data or lexicon. This is particularly important when the difference between known word accuracy and unknown word accuracy is large, as is the case with Stagger. Adding a morphological analyzer for Swedish would likely improve accuracy somewhat.

To handle the language used in blogs and other user-generated content, it would seem most urgent to obtain more training data in this genre, since the first such manually annotated corpus for Swedish (see Section 4.3) is still in its cradle.

References

Baum, Leonard E. 1972. An inequality and associated maximization technique in statistical estimation for probabilistic functions of a markov process. *Inequalities* 3:1–8.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3:1137–1155.

Bengio, Yoshua, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *26th Annual International Conference on Machine Learning*, ICML 2009, pages 41–48. Montreal, Canada.

Berger, Adam L., Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics* 22:39–71.

Borin, Lars and Markus Forsberg. 2009. All in the family: A comparison of SALDO and WordNet. In *Nodalida 2009 Workshop on WordNets and other Lexical Semantic Resources – between Lexical Semantics, Lexicography, Terminology and Formal Ontologies*, pages 7–12. Odense, Denmark.

Brants, Thorsten. 2000. TnT – A Statistical Part-of-Speech Tagger. In *6th Applied Natural Language Processing Conference*, pages 224–231. Seattle, WA, USA.

Brill, Eric. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics* 21:543–565.

Brown, Peter F., Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics* 18:467–479.

Carlberger, Johan and Viggo Kann. 1999. Implementing an efficient part-of-speech tagger. *Software–Practice and Experience* 29:815–832.

Collins, Michael. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Conference on Empirical Methods in Natural Language Processing*, EMNLP 2002, pages 1–8. Philadelphia, PA, USA.

Collobert, Ronan and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *25th international conference on Machine learning*, ICML 2008, pages 160–167. Helsinki, Finland.

Collobert, Ronan, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2493–2537.

Daelemans, Walter, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2001. Timbl: Tilburg memory-based learner version 4.0. reference guide. Tech. rep., ILK.

Ejerhed, E., G. Källgren, O. Wennstedt, and M. Åström. 1992. The linguistic annotation system of the stockholm-umeå project. Tech. rep., Department of Linguistics, University of Umeå.

Elworthy, David. 1994. Does Baum-Welch re-estimation help taggers? In *Fourth conference on applied natural language processing*, ANLC 1994, pages 53–58. Stuttgart, Germany.

Forsbom, Eva. 2008. Good tag hunting: Tagability of granska tags. In B. M. Joakim Nivre, Mats Dahllöf, ed., *Resourceful Language Technology: Festschrift in Honor of Anna Sågvall Hein*, pages 77–85. Acta Universitatis Upsaliensis.

Forsbom, Eva and Kenneth Wilhelmsson. 2010. Revision of part-of-speech tagging in stockholm umeå corpus 2.0. In *Swedish Language Technology Conference*, SLTC 2010.

Giménez, Jesús and Lluís Màrquez. 2003. Fast and accurate part-of-speech tagging: The svm approach revisited. In *Recent Advances in Natural Language Processing*, RANLP 2003, pages 153–163. Borovets, Bulgaria.

Huang, Fei and Alexander Yates. 2009. Distributional representations for handling sparsity in supervised sequence-labeling. In *Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*, ACL 2009, pages 495–503. Singapore.

Källgren, Gunnel. 1996. Linguistic indeterminacy as a source of errors in tagging. In *Proceedings of the 16th conference on Computational linguistics*, COLING 1996, pages 676–680. Copenhagen, Denmark.

Källgren, Gunnel. 2006. Documentation of the Stockholm Umeå Corpus. In S. Gustafson-Čapková and B. Hartmann, eds., *Manual of the Stockholm Umeå Corpus version 2.0*, pages 5–85. Department of Linguistics, Stockholm University.

Lafferty, John D., Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Eighteenth International Conference on Machine Learning*, ICML 2001, pages 282–289. San Francisco, CA, USA.

Lavergne, Thomas, Olivier Cappé, and François Yvon. 2010. Practical very large scale crfs. In *48th Annual Meeting of the Association for Computational Linguistics*, ACL 2010, pages 504–513. Uppsala, Sweden.

Loftsson, Hrafn and Robert Östling. 2013. Tagging a morphologically complex language using an averaged perceptron tagger: The case of icelandic. In *19th Nordic Conference on Computational Linguistics*, NoDaLiDa 2013, pages 105–119. Oslo, Norway.

Megyesi, Beata. 2001. Comparing data-driven learning algorithms for pos tagging of swedish. In *Conference on Empirical Methods in Natural Language Processing*, EMNLP 2001, pages 151–158. Carnegie Mellon University, Pittsburgh, PA, USA.

Mnih, Andriy and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *24th international conference on Machine learning*, ICML 2007, pages 641–648. Corvallis, OR, USA.

Ngai, G. and R. Florian. 2001. Transformation-based learning in the fast lane. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*, NAACL 2001, pages 40–47. Pittsburgh, PA, US.

Östling, Robert. 2012. Stagger: A modern POS tagger for Swedish. In *Fourth Swedish Language Technology Conference*, SLTC 2012, pages 83–84. Lund, Sweden.

Ratnaparkhi, Adwait. 1996. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*, EMNLP 1996, pages 133–142. Philadelphia, PA, USA.

Schmid, Helmut. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49. Manchester, UK.

Shen, Libin, Giorgio Satta, and Aravind Joshi. 2007. Guided learning for bidirectional sequence classification. In *45th Annual Meeting of the Association of Computational Linguistics*, ACL 2007, pages 760–767. Prague, Czech Republic.

Sjöbergh, Jonas. 2003a. Combining pos-taggers for improved accuracy on swedish text. In *14th Nordic Conference of Computational Linguistics*, NoDaLiDa 2003. Reykjavik, Iceland.

Sjöbergh, Jonas. 2003b. Stomp, a pos-tagger with a different view. In *Recent Advances in Natural Language Processing Conference*, RANLP 2003, pages 54–60.

Søgaard, Anders. 2011. Semisupervised condensed nearest neighbor for part-of-speech tagging. In *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, HLT 2011, pages 48–52. Portland, OR, USA.

Spoustová, Drahomíra, Jan Hajič, Jan Raab, and Miroslav Spousta. 2009. Semi-supervised training for the averaged perceptron pos tagger. In *12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL 2009, pages 763–771. Athens, Greece.

Subramanya, Amarnag, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2010, pages 167–176. Cambridge, MA, USA.

Suzuki, Jun and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Annual Meeting of the Association for Computational Linguistics: Human Language Technology*, ACL-HLT 2008, pages 665–673. Columbus, OH, USA.

Toutanova, Kristina, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, NAACL 2003, pages 173–180. Edmonton, Canada.

Tsuruoka, Yoshimasa, Yusuke Miyao, and Jun'ichi Kazama. 2011. Learning with lookahead: can history-based models rival globally optimized models? In *Fifteenth Conference on Computational Natural Language Learning*, CoNLL 2011, pages 238–246. Portland, OR, USA.

Turian, Joseph, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *48th Annual Meeting of the Association for Computational Linguistics*, ACL 2010, pages 384–394. Uppsala, Sweden.