Automatic Rule Induction for Unknown Word Guessing

Andrei Mikheev *
University of Edinburgh

Words unknown to the lexicon present a substantial problem to NLP *modules which rely on morpho-syntactic information such as part-of-speech taggers or syntactic parsers. In this paper we present a technique for fully automatic acquisition of rules which guess possible part-of-speech tags for unknown words using their starting and ending segments. The learning is performed from a general-purpose lexicon and word frequencies collected from a raw corpus. Three complimentary sets of word-guessing rules are statistically induced: prefix morphological rules, suffix morphological rules and ending guessing rules. Using the proposed technique unknown-word guessing rule-sets were induced and integrated into a stochastic and a rule-based tagger which then were applied to texts with unknown words.*

Introduction

Words unknown to the lexicon present a substantial problem to NLP modules which rely on information about words, such as their part-of-speech, number, gender, case, etc., as, for instance, part-of-speech (POS-) taggers. Taggers assign a single POS-tag to a word-token, provided that it is known what POS-tags this word can take on in general and the context this word was used. A POS-tag stands for a unique set of morpho-syntactic features as exemplified in Table 1 and a word can take several POS-tags which constitute an *ambiguity class* or POS-*class* for this word. Words with their POS-classes are usually kept in a lexicon. For every input word-token the tagger accesses the lexicon, determines possible POS-tags this word can take on and then chooses the most appropriate one. However, some domain-specific words or infrequently used morphological variants of general purpose words can be missing from the lexicon and thus, their POS-classes should be guessed by the system and only then sent to the disambiguation module.

The simplest approach to POS-class guessing is either to assign all possible tags to an unknown word or to assign the most probable one which is *proper singular noun* for capitalized words and *common singular noun* otherwise. The appealing feature of these approaches is their extreme simplicity. Not surprisingly, their performance is quite poor: if a word is assigned all possible tags, the search space for the disambiguation of a single POS-tag increases and makes it fragile; if every unknown word is classified as a noun, there will be no difficulties for the disambiguation but the accuracy of such guess will not be reliable enough. Although to assign capitalized unknown words as proper nouns seems as a good heuristic "...some capitalized words are proper nouns and some are not" was argued in Church (1988) where a more elaborated heuristic was proposed. Dermatas and Kokkinakis (1995) proposed a simple probabilistic approach to the unknown word guessing: the probability that an unknown word has a particular POS-tag is estimated

* HCRC, Language Technology Group, University of Edinburgh, 2 Buccleuch Place, Edinburgh EH8 9LW, Scotland, UK.

Table 1
The most frequent open-class tags from the Penn tag-set.

Tag	Meaning	Example	Tag	Meaning	Example
NN	common noun	table	VB	verb base form	take
NNS	noun plural	tables	VBD	verb past	took
NNP	proper noun	John	VBG	gerund	taking
NNPS	plural proper noun	Vikings	VBN	past participle	taken
JJ	adjective	green	VBZ	verb present, 3d person	takes
RB	adverb	naturally	VBP	verb, present, non-3d	take

from the probability distribution of words which occur only once (so-called hapax-words) in the previously seen texts[1]. Whereas such a guesser is more accurate than the naive assignments and easily trainable, the tagging performance on unknown words is reported to be only about 66% correct for English[2].

More advanced word-guessing methods use word features such as, leading and trailing word segments to determine possible tags for unknown words. Such methods can achieve better performance reaching up-to 85% of tagging accuracy on unknown words for English (Brill, 1994; Weischedel et al., 1993). The Xerox tagger (Cutting et al., 1992) comes with a set of rules which assign an unknown word a set of possible POS-tags (i.e. POS-class) on the basis of its ending segment. We call such rules *ending-guessing* rules because they rely only on ending segments in their predictions. For example, an ending-guessing rule can predict: a word is a *gerund* or an *adjective* if it ends with "ing". The ending-guessing approach was elaborated in Weischedel et al. (1993) where an unknown word was guessed by using the probability for an unknown word to be of a particular POS-tag, given its capitalization feature and its ending. Brill (1994) and (1995) describes a system of rules which uses both ending-guessing and more morphologically motivated rules. Unlike an ending-guessing rule, a morphological rule in its prediction uses information about morphologically related words already known to the lexicon. For instance, a morphologically motivated guessing rule can say: a word is an *adjective* if adding the suffix "ly" to it will result in a word. Clearly, ending-guessing rules have wider coverage than morphologically oriented ones, but their predictions can be less accurate.

The major topic in the development of word-POS guessers is the strategy which is to be used for the acquisition of the guessing rules. A rule-based tagger described in Voutilainen (1995) was equipped with a set of guessing rules which had been hand-crafted using knowledge of English morphology and intuitions. A more appealing approach is automatic acquisition of such rules from available lexical resources since it is usually less labor-intensive and less error-prone. Zhang and Kim (1990) developed a system for automated learning of morphological word-formation rules. This system divides a string into three regions and infers from training examples their correspondence to underlying morphological features. Kupiec (1992) describes a guessing component which uses a pre-specified list of suffixes (or rather endings) and then statistically learns the predictive properties of those endings from an untagged corpus. In (Brill, 1994; Brill, 1995) a transformation-based learner which learns guessing rules from a pre-tagged training

1 A similar idea for estimating lexical prior probabilities for unknown words was suggested in Baayen and Sproat (1995).
2 The best result was detected for German – 82% accuracy and the worst result for Italian – 50% accuracy.

corpus is outlined: First the unknown words are labelled as common nouns and a list of generic transformations is defined. Then the learner tries to instantiate the generic transformations with word features observed in the text. A statistical-based suffix learner is presented in Schmid (1994). From a training corpus it constructs a suffix tree where every suffix is associated with its information measure to emit a particular POS-tag. Although the learning process in these systems is fully automated and the accuracy of obtained guessing rules reaches current state-of-the-art, for estimation of their parameters they require significant amounts of specially prepared training data – a large training corpus (usually pre-tagged), training examples, etc.

In this paper we describe a novel fully-automatic technique for the induction of POS-class guessing rules for unknown words. This technique has been partially outlined in (Mikheev, 1996b; Mikheev, 1996a) and, along with the higher than previously quoted accuracy of the induced rules, it has an advantage in terms of quantity and simplicity of annotation of data for training. Unlike many other approaches which implicitly or explicitly assume that the surface manifestations of morpho-syntactic features of unknown words are different from those of general language we argue that *within the same language unknown words obey general morphological regularities*. In our approach we do not require large amounts of annotated text but employ fully automatic statistical learning using a pre-existing general-purpose lexicon mapped to a particular tag-set and word-frequency distribution collected from a raw corpus. The proposed technique is targeted to the acquisition of both morphological and ending-guessing rules which then can be applied cascadingly using the most accurate guessing rules first. The rule induction process is guided by a thorough guessing-rule evaluation methodology which employs precision, recall and coverage as evaluation metrics.

In the rest of the paper we first introduce the kinds of guessing rules to be induced and then present a semi-unsupervised[3] statistical rule induction technique using data derived from the CELEX lexical database (Burnage, 1990). Finally we evaluate the induced guessing rules by removing all the hapax-words from the lexicon and tagging the Brown Corpus (Francis and Kucera, 1982) by a stochastic and a rule-based tagger.

1. Guessing Rule Schemata

There are two kinds of word-guessing rules employed by our cascading guesser: *morphological rules* and non-morphological *ending-guessing rules*. Morphological word-guessing rules describe how one word can be guessed given that another word is known. Unlike morphological guessing rules, non-morphological rules do not require the base form of an unknown word to be listed in the lexicon. Such rules guess the POS-class for a word on the basis of its ending or leading segments alone. This is especially important when dealing with un-inflected words and domain-specific sublanguages where many highly specialized words can be encountered. In English, as in many other languages, morphological word formation is realised by affixation: prefixation and suffixation. Thus, in general, each kind of guessing rule can be further subcategorized depending on whether it is applied to the beginning or tail of an unknown word. To mirror this classification we will introduce a general schemata for guessing rules and a guessing rule will be seen as a particular instantiation of this schemata.

Definition

3 The induction technique can be considered to be semi-unsupervised since it uses the annotation stated in the lexicon. At the same time it does not require additional annotation since that annotation already exists regardless of the rule induction task.

A *guessing-rule schemata* is a structure $G =_{x:\{b,e\}}$[-S +M ?I-class →R-class] where

- x indicates whether the rule is applied to the beginning or end of a word and has two possible values, b-beginning and e-end;
- S is the affix to be segmented; it is *deleted* (-) from the beginning or end of an unknown word according to the value of x;
- M is the mutative segment (possibly empty) which should be *added* (+) to the result string after the segmentation;
- I-class is the required POS-class (set of one or more POS-tags) of the stem; the result string after the "-S" and "+M" operations should be *checked* (?) in the lexicon for having this particular POS-class; if I-class is set to be "void" no checking is required;
- R-class is the POS-class to *assign* (→) to the unknown word if all the above operations (-S +M ?I) have been successful.

For example: the rule
$_e$[-ied +y ?(VB VBP) →(JJ VBD VBN)]
says that if there is an unknown word which ends with "ied", we should strip this ending from it and append the string "y" to the remaining part. If we then find this word in the lexicon as (VB VBP) (base verb or verb of present tense non-3d form), we conclude that the unknown word is of the category (JJ VBD VBN) (adjective, past verb or participle). Thus, for instance, if the word "specified" was unknown to the lexicon, this rule first would try to segment the required ending "ied" (specified-ied=specif), then add to the result the mutative segment "y" (specif+y=specify), and, if the word "specify" was found in the lexicon as (VB VBP), the unknown word "specified" would be classified as (JJ VBD VBN).

Since the mutative segment can be an empty string, regular morphological formations can be captured as well. For instance, the rule
$_b$[-un +"" ?(VBD VBN) →(JJ)]
says that if segmenting the prefix "un" from an unknown word results in a word which is found in the lexicon as a past verb and participle (VBD VBN), we conclude that the unknown word is an adjective (JJ). This rule will, for instance, correctly classify the word "unscrewed" if the word "screwed" is listed in the lexicon as (VBD VBN).

When setting the S segment to an empty string and the M segment to an non-empty string, the schemata allows for cases when a secondary form is listed in the lexicon and the base form is not. For instance, the rule
$_e$[-"" +ed ?(VBD VBN) →(VB VBP)]
says that if adding the segment "ed" to the end of an unknown word results in a word which is found in the lexicon as a past verb and participle (VBD VBN), then the unknown word is a base or non-3d present verb (VB VBP).

The general schemata can also capture ending-guessing rules if the I-class is set to be "void". This indicates that no stem lookup is required. Naturally, the mutative segment of such rules is always set to an empty string. For example, an ending-guessing rule
$_e$[-ing +"" ?— →(JJ NN VBG)]
says that if a word ends with "ing" it can be an adjective, a noun or a gerund. Unlike a morphological rule, this rule does not check whether the substring preceding the "ing"-ending is listed in the lexicon with a particular POS-class.

The proposed guessing rule schemata is in fact quite similar to the developed by Brill (Brill, 1995) set of generic transformations for unknown word guessing. There are, however, three major differences:

- Brill's transformations do not check whether the stem belongs to a particular POS-class while the schemata proposed here does (?I-class) and imposes therefore more rigorous constraints;
- Brill's transformations do not account for irregular morphological cases like "try-tries" whereas our schemata does (+M segment);
- Brill's guessing rules produce a single most likely tag for an unknown word, whereas our guesser is intended to imitate the lexicon and produce *all* possible tags.

There are also two of Brill's transformations which our schemata do not capture: when a particular character appears in a word and when a word appears in a particular context. As for the latter, this, actually, can be attributed to the peculiarities of Brill's tagging algorithm and, in other approaches, is captured at the disambiguation phase of the tagger itself. The former feature is indirectly captured in our approach as follows. It has been noticed e.g. (Weischedel et al., 1993) that capitalized and hyphenated words have a different distribution from other words. Our morphological rules account for this difference checking the stem of the word. The ending-guessing rules, on the other hand, do not use the information about stems. Thus if the ending "s" predicts that a word can be a *plural noun* or a *3-d form of a verb*, the information that this word was capitalized can narrow the considered set of POS-tags to the *plural proper noun*. We therefore decided to collect ending guessing rules separately for capitalized words, hyphenated words and all other words. In our experiments we restricted ourselves to the production of six different guessing rule-sets which seemed most appropriate for English:

- Suffix^0 - suffix morphological rules with no mutative endings (0). Such rules account for the regular suffixation as, for instance, book+ed=booked;
- Suffix^1 - suffix morphological rules with a mutative ending in the last letter. Such rules account for many cases of the irregular suffixation as, for instance, try-y+ied=tried;
- Prefix - prefix morphological rules with no mutative segments (0). Such rules account for the regular prefixation as, for instance, un+screw=unscrew;
- Ending$^-$ - ending guessing rules for hyphenated words;
- Ending^C - ending guessing rules for capitalized words;
- Ending* - ending guessing rules for all other (non-hyphenated and non-capitalized) words.

2. Guessing Rule Induction

As mentioned already, we see features that our guessing rule schemata is intended to capture, as general language regularities rather than properties of rare or corpus-specific words only. This significantly simplifies training data requirements: we can induce guessing rules from a general purpose[4] lexicon. First, we no longer depend on the size or even

4 as opposed to a corpus-specific one

existence of an annotated training corpus. Second, we do not require any annotation to be done for the training and reuse the information stated in the lexicon which we can automatically map to a particular tag-set that a tagger is trained to. We also use the actual frequencies of word-usage, collected from a raw corpus. This allows for the discrimination between rules which are no longer productive (but have left their imprint on the basic lexicon) and rules that are productive in real-life texts. For guessing rules to capture general language regularities, the lexicon should be as general as possible (i.e. should list *all* possible POS-tags for a word) and large. The corresponding corpus should also be large enough to obtain reliable estimates of word-frequency distribution for at least 10,000-15,000 words.

Since a word can take on several different POS-tags, in the lexicon it can be represented as a [string/POS-class] record, where the POS-*class* is a set of one or more POS-tags. For instance, the entry for the word "book" which can be a noun (NN) or a verb (VB) would look like [book (NN VB)]. Thus the Nth entry of the lexicon (w_n) can be represented as $[W\ C]_n$ where W is the surface lexical form and C is its POS-class. Different lexicon entries can share the same POS-class but they can not share the same surface lexical form. In our experiments we used a lexicon derived from CELEX (Burnage, 1990), a large multi-lingual database which includes extensive lexicons of English, Dutch and German. We constructed an English lexicon of 72,136 word-forms with morphological features which we then mapped into the Penn Treebank tag-set (Marcus, Marcinkiewicz, and Santorini, 1993). The most frequent open-class tags of this tag-set are shown in Table 1. Word-frequency distribution was estimated from the Brown Corpus, which reflects multi-domain language use.

As usual, we separated the test sample from the training sample. Here we followed the suggestion that the unknown words actually are quite similar to words which occur only once (hapax words) in the corpus (Dermatas and Kokkinakis, 1995; Baayen and Sproat, 1995). We put all the hapax words from the Brown Corpus and which were found in the CELEX-derived lexicon into the test collection (test lexicon) and all other words from the CELEX-derived lexicon into the training lexicon. In the test lexicon we also included the hapax words which were not found in the CELEX-derived lexicon, assigning them the POS-tags they were found with in the Brown Corpus. Then we filtered out words shorter than four characters, non-words such as numbers or alpha-numerals which usually are handled at the tokenization phase and all closed-class[5] words which we assume will always be present in the lexicon. Thus after all these transformations we obtained a lexicon of 59,268 entries for training and the test lexicon of 17,868 entries.

Our guessing rule induction technique uses the training and test data prepared as described above and can be seen as a sampling for the best performing rule-set from a collection of automatically produced rule-sets. Here is a brief outline of its major phases:

- Rule Extraction Phase (section 2.1) – sets of word-guessing rules, (e.g. Prefix, Suffix^0, Suffix^1, Ending, etc.) are extracted from the lexicon and cleaned of redundant and infrequently used rules;

- Rule Scoring Phase (section 2.2) – each rule from the extracted rule-sets is ranked according to its accuracy, and rules which scored above a certain threshold are included in the working rule-sets;

- Rule Merging Phase (section 2.3) – rules which have not scored high enough

5 The closed class consists of a finite and well-established list of words such as prepositions, articles, wh-words, etc.

are merged together into more general rules, then re-scored, and, depending on their score, added to the working rule-sets;

- Evaluation of the Rule-Sets (sections 2.4) – working rule-sets produced with different thresholds are evaluated to obtain the best performing ones.

2.1 Rule Extraction Phase

For the extraction of the initial sets of prefix and suffix morphological guessing rules (Prefix, Suffix^0 and Suffix^1), we define the operator "$\bigtriangledown_n$" where the index n specifies the length of the mutative ending of the main word. Thus when the index n is set to 0 the result of the application of the $\bigtriangledown_0$ operator will be a morphological rule with no mutative segment. The $\bigtriangledown_1$ operator will extract the rules with the alterations in the last letter of the main word. When the $\bigtriangledown$ operator is applied to a pair of entries from the lexicon ($[W\ C]_i$ and $[W\ C]_j$), first, it segments the last (or first) n characters of the shorter word (W_j) and stores this in the M element of the rule. Then it tries to segment an affix by subtracting the shorter word (W_j) without the mutative ending from the longer word (W_i). If the subtraction results in an non-empty string and the mutative segment is not duplicated in the affix, the system creates a morphological rule with the POS-class of the shorter word (C_j) as the I-class, the POS-class of the longer word (C_i) as the R-class and the segmented affix itself in the S field. For example:

[booked (JJ VBD VBN)] $\bigtriangledown_0$ [book (NN VB)] $\rightarrow$ $_e$[-ed +"" ?(NN VB) $\rightarrow$(JJ VBD VBN)]

[advisable (JJ)] $\bigtriangledown_1$ [advise (NN VB)] $\rightarrow$ $_e$[-able +"e" ?(NN VB) $\rightarrow$(JJ)]

The $\bigtriangledown$ operator is applied to all possible lexicon-entry pairs sequentially, and, if a rule produced by such an application has already been extracted from another pair, its frequency count (f) is incremented. Thus prefix and suffix morphological rules together with their frequencies are produced. Next, we cut out the most infrequent rules which might bias further learning. To do that we eliminate all the rules with frequency f less than a certain threshold θ, which usually is set quite low: 2–4. Such filtering reduces the rule-sets more than tenfold.

To collect the ending-guessing rules we set the upper limit on the ending length equal to five characters and thus collect from the lexicon all possible word-endings of length 1, 2, 3, 4 and 5, together with the POS-classes of the words in which these endings appeared. We also set the minimum length of the remaining substring to three characters. We define the unary operator $\triangle$ which produces a set of ending guessing rules from a word in the lexicon ($[W\ C]_i$). For instance, from a lexicon entry [different (JJ)] the operator $\triangle$ will produce five ending-guessing rules:

$$\triangle\ \text{[different (JJ)]} = \begin{cases} {}_e[-t\ \ +\text{``''}\ ?-\ \rightarrow (JJ)] \\ {}_e[-nt\ \ +\text{``''}\ ?-\ \rightarrow (JJ)] \\ {}_e[-ent\ \ +\text{``''}\ ?-\ \rightarrow (JJ)] \\ {}_e[-rent\ \ +\text{``''}\ ?-\ \rightarrow (JJ)] \\ {}_e[-erent\ \ +\text{``''}\ ?-\ \rightarrow (JJ)] \end{cases}$$

The $\triangle$ operator is applied to each entry in the lexicon, and if a so produced rule has already been extracted from another entry in the lexicon, its frequency count (f) is incremented. Then the infrequent rules with $f < \theta$ are eliminated from the ending-guessing rule-set.

After applying the $\triangle$ and $\bigtriangledown$ operations to the training lexicon, we obtained rule collections of 40,000-50,000 entries. Filtering out the rules with frequency counts of 1 reduced the collections to 5,000-7000 entries.

2.2 Rule Scoring Phase

Of course, not all acquired rules are equally good at predicting word-classes: some rules are more accurate in their guesses and some rules are more frequent in their application. For every acquired rule we need to estimate whether it is an effective rule which is worth retaining in the working rule-set. For such estimation we perform a statistical experiment as follows: we take each rule from the extracted rule-sets one-by-one, take each word-type from the training lexicon and guess its POS-class using the rule if the rule is applicable to the word. For example, if a guessing-rule strips off a particular suffix and a current word from the lexicon does not have this suffix, we classify that word and the rule as incompatible and the rule as not applicable to that word. If a rule is applicable to a word, we compare the result of the guess with the information listed in the lexicon. If the guessed class is the same as the class stated in the lexicon, we count it as a hit or success, otherwise it is a failure. Then, since we are interested in the application of the rules to word-tokens in the corpus, we multiply the result of the guess by the corpus frequency of the word. If we keep the sample space for each rule separate from the others, we have a *binomial experiment*. The value of a guessing rule closely correlates with its *estimated proportion of success* ($\hat{p}$) which is the proportion of all positive outcomes (x) of the rule application to the total number of the trials (n), which are, in fact, the number of all the compatible to the rule word-tokens in the corpus:

$$\hat{p} = \frac{x\ :\ number\ of\ successful\ guesses}{n\ :\ number\ of\ the\ comatible\ to\ the\ rule\ word-tokens}$$

The $\hat{p}$ estimate is a good indicator of the rule accuracy but it frequently suffers from large *estimation error* due to insufficient training data. For example, if a rule was found to apply just once and the total number of observations was also one, its estimate $\hat{p}$ has the maximal value (1) but clearly this is not a very reliable estimate. We tackle this problem by calculating the *lower confidence limit* π_L for the rule estimate which can be seen as the minimal expected value of $\hat{p}$ for the rule if we were to draw a large number of samples. Thus with certain confidence α we can assume that if we used more training data, the rule estimate $\hat{p}$ would be not worse than the π_L. The rule estimate then will be taken at its lowest possible value which is the π_L limit itself. First we adjust the rule estimate so that we don't have zeros in positive ($\hat{p}$) or negative ($1-\hat{p}$) outcome probabilities by adding some floor values to the numerator and denominator: $\hat{p}^*_i = \frac{x_i+0.5}{n_i+1}$. The lower confidence limit π_L then is calculated as (Hayslett, 1981):

$$\pi_L = \hat{p}^* - t^{(n-1)}_{(1-\alpha)/2} * s_p = \hat{p}^* - t^{(n-1)}_{(1-\alpha)/2} * \sqrt{\frac{\hat{p}^*(1-\hat{p}^*)}{n}}$$ where

$t^{df}_{(1-\alpha)/2}$ is a coefficient of the t-distribution. It has two parameters: α - the level of confidence and *df* - the number of degrees of freedom which is one less than the sample size ($df = n-1$). $t^{df}_{(1-\alpha)/2}$ can be looked up in the tables for the t-distribution listed in every text-book on statistics. We adopted 90% confidence for which $t^{df}_{(1-0.90)/2}$=$t^{df}_{0.05}$ takes values depending on the sample size as follows:

df	1	2	3	4	5	.....	30	40	60	infinity
$t^{df}_{0.05}$	6.314	2.920	3.353	2.132	2.015	.....	1.697	1.684	1.671	1.645

Using π_L instead of $\hat{p}$ for rule scoring, favors higher estimates ($\hat{p}$) obtained over larger samples (n). Even if one rule has a high estimate value but that estimate was obtained over a small sample, another rule with a lower estimate value but obtained over a large sample might be valued higher by π_L. This rule-scoring function resembles the one used by Tzoukermann, Radev, and Gale (1995) for scoring POS-disambiguation rules for the French tagger. The main difference between the two functions is that there the t value was implicitly assumed to be 1 which corresponds to a confidence level of 68% on a very large sample.

Table 2
Top scored Prefix and Suffix^0 guessing rules.

Prefix	*I*-class	*R*-class	**Suffix**	*I*-class	*R*-class
re	JJ NN VBG	JJ NN VBG	ment	VB VBP	NN
ex	NN	NN	ing	NN VB VBP	JJ NN VBG
self-	NN	NN	ed	NN VB VBP	JJ VBD VBN
inter	JJ	JJ	s	NN VB VBP	NNS VBZ
non	JJ	JJ	ment	NN VB VBP	NN
un	RB	RB	ly	JJ NN RB	RB
dis	JJ	JJ	ness	JJ	NN
anti-	NN	JJ	ship	NN	NN
de	JJ VBD VBN	JJ VBD VBN	able	NN VB VBP	JJ
in	RB	RB	s	NN	NNS

Another important consideration for rating a word-guessing rule is that the longer the affix or ending (S) of this rule, the more confident we are that it is not a coincidental one, even on small samples. For example, if the estimate for the word-ending "o" was obtained over a sample of 5 words and the estimate for the word-ending "fulness" was also obtained over a sample of 5 words, the latter case is more representative even though the sample size is the same. Thus we need to adjust the estimation error in accordance with the length of the affix or ending. A good way to do this is to decrease it proportionally to a value which increases along with the increase of the length. A suitable solution is to use the logarithm of the affix length:

$$score_i = \hat{p}_i^* - t_{0.05}^{(n_i-1)} * \sqrt{\frac{\hat{p}_i^*(1-\hat{p}_i^*)}{n_i}} / (1 + \log(|S_i|))$$

When the length of S (the affix or ending) is 1, the estimation error is not changed since $\log(1)$ is 0. For the rules with the affix or ending length of 2 the estimation error is reduced by $1 + \log(2) = 1.3$, for the length 3 this will be $1 + \log(3) = 1.48$, etc. The longer the length, the smaller the sample which will be considered representative enough for a confident rule estimation.

Setting the threshold (θ_s) at a certain level we include in the working rule-sets only those rules whose scores are higher than the threshold. The method for finding the optimal threshold is based on empirical evaluations of the rule-sets and is described in Section 2.4. Usually the threshold is set in the range of 65-80 points and the rule-sets are reduced down to a few hundred entries. For example, when we set the threshold (θ_s) to 75 points, the obtained ending-guessing rule collection (Ending*) comprised 1876 rules, the suffix rule collection without mutation (Suffix^0) comprised 591 rules, the suffix rule collection with mutation (Suffix^1) comprised 912 entries and the prefix rule collection (Prefix) comprised 235 rules. Table 2 shows the highest rated rules from the induced Prefix and Suffix^0 rule-sets. In general it looks like the induced morphological guessing rules mostly consist of the standard rules of English morphology and also include a small proportion of rules which do not belong to the known morphology of English. For instance, the suffix rule $_e$[-et +"" ?(NN) $\rightarrow$(NN)] does not stand for any well-known morphological rule, but its prediction is as good as those of the standard English rules. The same situation can be seen with the prefix rule $_b$[-st +"" ?(NNS) $\rightarrow$(NNS)], which is quite predictive but at the same time is not a standard English morphological rule. The ending guessing rules, naturally, include some proper English suffixes but mostly they are simply highly predictive ending segments of words.

2.3 Rule Merging Phase

Rules which have scored lower than the threshold are merged together into more general rules. These new rules, if they score above the threshold, can also be included in the working rule-sets. We merge together two rules if they scored below the threshold and have the same affix (S), mutative segment (M) and the initial class (I)[6]. We define the rule merging operator $\oplus$:

$A_i \oplus A_j = A_r : [S_i, M_i, I_i, R_i \cup R_j]$ if $S_i = S_j$ & $M_i = M_j$ & $I_i = I_j$

This operator merges two rules with the same affix (S), mutative segment(M) and the initial class (I) into one rule with the resulting class being the union of the two merged resulting classes. For example,

$_e$[-s +“” ?(NN VB) →(NNS)] $\oplus$ $_e$[-s +“” ?(NN VB) →(NNS VBZ)]
= $_e$[-s +“” ?(NN VB) →(NNS VBZ)]
$_b$[-un +“” ?(VBD VBN) →(JJ)] $\oplus$ $_b$[-un +“” ?(VBD VBN) →(VBN)]
= $_b$[-un +“” ?(VBD VBN) →(JJ VBN)]

The score of the resulting rule will be higher than the scores of the individual rules since the number of positive observations increases and the number of the trials remains the same. After a successful application of the $\oplus$ operator, the resulting general rule is substituted for the two merged ones. To perform such rule-merging over a rule-set the rules which have not been included into the working rule-set are first sorted by their score and the rules with the best scores are merged first. After each successful merging the resulting rule is re-scored. This is done recursively until the score of the resulting rule does not exceed the threshold, at which point it is added to the working rule-sets. This process is applied until no merges can be done to the rules that scored poorly. In our experiment we noticed that the merging added 30-40% new rules to the working rule-sets, and therefore the final number of rules for the induced sets were: Prefix – 348, Suffix^0 – 975, Suffix^1– 1263 and Ending* – 2196.

2.4 Direct Evaluation Phase

There are two important questions which arise at the rule acquisition stage: how to choose the scoring threshold θ_s and what the performance of the rule-sets produced with different thresholds is. The task of assigning a set of POS-tags to a word is actually quite similar to the task of document categorisation where a document is assigned a set of descriptors which represent its contents. There are a number of standard parameters (Lewis, 1991) which are used for measuring performance on this kind of task. For example, suppose that a word can take on one or more POS-tags from the set of open-class POS-tags: (JJ NN NNS RB VB VBD VBG VBN VBZ). To see how well the guesser performs, we can compare the results of the guessing with the POS-tags which are known to be true for the word (i.e. listed in the lexicon). Let us take, for instance, a lexicon entry [developed (JJ VBD VBN)]. Suppose that the guesser categorized it as [developed (JJ NN RB VBD VBZ)]. We can represent this situation as follows:

Possible Tags	JJ	NN	NNS	RB	VB	VBD	VBG	VBN	VBZ
Lexicon Info.	$\vee$					$\vee$		$\vee$	
Guesser Assign	$\vee$	$\vee$		$\vee$		$\vee$			$\vee$

The performance of the guesser can be measured in:

- *recall* - the percentage of POS-tags which were assigned correctly by the guesser,

6 For ending-guessing rules this is always the case.

Table 3
Comparative performance of different guessing rule-sets.

Measure	Sample	Xerox	Ending	Suffix^0	Suffix^1	Prefix	Cascade
Recall	Training	0.958045	0.965378	0.978751	0.966475	0.973135	0.966327
	Test	0.956262	0.951916	0.973245	0.956031	0.947015	0.952491
Precision	Training	0.648983	0.760492	0.977273	0.969032	0.959782	0.82257
	Test	0.719206	0.782712	0.979964	0.96761	0.935075	0.851626
Coverage	Training	0.872842	0.946309	0.493283	0.307658	0.048635	0.950581
	Test	0.856372	0.918876	0.367574	0.26542	0.0653175	0.926553

i.e two (JJ VBD) out of three (JJ VBD VBN) or 66%. 100% recall would mean that the guesser assigned all the correct POS-tags but not necessarily only the correct ones. So, for example, if the guesser assigned all possible POS-tags to the word its recall would have been 100%.

- *precision* - the percentage of POS-tags the guesser assigned correctly (JJ VBD) over the total number of POS-tags it assigned to the word (JJ NN RB VBD VBZ), i.e 2/5 or 40%. 100% precision would mean that the guesser did not assign incorrect POS-tags, although not necessarily all the correct ones were assigned. So if the guesser assigned only (JJ) its precision would have been 100%.
- *coverage* - the proportion of words which the guesser was able to classify, but not necessarily correctly. So, for example, if we evaluated a guesser with a random hundred words from the lexicon and the guesser assigned something to 80 of them, the coverage would have been 80%.

The interpretation of these percentages is by no means straightforward as there is no straightforward way of combining these different measures into a single one. For example, these measures assume that all combinations of POS-tags will be equally hard to disambiguate for the tagger, which is not necessarily the case. Obviously, the most important measure is *recall* since we want all possible categories for a word to be guessed. *Precision* seems to be slightly less important since the disambiguator should be able to handle additional noise but obviously not in large amounts. *Coverage* is a very important measure for a rule set since, if a rule-set can guess very accurately but only for a tiny proportion of words, its value is in doubt. Thus we will try to maximize recall first, then coverage and only then precision. We will measure the aggregate by averaging over measures per word (micro-average), i.e for every single word from the test collection the precision and recall of the guesses are calculated, and then we average over these values.

To find the optimal threshold (θ_s) for the production of a guessing rule-set, we generated a number of similar rule-sets using different thresholds and evaluated them against the training lexicon and the test lexicon of unseen 17,868 hapax words. Every word from the two lexicons was guessed by a rule-set and the results were compared with the information the word had in the lexicon. For every application of a rule-set to a word we computed the precision and recall, and then using the total number of guessed words we computed the coverage. We noticed certain regularities in the behavior of the metrics in response to the change of the threshold: the *recall* improves as the threshold increases while the *coverage* drops proportionally. This is not surprising: the higher the threshold, the fewer inaccurate rules that are included in the rule-set, but at the same time the fewer words that can be handled. An interesting behavior is shown by the *precision*:

first, it grows proportionally along with the increase of the threshold, but then at high thresholds it decreases. This means that among very confident rules with very high scores there are many quite general ones. The best thresholds were obtained in the range of 70-80 points.

Table 3 displays the metrics for the best scored (by aggregate of the three metrics on the training and the test samples) rule-sets. As the base-line standard we took the ending-guessing rule-set of supplied with the Xerox tagger (Cutting et al., 1992). When we compared the Xerox ending-guesser with the induced ending-guessing rule-set (Ending*), we saw that its precision was about 6% poorer and, most importantly, it could handle 6% fewer unknown words. Finally we measured the performance of the cascading application of the induced rule-sets when the morphological guessing rules were applied before the ending guessing rules (Prefix+Suffix^0+Suffix^1+Ending^{-C*}). We detected that the cascading application of the morphological rule-sets together with the ending-guessing rules increases the overall precision of the guessing by about 8%. This made the improvement over the base-line Xerox guesser 13% in precision and 7% in coverage on the test sample.

3. Unknown Word Tagging

The direct evaluation phase gave us a basis for setting the threshold to produce the best performing rule sets. The task of unknown word guessing is, however, a subtask of the overall part-of-speech tagging process. Thus we are mostly interested in how the advantage of one rule-set over another will affect the tagging performance. Therefore, we performed an evaluation of the impact of the word guessers on tagging accuracy. In this evaluation we used the cascading guesser with two different taggers: a C++ implemented bi-gram HMM tagger akin to one described in Kupiec (1992) and the rule-based tagger of Brill (Brill, 1995). Because of the similarities in the algorithms with the LISP implemented Xerox tagger, we could directly use the Xerox guessing rule-set with the HMM tagger. Brill's tagger came pre-trained on the Brown Corpus and had a corresponding guessing component. This, actually, gave us a search-space of four basic combinations: the HMM tagger equipped with the Xerox guesser, the Brill tagger with its original guesser, the HMM tagger with our cascading (Prefix+Suffix^0+Suffix^1+Ending^{-C*}) guesser and the Brill tagger with the cascading guesser. We also tried hybrid tagging using the output of the HMM tagger as the input to Brill's final state tagger, but it gave poorer results than either of the taggers and we decided not to consider this tagging option.

3.1 Setting the Experiment

We evaluated the taggers with the guessing components on all fifteen sub-corpora of the Brown Corpus one after another. The HMM tagger was trained on the Brown Corpus in such a way that the sub-corpus used for the evaluation was not seen at the training phase. All the hapax words and capitalized words with frequency less than 20 were not seen at the training of the cascading guesser. These words were not used in the training of the tagger as well. This means that neither the HMM tagger nor the cascading guesser had been trained on the texts and words used for evaluation. We do not know whether the same holds for the Brill tagger and the Brill and Xerox guessers since we took them pre-trained. For words which failed to be guessed by the guessing components, we applied the standard method of classifying them as common nouns (NN) if they were not capitalized inside a sentence and proper nouns (NNP) otherwise. When we used the cascading guesser with the Brill tagger we interfaced them on the level of the lexicon: we guessed the unknown words *before* the tagging and added them to the lexicon listing the most likely tags[7] first as required. Here we want to clarify that we evaluated the overall results of the Brill tagger rather than just its unknown-word tagging component. Another point to mention is that, since we included the guessed words in the lexicon, the Brill tagger could use for the transformations *all relevant* POS-tags for unknown words. This is quite different from the output of the original Brill's guesser which provides only one POS-tag for an unknown word.

In our tagging experiments we measured the error rate of tagging on unknown words using different guessers. Since, arguably, the guessing of proper nouns is easier than it is for other categories, we also measured the error rate for capitalized sub-categoriy of unknown words separately. The error rate for a category of words was calculates as follows:

$Error_X = \frac{Wrongly\_Tagged\_Words\_from\_Set\_X}{Total\_Words\_in\_Set\_X}$

Thus, for instance, the error rate of tagging the unknown words is the proportion of the mistagged unknown words to all unknown words. To see the distribution of the work-load between different guessing rule-sets we also measured the coverage of a guessing rule-set:

$Coverage_R = \frac{Assigned\_Words\_by\_Rule-Set\_R}{Total\_Unknown\_Words}$

We collected the error and coverage measures for each of the fifteen sub-corpora[8] of the Brown Corpus separately, and, using the *bootstrap replicate technique* (Efron and Tibshirani, 1993), we calculated the mean and the standard error for each combination of the taggers with the guessing componenst. For the fifteen accuracy means $\{\tilde{a_1}, \tilde{a_2}, ..., \tilde{a_{15}}\}$ obtained upon tagging the fifteen sub-corpora of the Brown Corpus, we generated a large number of bootstrap replicates of the form $\{b_1, b_2, ..., b_{15}\}$ where each mean was randomly chosen with replacements such as, for instance,

$\{b_1 = \tilde{a_{11}}, b_2 = \tilde{a_4}, b_3 = \tilde{a_7}, b_4 = \tilde{a_{11}}, ..., b_{14} = \tilde{a_9}, b_{15} = \tilde{a_4}\}$.

Using these replicates we calculated the mean and standard error of the whole bootstrap distribution as follows:

$\hat{se}_B = \{\sum_{b=1}^{B}[\hat{\theta}^*(b) - \hat{\theta}^*(\cdot)]^2/(B-1)\}^{1/2}$

where

- B is the number of bootstrap replications;
- $\hat{\theta}^*(b)$ - is the mean estimate of the b-th bootstrap replication;
- $\hat{\theta}^*(\cdot) = \sum_{b=1}^{B}\hat{\theta}^*(b)/B$ - is the mean estimate of the whole bootstrap distribution;

This way of calculating the estimated standard error for the mean does not assume the normal distribution and hence provides more accurate results.

We noticed a certain inconsistency in the markup of proper nouns (NNP) in the Brown Corpus supplied with the Penn Treebank. Quite often obvious proper nouns as, for instance, “Summerdale”, “Russia” or “Rochester” were marked as common nouns (NN) and sometimes lower-cased common nouns such as “business” or “church” were marked as proper nouns. Thus we decided not to count as an error the mismatch of the NN/NNP tags. Using the HMM tagger with the lexicon containing all the words from the Brown Corpus, we obtained the error rate (mean) $\hat{\theta}^*(\cdot)$=4.003093 with the standard error $\hat{se}_B$=0.155599. This agrees with the results on the *closed* dictionary (i.e. without unknown words) obtained by other researchers for this class of the model on the same corpus (Kupiec, 1992; DeRose, 1988). The Brill tagger showed some better results:

7 which we estimated from the training data
8 Each sub-corpus belongs to a different genre ranging from news to fiction.

Table 4
Results of tagging the unknown words in the Brown Corpus.

Tagger	Guesser	Metrics	Unknown Words Er.	Unknown common words Error	Unknown common words Coverage	Unknown proper nouns Error	Unknown proper nouns Coverage
HMM	Xerox	mean	17.851643	30.022169	37.567270	10.785563	63.797113
		s-error	0.484710	0.469922	1.687396	0.613745	1.714969
HMM	Cascade	mean	12.378716	21.266264	36.507909	7.776456	64.795969
		s-error	0.917656	0.403957	2.336381	0.853958	2.206457
Brill	Brill	mean	14.688501	27.411736	38.998687	6.439525	62.160917
		s-error	0.908172	0.539634	2.627234	0.501082	4.010992
Brill	Cascade	mean	11.327863	20.986240	37.933048	5.548990	63.816586
		s-error	0.761576	0.480798	2.353510	0.561009	3.775991

error rate (mean) $\hat{\theta}^*(\cdot)$=3.327366 with the standard error $\hat{se}_B$=0.123903. Although our primary goal was not to compare the taggers themselves but rather their performance with the guessing components, we attribute the difference in their performance to the fact that Brill's tagger uses the information about the most likely tag for a *word* whereas the HMM tagger did not have this information and instead used the priors for a set of POS-tags (ambiguity class). When we removed from the lexicon all the hapax words and, following the recommendation of Church (Church, 1988), all the capitalaized words with frequency less than 20, we obtained some 51,522 unknown word-tokens (25,359 word-types) out of more than a million word-tokens in the Brown Corpus. We tagged the fifteen sub-corpora of the Brown Corpus by the four combinations of the taggers and the guessers using the lexicon of 22,260 word-types.

3.2 Results of the Experiment

Table 4 displays the tagging results on the unknown words obtained by the four different combinations of taggers and guessers. It shows the overall error rate on unknown words and also displays the distribution of the error rate and the coverage between unknown proper nouns and the other unknown words. Indeed the error rate on the proper nouns was much smaller than on the rest of the unknown words, which means that they are much easier to guess. We can also see a difference in the distribution (coverage) of the unknown words using different taggers. This can be accounted for by the fact that the unguessed capitalized words were taken by default to be proper nouns and that the Brill tagger and the HMM tagger had slightly different strategies to apply to the first word of a sentence. The cascading guesser outperformed the other two guessers in general and most importantly in the non-proper-noun category where it had an advantage of 6.5% over Brill's guesser and about 8.7% over Xerox's guesser. In our experiments the category of unknown proper nouns had a larger share (63-64%) than we expect in real life because all the capitalized words with frequency less than 20 were taken out of the lexicon. The cascading guesser also helped to improve the accuracy on unknown proper nouns by about 1% in comparison to Brill's guesser and about 3% in comparison to Xerox's guesser. The cascading guesser outperformed the other two guessers on *every* sub-corpus of the Brown Corpus. Table 5 shows the distribution of the workload and the tagging accuracy among the different rule-sets of the cascading guesser. The default assignment of the NN tag to unguessed words performed very poorly, having the error rate of 44%. When we compared this distribution to that of the Xerox guesser we saw

Table 5
Distribution of the error rate and coverage in the cascading guesser.

Metrics	Prefix		Suffix		M-Suffix		Ending		Default	
	error	cov.	error	cov.	error	cov.	error	cov.	error	cov.
mean	10.92	5.64	11.95	33.78	17.33	7.00	26.84	46.61	44.00	8.17
s-error	0.95	0.19	0.65	0.84	1.19	0.17	0.91	0.83	3.17	0.25

that the accuracy of the Xerox guesser itself was only about 6.5% lower[9] than that of the cascading guesser and the fact that it could handle 6% fewer unknown words than the cascading guesser resulted in the increase of incorrect assignments by the default strategy.

There were three types of mis-taggings on unknown words detected in our experiments. Mis-tagging of the first type occurred when a guesser provided a broader POS-class for an unknown word than a lexicon would, and the tagger had difficulties with its disambiguation. This was especially the case with the words which were guessed as noun/adjective (NN JJ) but, in fact, act only as one of them as do, for example, many hyphenated words. Another highly ambiguous group constitute the "ing" words which, in general, can act as nouns, adjectives and gerunds and only direct lexicalization can restrict the search space, as in the case with the word "going", which cannot act as an adjective. The second type of mis-tagging was caused by the incorrect assignments by the guesser. Usually this was the case with irregular words such as "cattle" or "data" which were wrongly guessed to be singular nouns (NN) but in fact were plural nouns (NNS). We also did not include the "foreign-word" category (FW) in the set of tags to guess, but this did not do too much harm because these words were very infrequent in the texts. And the third type of the mis-tagging occurred when the word-POS guesser assigned the correct POS-class to a word but the tagger still disambiguated this class incorrectly. This was the most frequent type of error, which accounted for more than 60% of the mis-taggings on unknown words.

4. Conclusion

We presented a technique for fully-automated statistical acquisition of rules which guess possible POS-tags for words unknown to the lexicon. This technique does not require specially prepared training data and uses for training a pre-existing general-purpose lexicon and word frequencies collected from a raw corpus. Using such training data, three types of guessing rules are induced: prefix morphological rules, suffix morphological rules and ending-guessing rules.

Evaluation of tagging accuracy on unknown words using texts and words unseen at the training phase showed that tagging with the automatically induced cascading guesser was consistently more accurate than previously quoted results known to the author (85%). Tagging accuracy on unknown words using the cascading guesser was 87.7-88.7%. The cascading guesser outperformed the guesser supplied with the Xerox tagger and the guesser supplied with Brill's tagger both on unknown proper nouns (which is a relatively easy-to-guess category of words) and on the rest of the unknown words where it had an advantage of 6.5.–8.5.%. When the unknown words were made known to the lexicon,

9 We attribute this to the 13% lower precision of the Xerox guesser.

the accuracy of tagging was 93.6–94.3% which makes the accuracy drop caused by the cascading guesser to be less than 6% in general.

Another important conclusion from the evaluation experiments is that the morphological guessing rules do improve the guessing performance. Since they are more accurate than ending-guessing rules they were applied first and improved the precision of the guesses by about 8%. This, actually, resulted in about 2% higher accuracy in the tagging of unknown words. The ending-guessing rules constitute the backbone of the guesser and cope with unknown words without clear morphological structure. For instance, discussing the problem of unknown words for the robust parsing Bod writes, "Notice that richer, morphological annotation would not be of any help here; the words "return", "stop" and "cost" do not have a morphological structure on the basis of which their possible lexical categories can be predicted (Bod, 1995, p.84)." When we applied the ending-guessing rules to these words, the words "return" and "stop" were correctly classified as noun/verbs (NN VB VBP) and only the word "cost" failed to be guessed by the rules.

The acquired guessing rules employed in our cascading guesser are, in fact, of a standard nature which, either in that form or another, is present in other word-POS guessers. For instance, our ending-guessing rules are akin to those of Xerox and the morphological rules resemble some rules of Brill, but ours use more constraints and provide a set of all possible tags for a word rather than a single best tag. The two additional types of features used by Brill's guesser are implicitly represented in our approach as well: One of the Brill schemata checks the context of an unknown word. In our approach we guess the words using their features only and provide several possibilities for a word; then at the disambiguation phase the context is used to choose the right tag. As for Brill's schemata which checks the presence of a particular character in an unknown word, we capture a similar feature by collecting the ending-guessing rules for proper nouns and hyphenated words separately. We believe that the technique for the induction of the ending-guessing rules is quite similar to that of Xerox[10] or Schmid (1994) but differs in the scoring and pruning methods. The major advantage of the proposed technique can be seen in the cascading application of the different sets of guessing rules and in far superior training data. We use for training a pre-existing general purpose (as opposed to corpus-tuned) lexicon. This has three advantages:

- the size of the training lexicon is large and does not depend on the size or even the existence of the annotated corpus. This allows for the induction of more rules than from a lexicon derived from an annotated corpus. For instance, the ending guesser of Xerox includes 536 rules whereas our Ending* guesser includes 2196 guessing rules;
- the information listed in a general-purpose lexicon can be considered of a better quality than that derived from an annotated corpus since it lists all possible reading for a word rather than only those which happen to occur in the corpus. We also believe that general-purpose lexicons contain less erroneous information that those derived from annotated corpora;
- the amount of work for preparing the training lexicon is minimal and does not require any additional manual annotation.

Our experiments with the lexicon derived from the CELEX lexical database and word frequencies derived from the Brown Corpus resulted in the guessing rule-sets which proved

10 Xerox's technique is not documented and can be determined only by inspection of the source code.

to be domain- and corpus-independent (but tag-set dependent), producing similar results on texts of different origins. An interesting by-product of the proposed rule-induction technique is the automatic discovery of the template morphological rules advocated in Mikheev and Liubushkina (1995). The induced morphological guessing rules turned out to consist mostly of the expected prefixes and suffixes of English and closely resemble the rules employed by the ispell UNIX spell-checker. The described rule acquisition and evaluation methods are implemented as a modular set of C++ and AWK tools, and the guesser is easily extendible to sub-language specific regularities and retrainable to new tag-sets and other languages, provided that these languages have affixational morphology.

Acknowledgments
I would like to thank the anonymous referees for helpful comments on the earlier draft of this paper.

References

Baayen, Harald and Richard Sproat. 1995. Estimating lexical priors for low-frequency morphologically ambiguous forms. *Computational Linguistics*, 22(3):155–166.

Bod, Rens. 1995. *Enriching Linguistics with Statistics: Performance Models of Natural Language*. Amsterdam: University of Amsterdam ILLC Dissertation Series 1995-14, Academishe Pers.

Brill, Eric. 1992. A simple rule-based part of speech tagger. *Proceedings of the Third Conference on Applied Natural Language Processing (ANLP-92)*, pages 152–155.

Brill, Eric. 1994. Some advances in transformation-based part of speech tagging. *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAAI-94)*.

Brill, Eric. 1995. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Burnage, G. 1990. *CELEX: A Guide for Users*. Nijmegen: Centre for Lexical Information.

Church, Kenneth W. 1988. A stochastic parts program and noun-phrase parser for unrestricted text. *Proceedings of the Second Conference on Applied Natural Language Processing (ANLP-88)*, pages 136–143.

Cutting, Doug, Julian Kupiec, Jan Pedersen, and Penelope Sibun. 1992. A practical part-of-speech tagger. *Proceedings of the Third Conference on Applied Natural Language Processing (ANLP-92)*, pages 133–140.

Dermatas, Evangelos and George Kokkinakis. 1995. Automatic stochastic tagging of natural language texts. *Computational Linguistics*, 21(2):137–164.

DeRose, Stephen. 1988. Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, 14(1):31–39.

Efron, Bradley and Robert J. Tibshirani. 1993. *An Introduction to the Bootstrap*. Brace&Co.

Francis, W. Nelson and Henry Kucera. 1982. *Frequency Analysis of English Usage:Lexicon and Grammar*. Boston: Houghton Mifflin.

Hayslett, H.T. 1981. *Frequency Analysis of English Usage:Lexicon and Grammar*. London: Heinemann.

Kupiec, Julian. 1992. Robust part-of-speech tagging using a hidden markov model. *Computer Speech and Language*, pages 225–241.

Lewis, David. 1991. Evaluating text categorization. *Speech and Natural Language: Proceedings of a Workshop held at Pacific Grove CA.*

Marcus, Mitchell, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–329.

Mikheev, Andrei. 1996a. Learning part-of-speech guessing rules from lexicon: Extension to non-concatenative operations. *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 770–775.

Mikheev, Andrei. 1996b. Unsupervised learning of word-category guessing rules. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-96)*, pages 327–334.

Mikheev, Andrei and Liubov Liubushkina. 1995. Russian morphology: An engineering approach. *Natural Language Engineering*, 1(3):235–260.

Schmid, Helmut. 1994. Part of speech tagging with neural networks. *Proceedings* *of the International Conference on Computational Linguistics (COLING-94)*, pages 172–176.

Tzoukermann, Evelin, Dragomir R. Radev, and William A. Gale. 1995. Combining linguistic knowledge and statistical learning in french part of speech tagging. *EACL SIGDAT Workshop*, pages 51–59.

Voutilainen, Atro. 1995. A syntax-based part-of-speech analyser. *Proceedings of the Seventh Conference of European Chapter of the Association for Computational Linguistics (EACL-95)*, pages 157–164.

Weischedel, Ralph, Marie Meteer, Richard Schwartz, Lance Ramshaw, and Jeff Palmucci. 1993. Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics*, 19(2):359–382.

Zhang, Byoung-Tak and Yung-Taek Kim. 1990. Morphological analysis and synthesis by automated discovery and acquisition of linguistic rules. *Proceedings of the 13th International Conference on Computational Linguistics (COLING-90)*, pages 431–435.