Natural Language Engineering **2** (2): 111–136 © 1996 Cambridge University Press

Unsupervised learning of part-of-speech guessing rules

ANDREI MIKHEEV

HCRC, Language Technology Group, University of Edinburgh,
2 Buccleuch Place, Edinburgh EH8 9LW, Scotland, UK
Email: Andrei.Mikheev@ed.ac.uk

(*Received 20 February 1996; revised 23 July 1996*)

Abstract

Words unknown to the lexicon present a substantial problem to part-of-speech tagging. In this paper we present a technique for fully unsupervised acquisition of rules which guess possible parts of speech for unknown words. This technique does not require specially prepared training data, and uses instead the lexicon supplied with a tagger 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. The acquisition process is strongly associated with guessing-rule evaluation methodology which is solely dedicated to the performance of part-of-speech guessers. Using the proposed technique a guessing-rule induction experiment was performed on the Brown Corpus data and rule-sets, with a highly competitive performance, were produced and compared with the state-of-the-art. To evaluate the impact of the word-guessing component on the overall tagging performance, it was integrated into a stochastic and a rule-based tagger and applied to texts with unknown words.

1 Introduction

Words unknown to the lexicon present a substantial problem to part-of-speech (POS) tagging of real-world texts. Apart from domain specific lexica many unknown words result from the high morphological productivity of natural language. For example, in English many nouns can be used as verbs, and from virtually any transitive verb using the suffix 'able' an adjective can be produced. It is clearly not feasible to state all possible word derivations and inflections for all the words in the lexicon. However, since for POS-tagging purposes a word must be initially associated with a set of possible POS-tags it can take on, for unknown words a guess should be made by a special component known as a word-guesser. Modern taggers can be automatically trained for new sub-languages and tag-sets. The same capability is vitally necessary for the guessing components, thus the development of accurate part-of-speech guessers and methods for their automatic training is an important task in the framework of robust analysis of unrestricted natural language texts.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

1.1 Part-of-speech tagging

The task of POS-tagging is to assign words which potentially can take on different syntactic categories in different contexts, with a single syntactic category (POS-tag) according to a particular context. For instance, the string 'books' generally can have two readings: in the phrase *'he books tickets'* the word 'books' is a verb of the third person but in the phrase *'he reads books'* it is a plural noun. Thus a POS-tagger should segment a word, determine its possible readings and assign the right one in the context. In general, the architecture of a POS tagger can be seen as a pipeline of three major components: the tokenizer, the morphological classifier and the morphological disambiguator.

The *tokenizer* is responsible, at least, for segmenting of the input text into words and sentences. More advanced tokenizers (also called preprocessors) attempt to recognize proper names, acronyms, phrasal constructions, etc., as single tokens, usually employing specialized dictionaries and finite-state sub-grammars. For example, phrasal constructions like 'as well as' or '15th Nov.95' can be recognised as single units rather than left as sequences of their subunits (words) in the text.

Text-tokens segmented by the tokenizer are essentially just strings. They are then sent to the *morphological classifier* which is responsible for classifying string-tokens as word-tokens with sets of possible morpho-syntactic features they can take on. This is usually implemented as lexicon lookup – words with their morpho-syntactic features are utilised in the lexicon and the lookup retrieves all possible readings for a given string. There are two major paradigms for organising the lexicon: the word-list lexicon where each word is declaratively stored together with its morpho-syntactic features and the morphological lexicon where the base forms of words (stems) are provided with the rules for the formation of their inflectional and derivational variants. Regardless of how the lexicon is implemented, the morphological classifier retrieves for a word its possible parts of speech and other related morpho-syntactic features such as number, case, gender, etc. In the POS-tagging framework, each combination of morpho-syntactic features is unambiguously mapped into a POS-tag. For example, a singular noun can be tagged as NN and a plural noun as NNS, the base form of a verb can be tagged as VB and its past form as VBD, etc.. Throughout this paper we will use the Brown Corpus (Francis and Kucera, 1982) tag-set whose most frequent open-class[1] tags are shown in Table 1. Naturally, one string can stand for more than one word simultaneously and thus, should be classified as several different words by the classifier. For example, the string 'books' is the surface lexical form for a plural noun and a verb of the third person, and the morphological classifier will assign two different tags (NNS and VBZ) to this string. Thus for an input string the classifier returns a set of possible POS-tags which is also called a POS-*class*.

[1] Open class is a class of words which does not consist from a finite number of words. As opposed to open classes, closed classes consist of a finite and well-established number of words as, for instance, prepositions, articles, wh-words, etc.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Table 1. *Open-class tags from the Brown Corpus tag-set.*

Tag	Meaning	Example	Tag	Meaning	Example
NN	common noun	table	VB	verb base and present	take
NN$	noun possessive	table's	VBD	verb past	took
NNS	noun plural	tables	VBG	gerund	taking
NNS$	noun plural possessive	tables'	VBN	participle	taken
NP	proper noun	John	VBZ	verb present, 3d person	takes
NP$	NP possessive	John's	JJ	adjective	green
NPS	NP plural	Johns	RB	adverb	naturally
NPS$	NP plural possessive	Johns'	QL	qualifier	awfully

Word-tokens together with their POS-tags then, are sent to the *disambiguator* which chooses a single POS-tag according to the context. There are two main approaches to POS-tag disambiguation – the connectionist (stochastic and neural net) approach and the rule-based approach. In the stochastic approach (e.g. Church, 1988; Kupiec, 1992, etc.), a word is assigned a single tag on the basis of the probability for it to be preceeded by the other tags in the sentence: $P(Tag_i|Tag_{i-1}..Tag_{i-n})$. Usually only one or two previous tags are considered: $P(Tag_i|Tag_{i-1})$ or $P(Tag_i|Tag_{i-1}Tag_{i-2})$. Since the probability that a noun will follow a pronoun is very low but the probability that a verb follows a pronoun is high, in the phrase 'he/*pronoun* books/?' the word 'books', which can be a noun or a verb, will be disambiguated as a verb rather than a noun. Similar dependences can be captured in neural nets (e.g. Schmid, 1994) as weights of the connections between the units. In the rule-based approach (e.g. Brill, 1992; Voutilainen, 1995), contextual dependences are captured more directly by a set of linguistically motivated rules. Regardless of the disambiguation strategy, at the end of the tagging process the input text-string is segmented into words each of which is assigned a single POS-tag:

He/*PPS* books/*VBZ* tickets/*NNS*./*SENT*[2]

State-of the-art taggers are reported to achieve 96–98% accuracy when all the words are found in the lexicon and associated with their POS-classes by the morphological classifier. However, 3–5% of word-tokens are usually missing from the lexicon when tagging real-world texts. This is where word-POS guessers take their place. A guesser should guess possible categorisations for the string-tokens which have not been categorised, i.e. it should provide a word with its POS-class – set of POS-tags the word can take on. In the modular architecture described above a word-guessing component, as well as the other modules can be independent of the actual tagging (disambiguation) strategy, and thus reusable with different taggers. Of course, different taggers may handle the ambiguity created by a particular word-guesser with different rates of success, and the optimal combination of a particular guesser with a particular tagger is an empirical matter.

[2] PPS – personal pronoun as subject; SENT – end of sentence.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

1.2 Existing approaches to word-guessing

The two simplest approaches to part-of-speech guessing is to assign either all possible categories to an unknown word or the most probable one[3]. The appealing feature of these approaches is the extreme simplicity of the supporting procedures for their application and porting to new tag-sets. Not surprisingly, their performance is quite poor: if a word is assigned all possible categories, the disambiguation becomes very fragile; if every unknown word is classified as a noun, the accuracy of such guess is clearly quite low. In Dermatas and Kokkinakis (1995), a simple probabilistic approach to the unknown word guessing is proposed: the probability of an unknown word to be of a particular tag is estimated from the tags' probability distribution of words which occur only once in the previously seen texts. Thus using words which occured only once in the text the proportion for different POS-tags is collected and used to predict the tags for unknown words. Whereas such a guesser is easily trainable, the tagging performance on unknown words is reported to be only about 66% correct for English[4].

More advanced word-guessing methods use the analysis of word features as, for instance, word leading and trailing characters to determine possible categories for it. Such methods can achieve sufficiently higher levels of accuracy[5]. In Church (1988), a heuristics-based method for dealing with capitalised words is described. A set of rules which on the basis of ending characters of unknown words, assign them with sets of possible POS-tags is supplied with the Xerox tagger (Kupiec, 1992). Although these rules include many proper suffixes of English – they are not morphological in the sense that they do not describe the formation of one word from another. A similar approach was taken in Weischedel *et al.* (1993), where an unknown word was guessed by using the probabilities for an unknown word to be of a particular POS, given its capitalization feature and its ending. In Brill (1994, 1995), a system of rules which uses both ending-guessing and more morphologically motivated rules is described. For example, an ending-guessing rule says: a word is a *plural common noun* if it ends with 's'. A morphologically motivated rule says: 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 intuition. A more appealing approach is an automatic acquisition of such rules from available lexical resources since it is usually attributed to be less labor-intensive and error-prone. Statistical clusterisation of words into syntactic categories, such as Finch and Chater (1993), cannot be used for this task, since first, unknown words tend to be quite rare in

[3] Usually this is *proper noun* (NP) for capitalised words and *common noun* (NN) otherwise.
[4] The best result is achieved for German – 82% accuracy and the worst result for Italian – 50% accuracy.
[5] In Brill (1994) and Weischedel *et al.* (1993), 85% of tagging accuracy on unknown words is quoted.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

comparison with other words, and second, we cannot collect collocation vectors for words which have not been previously seen. In Zhang and Kim (1990), a system for an automated learning of morphological word-formation rules is described. This system divides a string into three regions and from training examples, infers their correspondence to underlying morphological features. In Brill (1994, 1995), a transformation-based learner which learns guessing rules from a pre-tagged training corpus is outlined. First the unknown words are labelled as common nouns and the 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 unsupervised and the accuracy of obtained guessing rules reaches current state-of-the-art, they require huge amounts of specially prepared training data – a pre-tagged training corpus, training examples, etc., for the estimation of the parameters.

1.3 Novel approach – induction of the cascading guesser

In this paper we describe a novel fully automatic technique for the induction of part-of-speech guessing rules. This technique has been partially outlined in Mikheev (1996), and along with the high accuracy of the induced rules, it has an advantage in simplified, in terms of annotation and quantity, requirements to the data for training. Unlike other techniques it does not require large amounts of accurately tagged texts and employs fully unsupervised statistical learning using the lexicon supplied with a tagger and word-frequency distribution collected from a raw corpus. The proposed technique is targeted to the acquisition of both morphological and word-ending regularities for the guessing of possible parts-of-speech of the unknown words. These different kinds of rules are applied cascadingly one after another thus, constituting the *cascading unknown-word guesser*. The rule induction process is guided by a thorough guessing-rule evaluation methodology which is solely dedicated to the performance of part-of-speech guessers and as evaluation metrics employs precision, recall and coverage.

In the rest of the paper we first introduce the kinds of guessing rules to be induced, and the algorithms for their application. Then we present an unsupervised statistical rule induction technique and demonstrate its application to the data derived from the Brown Corpus (Francis and Kucera, 1982). And finally, we evaluate the induced guessing rules in the overall tagging framework using a stochastic and a rule-based tagger, taking as the base-line the word-guessing components supplied with the Xerox (Kupiec, 1992; Cutting *et al*., 1992) and the Brill (Brill, 1995) taggers.

2 Unknown word guessing rules

2.1 Kinds of guessing rules

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

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

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 $[S\ C]_n$, where S is the surface lexical form and C is its POS-class. Different lexicon entries can share the same POS-class but not the same surface lexical form.

In English, as in many other languages[6], morphological word formation is realised by affixation: prefixation and suffixation. Although sometimes affixation is not just a straightforward concatenation of the affix with the stem[7], the majority of cases usually obey simple concatenative regularities. Since it is not entirely clear how to acquire non-concatenative rules without specially prepared training examples in general[8], we will concentrate only on the pure concatenative cases. The guessing rules employed by the *cascading guesser* are closely correlated, in general, with the structure of the lexicon and have the format as follows:

[⊢S ?*I*-class →*R*-class] where

- S is the affix to be segmented; it is *deleted* from the beginning (⊢) or end (⊣) of an unknown word;
- *I*-class - *check* (?) the resulting from the segmentation string in the lexicon for having this particular POS-class;
  Precondition: the result of the segmentation is not an empty string;
- *R*-class - *assign* (→) to the unknown word this POS-class;
  Precondition: the result of the *I*-class check is 'TRUE'.

For example, if the word 'undeveloped' is not in the lexicon, the rule [⊢un ?(VBD VBN) →(JJ)] makes the guess as follows:

- undeveloped ⊢ un = developed;
- ?(developed (VBD VBN)) = TRUE;
- →(JJ),

thus the word 'undeveloped' is guessed to be of the POS-class which includes only one POS-tag – JJ.

There are two kinds of word-guessing rules employed by the cascading guesser: *morphological rules* and *ending guessing rules*. Morphological word-guessing rules describe how one word can be guessed given that another word is known. In English, as in many other languages, morphological word formation is realised by affixation: prefixation and suffixation, so there are two kinds of morphological rules: suffix rules (A^s) — rules which are applied to the tail of a word, and prefix rules (A^p)

[6] One noticeable exception is Arabic.
[7] Consider: try – tried; advise – advisable.
[8] In English the suppletion in the last letter of the stem is quite common and can be reliably detected, but in Russian, for example, the suppletion can occur in the middle of the stem (see Mikheev and Liubushkina, 1995) making its unsupervised detection rather difficult.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

— rules which are applied to the beginning of a word. For example, the prefix rule:

A^p : [⊢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 works, for instance, for words [developed — undeveloped]. An example of a suffix rule is:

A^s : [⊣ed ?(NN VB) →(JJ VBD VBN)]

This rule says that if by stripping the suffix 'ed' from an unknown word we produce a word with the POS-class noun/verb (NN VB), the unknown word is of the class adjective/past-verb/participle (JJ VBD VBN). This rule works, for instance, for word pairs [book — booked], [water — watered], etc.

Unlike morphological guessing rules, ending-guessing rules do not require the base form of an unknown word to be listed in the lexicon. These rules guess the POS-class for a word just on the basis of its ending characters. This is especially important when dealing with un-inflected words and domain specific sublanguages where a lot of highly specialized words can be encountered. The format of an ending guessing rule is similar to that of the morphological rules but the I-class is always void. This indicates that no stem lookup is necessary. For example, an ending-guessing rule

A^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 ask to check whether the substring preceeding the 'ing'-ending is a word with a particular POS-tag. Such rules are able to cover more unknown words than the morphological guessing rules but their accuracy is not expected to be as high.

2.2 Guessing rule-sets

Multiple word-guessing rules of the same kind can be grouped into a rule-set which acts as a macro-rule and applies individual rules in a certain order. A list of guessing rules of the same kind is converted into an economic and efficient representation – a packed tree format also known as a *trie* (Knuth, 1973), where all endings, suffixes and prefixes are represented as shareable chains of characters. A character in such chain can be terminal, transit or both. Terminal characters have special links to the I and R classes of the associated rules. Each rule collection is represented as a single trie where a morphological affix is associated with a set of I-class/R-class pairs while an ending is associated with just one R-class. First, each set of guessing rules is converted into the form – *affix: list of associated classes*. For example, the suffix 'ed' can be associated with several I-class/R-class pairs, thus representing several guessing rules:

[⊣ed ?(VB)→(VBD VBN), ?(NN VB)→(VBD VBN)]

Endings of the ending guessing rules are associated each with only one R-class:

[⊣ed →(VBD VBN)]

Next, the affixes are represented as shareable chains of characters. Suffixes and endings are represented in the inverted form because their analysis starts from the

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

tail of the word and prefixes are represented in the direct form. For instance, if we wanted to represent the suffix morphological rules:

[⊣less : ?(NN)→(JJ), ?(NN VB)→(JJ)]
[⊣ness : ?(JJ)→(NN), ?(JJ RB)→(NN)]
[⊣s : ?(NN)→(NNS VBZ), ?(VB)→(NNS VBZ), ?(NN VB)→(NNS VBZ)]
[⊣'s : ?(NN)→(NN$), ?(NN VB)→(NN$)]
[⊣r's : ?(NN)→(NN$), ?(NN VB)→(NN$)]

we would have had the following inverted trie structure:

Lists of *I*-*R* classes pairs	Tree of Affixes
1: ?(NN)→(NNS VBZ), ?(VB)→(NNS VBZ), ?(NN VB)→(NNS VBZ) 2: ?(NN)→(JJ), ?(NN VB)→(JJ) 3: ?(JJ)→(NN), ?(JJ RB)→(NN) 4: ?(NN)→(NN$), ?(NN VB)→(NN$)	s#1 →s →e→l#2 ↘ '#4 ↘ n#3 ↘ r#4

where #N represents a link from a terminal letter to the corresponding list of classes.

In general, longer affixes and endings are more specific than shorter ones. For instance, the ending rule [⊣ings ?- →(NNS)] is more specific than [⊣s ?- →(NNS VBZ)]. This serves as a natural ordering from more specific to more general guessing rules within the same rule-set, and the analysis starts from the longest matching affixes. Even if a longer ending does not specify a more specific guess, as for example, [⊣ity ?- →(JJ)] and [⊣ty ?- →(JJ)], it is important to retain both rules in the rule set because they enforce the right order of the application. The algorithm for the application of the ending-guessing rules is very simple: match in the trie the longest ending of the unknown word and assign to that word the POS-class associated with the matched ending. This will guarantee the most specific guess for the unknown word. For the morphological guessing rules, however, the algorithm is more complicated:

1. find in the affix trie a set of possible affixes for the word;
2. for every affix in the set starting with the longest one
   (a) strip the affix from the input word;
   (b) lookup the stripped word in the lexicon;
   (c) if the lookup failed goto 2; else
   - check if the POS-class of the retrieved word is among the *I*-classes of the affix. If 'true' then assign the word the corresponding *R*-class; STOP.
   - decompose the POS- class of the stem into the minimal number of the *I*-classes of the affix. If the decomposition is successful assign the unknown word with the union of the *R*-classes of the elements of the decomposition; STOP.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

For example, for the word 'smoker' two suffixes [⊣er ?(NN VB) →(NN)] and [⊣r ?(NN)→(NN), ?(VB)→(NN)] are retrieved from the suffix trie. First, the algorithm strips the longer suffix 'er', performs the lexicon lookup for the word 'smok' and fails. Then the suffix 'r' is stripped and the word [smoke (NN VB)] is successfully retrieved from the lexicon. The suffix 'r' does not have among its *I*-classes the (NN VB) one so the direct match fails. However, the (NN VB) class can be decomposed in terms of the two *I*-classes of the suffix 'r': NN and VB. The resulting POS-class for the word 'smoker' is then the union of the two corresponding *R*-classes: (NN) ∪ (NN) = (NN).

3 Guessing rule induction

One of the challenges in many techniques for automatic acquisition of part-of-speech guessing rules, is that they require large amounts of specially prepared training data – a pre-tagged training corpus, training examples, etc. In our approach we decided to reuse the data which come naturally with a tagger – the lexicon. Another source of information in use is the distribution of word-frequencies collected from a raw (not prepared specially for the task) text corpus. Development of such data for training is considerably simpler than the development of the training examples or a correctly tagged corpus, and the quantitative requirements are less demanding too. In our experiments we used the lexicon derived from the Brown Corpus (Francis and Kucera, 1982) and collected word-frequency distribution from the same corpus. There were a number of reasons for choosing the Brown Corpus data for training. The most important ones are that the Brown Corpus provides a model of general multi-domain language use and that there are data trained on the Brown Corpus available for many taggers. Thus throughout this paper we use the Brown Corpus tag-set whose most frequent open-class tags are shown in Table 1. This, however, by no means restricts the described technique to that or any other tag-set or to that or any other lexicon and corpus.

The induction of the guessing rules is implemented as a two-staged process with feedback. First, setting certain parameters a set of guessing rules is acquired as explained in section 3.1, then it is evaluated (see section 3.2) and the results of the evaluation are used for re-acquisition of a better tuned rule-set.

3.1 Acquisition of part-of-speech guessing rules

The word-POS guessing rule acquisition is a three-step procedure which consists of the *rule extraction*, *rule scoring* and *rule merging* phases. At the rule extraction phase (section 3.1.1), three sets of word-guessing rules (morphological prefix guessing rules, morphological suffix guessing rules and ending guessing rules) are extracted from the lexicon and then cleaned of redundant and infrequent rules. At the scoring phase (section 3.1.2), each rule is scored in accordance with its accuracy of guessing and the best scored rules are included into the final rule-sets. At the merging phase (section 3.1.3), rules which have not scored high enough to be included into the final rule-sets are merged into more general rules, then re-scored and depending on their score added to the final rule-sets.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

3.1.1 Rule extraction phase

The rule extraction technique proposed in this paper is extensively based on string comparisons, and therefore we need to introduce some operations over strings:

- $|S|$ – operator of taking the string length
  Example: |book| = 4
- $\circ$ – operator of string concatenation: $S_i \circ S_j = S_i S_j$
  Example: 'book'$\circ$'ed' = 'booked'
- $\vdash$ – operator of leftmost string subtraction: $S_i \vdash S_j = S_k \;\; if \;\; S_i = S_j \circ S_k$
  Example: 'booked'$\vdash$'book'='ed'
- $\dashv$ – operator of rightmost string subtraction: $S_i \dashv S_j = S_k \;\; if \;\; S_i = S_k \circ S_j$
  Example: 'booked'$\dashv$'ed'='book'
- $\overset{n}{\vdash}$ – operator of leftmost segmentation of the substring of the length n:
  $\overset{n}{\vdash} S_i = S_k \;\; if \;\; S_i \vdash S_k \neq \emptyset \;\; and \;\; |S_k| = n$
  Example: $\overset{4}{\vdash}$'booked'='book'
- $\overset{n}{\dashv}$ – operator of rightmost segmentation of the substring of the length n:
  $\overset{n}{\dashv} S_i = S_k \;\; if \;\; S_i \dashv S_k \neq \emptyset \;\; and \;\; |S_k| = n$
  Example: $\overset{2}{\dashv}$'booked'='ed'

For the extraction of the initial sets of prefix and suffix morphological guessing rules, we define the operator '$\bigtriangledown$' which tries to create a morphological rule from a pair of lexicon entries $[S\ C]_i$ and $[S\ C]_j$, as follows:

$$[S\ C]_i \bigtriangledown [S\ C]_j = \begin{cases} A^s = [\dashv (S_i \vdash S_j) \;\; ?C_j \;\; \rightarrow C_i\,] & if \;\; S_j \vdash S_i \neq \emptyset \\ A^p = [\vdash (S_i \dashv S_j) \;\; ?C_j \;\; \rightarrow C_i\,] & if \;\; S_j \dashv S_i \neq \emptyset \end{cases}$$

The operator '$\bigtriangledown$' tries to segment an affix by leftmost ($\vdash$ for suffixes) and rightmost ($\dashv$ for prefixes) string subtraction, and if the subtraction result is not an empty string it creates a morphological rule by storing the POS-class of the shorter word as the I-class and the POS-class of the longer word as the R-class. For example:

[interested (JJ VBD VBN)] $\bigtriangledown$ [interest (NN VB)]
$= A^s$:[$\dashv$ed ?(NN VB) $\rightarrow$(JJ VBD VBN)]
[undeveloped (JJ)] $\bigtriangledown$ [developed (VBD VBN)]
$= A^p$:[$\vdash$un ?(VBD VBN) $\rightarrow$(JJ)]

The $\bigtriangledown$ operator is applied to all possible lexicon-entry pairs and if a rule produced by such an application has already been extracted from another pair, its frequency count (f) is incremented. Thus two different sets of guessing rules — prefix and suffix morphological rules together with their frequencies — are produced. Next, from these sets of guessing rules we need to cut out infrequent rules which might bias the further learning process. To do that we eliminate all the rules with the frequency f less than a certain threshold θ[9]. Such filtering reduces the rule-sets more than tenfold and does not leave clearly coincidental cases among the rules.

[9] Usually we set this threshold quite low: 2–4.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

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 these endings were detected to appear. We define the unary operator $\triangle$ which produces a set of ending guessing rules from a word in the lexicon as follows:

$$\triangle[S\ C]_i = \begin{cases} A^e = [\dashv (\overset{1}{\dashv} S_i)\ \ ?-\ \ \rightarrow C_i] & \textit{if}\ \ |S_i| > 3 \\ A^e = [\dashv (\overset{2}{\dashv} S_i)\ \ ?-\ \ \rightarrow C_i] & \textit{if}\ \ |S_i| > 4 \\ A^e = [\dashv (\overset{3}{\dashv} S_i)\ \ ?-\ \ \rightarrow C_i] & \textit{if}\ \ |S_i| > 5 \\ A^e = [\dashv (\overset{4}{\dashv} S_i)\ \ ?-\ \ \rightarrow C_i] & \textit{if}\ \ |S_i| > 6 \\ A^e = [\dashv (\overset{5}{\dashv} S_i)\ \ ?-\ \ \rightarrow C_i] & \textit{if}\ \ |S_i| > 7 \end{cases}$$

Thus from a word [different (JJ)] the operator $\triangle$ will produce five ending-guessing rules:

$$\triangle\ [\text{different (JJ)}] = \begin{cases} A^e = [\dashv t\ \ ?-\ \ \rightarrow (JJ)] \\ A^e = [\dashv nt\ \ ?-\ \ \rightarrow (JJ)] \\ A^e = [\dashv ent\ \ ?-\ \ \rightarrow (JJ)] \\ A^e = [\dashv rent\ \ ?-\ \ \rightarrow (JJ)] \\ A^e = [\dashv erent\ \ ?-\ \ \rightarrow (JJ)] \end{cases}$$

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

3.1.2 Estimating the rule confidence

Of course, not all acquired rules are equally good as plausible guesses about word-classes: some rules are more accurate in their guessings 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 final rule-set. For such estimation we perform a statistical experiment as follows: for every rule we calculate the number of times the rule was applied to a word-token from the corpus[10] and the number of times it gave the right answer[11]. If a rule is correct for the majority of the times it had been applied it is obviously a good rule. If a rule is wrong too 0ften it is a bad rule which should not be included into the final rule-set.

To perform such an experiment we take each rule from the rule-sets one-by-one, take each word-type from the lexicon and guess its POS-class using the rule if the rule is applicable to the word. For example, if a guessing-rule strips a particular

[10] Here we don't actually use the corpus itself but rather frequencies of words in the corpus. Thus we apply the rule just once to each word-type from the lexicon and then multiply it to the frequency of this word-type in the corpus.

[11] Note that the task of the rule is not to disambiguate a word's POS but to provide *all* and *only* possible POSs it can take on.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

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 look-up in the lexicon for this word and then 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. Such an experiment, actually, can be classified as a *binomial experiment* since it conforms to the four necessary conditions (Hayslett 1981, p.64):

- it consists of a fixed number of trials equal to the number of the words from the text the rule is applicable;
- every rule trial has only two possible outcomes – 'success' and 'failure';
- the outcomes of the trials are independent from each other, i.e. the failure of a rule to classify correctly a particular word-token from the text does not increase its chances for the success on the next word-token;
- we also assume that the probability of 'success' for a rule is constant from trial to trial.

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, attempts to apply the rule to all the compatible word-tokens in the corpus:

$$\hat{p} = \frac{x}{n} .$$

The number of trials n for a rule is the cardinality of the set G of all word-tokens from the corpus T to which the rule can be applied. An application of a morphological rule $A : [S\ I\ R]$ to a word w is constrained by the requirement that the affix stripping from the surface lexical form of the word ($w : \quad [S\ \ C]$) should result in a word which belongs to the lexicon L and has the POS-class equal to that stated in the I-class of the rule. For the i*th* prefix rule this is:

$$n_i^p = |G| \quad : \quad G = \{w \in T \ \mid \ (S_w \vdash S_i) \in L \ \text{ and } \ C_{(S_w \vdash S_i)} = I_i\}$$

and for the i*th* suffix rule this is:

$$n_i^s = |G| \quad : \quad G = \{w \in T \ \mid \ (S_w \dashv S_i) \in L \ \text{ and } \ C_{(S_w \dashv S_i)} = I_i\} \qquad \text{where}$$

- T is the text corpus and w is a word from the corpus;
- L is the lexicon;
- $S_w \dashv S_i$ and $S_w \vdash S_i$ are the results of stripping off the affix (suffix or prefix) S_i from the surface lexical form of the word w;
- $C_{(S_w \dashv S_i)}$ and $C_{(S_w \vdash S_i)}$ is the POS-class of the resulting word.

Unlike morphological rules, word-ending rules do not require the word produced as the result of the ending stripping to be of a particular POS-class and the only

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

requirement for a word to be included into the trial set G for an ending-guessing rule is to have the necessary ending:

$$n_i^e = |G| \quad : \quad G = \{w \in T \mid (S_w \dashv S_i) \neq \emptyset\}$$

The number of positive observations for a rule can be computed as the cardinality of the subset of the set G where all the words have the same POS-class as the resulting class R of the rule:

$$x_i = |H| \quad : \quad H = \{w \in G \mid C_w = R_i\}$$

It is trivial to prove that within the same set of rules the sum of the estimate probabilities for all guessing rules with the same affix S and the same R-class is less than or equal to 1.

The $\hat{p}$ estimate is a good indicator of the rule accuracy, however, it frequently suffers from large *estimation error* due to insufficient training data. For example, if a rule was detected to work 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 because of the tiny size of the sample. Several smoothing methods (e.g. Good, 1953; Katz, 1987; Salton, 1985) have been proposed to reduce the estimation error. For different reasons all these smoothing methods are not very suitable in our case. For example, the Good–Turing formula (Good, 1953) for smoothing the maximum likelihood estimate operates on the same sample space for all the events. As we showed above each guessing rule has its own sample space consisting of all compatible words and therefore the Good–Turing method is not straightforwardly applicable to this case. The Katz's *back-off* procedure operates over n-gram parameters which is not the case with the guessing rules. In our approach we tackle this problem by calculating the *lower confidence limit* π_L for the rule estimate. This 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 limit. The rule estimate then will be taken at its lowest possible value which is the π_L limit itself. It can be shown (Hayslett, 1981, p.103) that under a binomial experiment with probability of success on a single trial π and the trial size n, the random variable **p** (the observed proportion of success) has a distribution approximately normal, with mean π and variance $\pi(1-\pi)/n$. Note also that we have already cut out the most infrequent rules which could have violated the assumption of normality. To calculate π_L the standard procedure of inferential statistics is used:

- decide upon the degree of confidence α and calculate the standard normal variable $z_{(1-\alpha)/2}$. For instance, if we want an approximate 95% confident interval the degree of confidence α is 0.95. From the standard normal tables we can find the value for $z_{0.025}$ which is 1.96.
- calculate the estimate of the true variance, which for a particular value of **p** - $\hat{p}$ is:
  $s_p = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

So as not to have zeros in positive ($\hat{p}$) or negative ($1-\hat{p}$) outcome probabilities, we smooth the rule estimate as: $\hat{p}_i^* = \frac{x_i+0.5}{n_i+1}$.

- the lower confidence limit π_L then is calculated as
  $\pi_L = \hat{p}^* - z_{(1-\alpha)/2} * s_p = \hat{p}^* - z_{(1-\alpha)/2} * \sqrt{\frac{\hat{p}^*(1-\hat{p}^*)}{n}}$
  This function favors the rules with higher estimates obtained over larger samples. 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 over a large sample might be valued higher.

Such estimation of the rule value, in fact, resembles the one used by Tzoukermann *et al.* (1995) for scoring POS-disambiguation rules for the French tagger. The main difference between the two functions is that there the z value was implicitly assumed to be 1, which corresponds to the confidence of 68%. A more standard approach is to adopt a rather high confidence value in the range of 90–95%. We adopted 90% confidence for which $z_{(1-0.90)/2} = z_{0.05} = 1.65$. Thus we can calculate the score for the i*th* rule as

$$\hat{p}_i^* - 1.65 * \sqrt{\frac{\hat{p}_i^*(1-\hat{p}_i^*)}{n_i}}.$$

Another important consideration for scoring a word-guessing rule is that the longer the affix or ending of the 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 five words and the estimate for the word ending 'fulness' was also obtained over a sample of five words, the later 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 that 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:

$$score_i = \hat{p}_i^* - 1.65 * \sqrt{\frac{\hat{p}_i^*(1-\hat{p}_i^*)}{n_i}}/(1 + \log(|S_i|)).$$

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 ($theta_s$) at a certain level permits only those rules whose scores are higher than the threshold to be included into the final rule-sets. The method for setting up the threshold is based on empirical evaluations of the rule-sets and is described in section 3.2. Table 2 shows the best scored rules from the three rule-sets induced from the Brown Corpus data.

3.1.3 Rule merging phase

Rules which have scored lower than the threshold can be merged together in more general rules. These rules if scored above the threshold can also be included into the final rule-sets. We can merge together the rules which have scored below the

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Table 2. *Top scored prefix, suffix and ending guessing rules.*

Prefix	***I*-class**	***R*-class**	**Suffix**	***I*-class**	***R*-class**	**Ending**	***R*-class**
un	JJ	JJ	ness	JJ	NN	tral	JJ
un	JJ NN	JJ	ion	VB	NN	ness	NN
self-	NN	NN	's	NN	NN$	ities	NNS
ad	NN	NN	ers	NN VB	NNS	gery	NN
non	JJ	JJ	'	NNS	NNS$	ility	NN
un	JJ RB	JJ	able	NN VB	JJ	rged	VBD VBN
in	JJ	JJ	ness	JJ RB	NN	tment	NN
ir	JJ	JJ	s	NN VBG	NNS	gical	JJ
re	VBD VBN	VBD VBN	ity	JJ NN	NN	sm	NN
pre	JJ	JJ	s	JJ NN	NNS	ings	NNS

threshold and have the same affix (S) and the initial class (I)[12]. We define the rule merging operator $\oplus$:

$$A_i \oplus A_j = A_r : [S_i, I_i, R_i \cup R_j] \text{ if } S_i = S_j \text{ and } I_i = I_j.$$

The system merges two rules with the same affix (S) and the initial class (I) into one rule with the resulting class being the union of the two merged resulting classes. For example,

$$[\dashv s\ ?(NN\ VB) \rightarrow (NNS)] \oplus [\dashv s\ ?(NN\ VB) \rightarrow (NNS\ VBZ)]$$
$$= [\dashv s\ ?(NN\ VB) \rightarrow (NNS\ VBZ)]$$
$$[\vdash un\ ?(VBD\ VBN) \rightarrow (JJ)] \oplus [\vdash un\ ?(VBD\ VBN) \rightarrow (VBN)]$$
$$= [\vdash un\ ?(VBD\ VBN) \rightarrow (JJ\ VBN)]$$

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 substitutes the two merged ones. To perform such rule-merging over a rule-set, first, the rules which have not been included into the final set are sorted by their score and best-scored rules 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, in which case it is added to the final rule-sets. This process is applied until no merges can be done to the low-scored rules.

3.2 Direct evaluation of the rule-sets

There are two important questions which arise at the rule acquisition stage – how to choose the scoring threshold θ_s, and what is the performance of the rule-sets produced with different thresholds. 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 should be assigned with a set of descriptors which represent its contents. There are a

[12] For ending-guessing rules this is always true.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

number of standard parameters which are used for measuring performance on this kind of task. For example, suppose that a word can take on one or more categories from the set of open class categories: (JJ NN NN$ NNS NNS$ QL RB VB VBD VBG VBN VBZ). To see how well the guessing performs we can compare the results of the guessing with the set of POSs which are known to be true for the word. Let us take, for instance, the word [developed (JJ VBD VBN)]. Suppose that the guesser categorized it as: [developed (JJ NN QL VBD VBZ)]. We can represent this situation as follows:

All	JJ	NN	NN$	NNS	NNS$	QL	RB	VB	VBD	VBG	VBN	VBZ
Real	∨								∨		∨	
Guess	∨	∨				∨			∨			∨

The performance of such guessing can be measured in:

1. *recall* – the percentage of POS-tags which were assigned correctly by the guesser, 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. So, for example if the guesser assigned all possible POS-tags to the word its recall would have been 100%.
2. *precision* – the percentage of POS-tags the word-class guesser assigned correctly (JJ VBD) over the total number of POS-tags it assigned to the word (JJ NN QL 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%.
3. *coverage* – the proportion of words which the guesser was able to classify, but not necessarily that classification was correct. 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 higher the coverage, the more unknown words can be handled.

The interpretation of these percentages is by no means straightforward. For example, the measures assume that all combinations of POS-tags will be equally hard to disambiguate for the tagger, which is not necessarily the case. There are also different ways of calculating these measures (Lewis, 1991):

1. macro-averaging – the average measures per a collection of words, i.e. first, all the words from the test collection are guessed and then macro-precision and macro-recall are calculated by averaging over true positives and negatives for all the words;
2. micro-averaging – the average measures per word, 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;

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Table 3. *Best-scored rule-sets: Prefix 80 – prefix morphological rules which scored over 80 points, Suffix 60 – suffix morphological rules which scored over 60 points and Ending 75 – ending-guessing rules which scored over 75 points. As the base-line model was taken the ending guesser developed by Xerox.*

Measure	**Test Data**	**Xerox**	**Ending 75**	**Suffix 60**	**Prefix 80**
Recall	Lexicon	0.956313	0.945726	0.95761	0.955748
	Corpus	0.944526	0.952016	0.97352	0.978515
Precision	Lexicon	0.460761	0.675122	0 .919796	0.922534
	Corpus	0.523965	0.745339	0.979351	0.977633
Coverage	Lexicon	0.917698	0.977089	0.37597	0.049558
	Corpus	0.893275	0.96104	0.320996	0.058372

3. descriptor measure – we can also measure the precision and recall not for a word but for a particular POS-tag. Thus we can find out which categories are assigned with a high level of confidence and which are more uncertain.

In our experiments we measured word precision and word recall (micro-average).

There were two types of test data in use at this stage. First, we evaluated the guessing rules against the actual lexicon: every word from the lexicon, except for closed-class words and words shorter than five characters[13], was guessed by different rule-sets and the results were compared with the information the word had in the lexicon. In the second evaluation experiment, we measured the performance of the guessing rules against the training corpus. For every word we computed its metrics exactly as in the previous experiment. Then we multiplied these results by the corpus frequency of this particular word and averaged them. Thus the most frequent words had the greatest influence on the aggregate measures.

3.2.1 Finding the optimal threshold

First we concentrated on finding out the best thresholds θ_s which had been used for the filtering out final rule-sets. To do that for each rule-set produced using different thresholds we recorded its precision, recall and coverage, and chose the one with the best metrics. We noticed certain regularities in the behaviour of the metrics in response to the change of the threshold. The *recall* improves along with the increase of the threshold while the *coverage* drops proportionally. This is not surprising: the higher the threshold, the fewer inaccurate rules are included into the rule-set but at the same time fewer words can be handled. An interesting behaviour is shown by the *precision*. First, it grows proportionally along with the increase of the threshold but then at high thresholds it drops. This means that among very confident rules with very high scores there are many quite general ones. Table 3 displays the metrics for the best scored (in aggregate) thresholds for the three rule-sets. The best scoring

[13] The actual size of the filtered lexicon was 47,659 entries out of 53,015 entries of the original Brown Corpus lexicon.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

thresholds were detected: for ending rules – 75 points, for suffix rules – 60, and for prefix rules – 80. In theory one would expect the best thresholds to be set rather high, say in the 90–95 points range, for obtaining accurate guessings. The reason why the best thresholds were detected as low as 60–75 points is that some false applications of a rule are eliminated by other rules from the rule-set which are applied before. Another reason is that the rule scoring function described in section 3.1.2 operates only on the *exact* match of the guess with the POS-class from the lexicon whereas this evaluation is more flexible and uses two metrics (precision and recall) instead of one.

One can notice a slight difference in the results obtained over the lexicon and the corpus. The corpus results are better because the rule scoring method explicitly targeted the rule-sets to the most frequent cases of the corpus rather than the lexicon. Ending guessing rules as we predicted cover over 96% of the unknown words. The precision of 74% roughly can be interpreted as that for words which take on three different POSs in their POS-classes, the ending-guessing rules will assign four, but in 95% of the times (recall) the three required POSs will be among the four assigned by the guess. In comparison with the Xerox word-ending guesser taken as the base-line, we detect a substantial increase in precision by about 22% and in coverage by about 6%. This means that the Xerox guesser creates more ambiguity for the disambiguator assigning five instead of three POSs for the example above. It also handles 6% fewer unknown words which, in fact, might decrease its performance even more. In comparison with the ending guessing rules, the morphological rules have much better precision, and, hence, better accuracy of guessing. Virtually almost every word which can be guessed by the morphological rules is guessed exactly correct (97% recall and 97% precision). Not surprisingly, the coverage of morphological rules is much lower than that of the ending-guessing ones – for the suffix rules it is less than 40% and for the prefix rules about 5–6%.

After obtaining the optimal rule-sets we performed the same experiment on a word-sample which was not included into the training lexicon and corpus. We gathered about three thousand words[14] from the lexicon developed for the *Wall Street Journal* corpus and collected frequencies of these words in this corpus. At this experiment we obtained similar metrics apart from the coverage which dropped about 0.5% for Ending 75 and Xerox rule-sets and 7% for the Suffix 60 rule-set. This, actually, did not come as a surprise, since many base forms required by the suffix rules were missing from the lexicon.

3.2.2 Measuring cascading performance

In the next experiment we evaluated whether the morphological rules add any improvement if they are used in conjunction with the ending-guessing rules. We also evaluated in detail whether a conjunctive application with the Xerox guesser would boost the performance. As in the previous experiment we measured precision,

[14] These words were not listed in the training lexicon.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Table 4. *Performance of the cascading application of the rule-sets over the training lexicon and training corpus.* P_{80} *– prefix rule-set scored over 80 points,* S_{60} *– suffix rule-set scored over 60 points,* E_{75} *– ending guessing rule-set scored over 75 points. As the base-line model we used the ending guesser developed by Xerox (X) for the Brown Corpus. The first part of the table shows that the* E_{75} *rule-set outperforms and fully supercedes the Xerox rule-set. The second part of the table shows that the cascading application of the morphological rule-sets together with the ending guessing rules increases the performance in precision by about 5%.*

Guessing Strategy	Lexicon			Corpus		
	Precision	**Recall**	**Coverage**	**Precision**	**Recall**	**Coverage**
Xerox (X)	0.460761	0.956331	0.917698	0.523965	0.944526	0.893275
Ending 75 (E_{75})	0.675122	0.945726	0.977089	0.745339	0.952016	0.96104
X+E_{75}	0.470249	0.95783	0.989843	0.519715	0.949789	0.969023
E_{75}+X	0.670741	0.943319	0.989843	0.743932	0.951541	0.969023
P_{80}+E_{75}	0.687126	0.946208	0.977488	0.748922	0.951563	0.96104
S_{60}+E_{75}	0.734143	0.945015	0.979686	0.792901	0.951015	0.963289
P_{80}+S_{60}+E_{75}	0.745504	0.945445	0.980086	0.796252	0.950562	0.963289

recall and coverage both on the lexicon and on the corpus. Table 4 demonstrates the results of this experiment. The first part of the table shows that when the Xerox guesser is applied before the E_{75} guesser we measure a drop in the performance. When the Xerox guesser is applied after the E_{75} guesser no sufficient changes to the performance are noticed. This actually proves that the E_{75} rule-set fully supercedes the Xerox rule-set. The second part of the table shows that the cascading application of the morphological rule-sets together with the ending-guessing rules increases the overall precision of the guessing by a further 5%. This makes the improvements against the base-line Xerox guesser 28% in precision and 7% in coverage.

4 Tagging with unknown words

The direct evaluation of the rule sets gave us the basis for the comparison and selection of the best performing guessing-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. The first one was a C++ implemented HMM tagger akin to the LISP implemented HMM Xerox tagger described in Kupiec (1992). The other tagger was the rule-based tagger of Brill (Brill, 1995). Both taggers come with data and word-guessing components pre-trained on the Brown Corpus which we re-used in our experiment[15]. This actually gave us the search-space of

[15] Since Brill's tagger uses a slightly modified version of the Brown Corpus tag-set known as the Penn Brown Corpus tag-set (Marcus *et al.*, 1993), we provided an additional mapping.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

four combinations: the HMM tagger equipped with the Xerox guesser, the Brill tagger with its original guesser, the HMM tagger with our cascading $P_{80}+S_{60}+E_{75}$ guesser and the Brill tagger with the cascading guesser. For words which failed to be guessed by the guessing rules we applied the standard method of classifying them as common nouns (NN) if they are not capitalized inside a sentence and proper nouns (NP) otherwise. As the base-line result we measured the performance of the taggers with all known words on the same word sample.

In the evaluation of tagging accuracy on unknown words we pay attention to two metrics. First we measure the accuracy of tagging solely on unknown words:

$$Unknown\ Score = \frac{Correctly\ \ Tagged\ \ Unkown\ \ Words}{Total\ \ Unknown\ \ Words}.$$

This metric gives us the exact measure of how a tagger performs on unknown words when equipped with a particular guessing module. In this case, however, we do not account for the known words which were mis-tagged because of the unknown ones. To put a perspective on that aspect we measure the overall tagging performance:

$$Total\ Score = \frac{Correctly\ \ Tagged\ \ Words}{Total\ \ Words}.$$

Since the Brown Corpus model is a general language model, it, in principal, does not put restrictions on the type of texts it can be used for, but its performance on some texts might be lower than that of a model specialised for a particular sublanguage. Here we want to stress that our primary task was not to evaluate or compare the taggers themselves, but rather their performance with the word-guessing modules. Therefore, we did not worry too much about tuning the taggers for the texts, and used the Brown Corpus model instead. By the same reason we did not evaluate the two taggers on the same tag-set.

We tagged several texts of different origins, except from the Brown Corpus. These texts were not seen at the training phase which means that neither the taggers nor the guessers had been trained on these texts, and they naturally had words unknown to the lexicon. For each text we performed two tagging experiments. In the first experiment we tagged the text using the Brown Corpus lexicon supplied with the taggers, and hence had only those unknown words which naturally occured in the text. In the second experiment we tagged the same text with the lexicon of closed-class[16] and short[17] words. This small lexicon contained only 5456 entries out of 53,015 entries of the original Brown Corpus lexicon. All other words were considered as unknown and had to be guessed by the guessers. We obtained quite stable results in these experiments.

Here is a typical example of tagging a text of 5970 words. This text was detected to have 347 unknown words. First, we tagged the text by the four different combinations of two taggers with three word-guessing components using the full-fledged lexicon. The results of this tagging are summarized in Table 5. When using the HMM tagger with the Xerox guesser, 63 unknown words were incorrectly tagged and the accuracy on the unknown words was measured at 81.8%. When the HMM tagger

[16] Articles, prepositions, conjunctions, etc.
[17] shorter than five characters.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Table 5. *Results of tagging a text with 347 unknown words by four different combinations of two taggers with three word-guessing modules.*

Tagger	**Guessing strategy**	**Total words**	**Unkn. words**	**Total mistag.**	**Unkn. mistag.**	**Total Score (%)**	**Unkn. Score (%)**
HMM	Xerox	5970	347	324	63	94.3	81.8
HMM	P_{80} +S_{60} +E_{75}	5970	347	292	33	95.1	90.5
Brill	Brill	5970	347	246	54	95.9	84.5
Brill	P_{80} +S_{60} +E_{75}	5970	347	219	27	96.3	92.2

was equipped with our cascading guesser its accuracy on unknown words increased by almost 9%. The score distribution among the guessing rules fell as follows:

Rule-Set	P 80	S 60	E 75	Xerox	NN len>3	NN len<4	Total
Correct/All	–	–	–	126/182	61/68	97/97	284/347
% Correct	–	–	–	69.2%	90%	100%	81.8%
Coverage	–	–	–	52%	20%	28%	100%
Correct/All	5/5	58/62	121/148	–	33/35	97/97	314/347
% Correct	100%	93.5%	81.7%	–	94.3 %	100%	90.5%
Coverage	1.5%	17.5%	43%	–	10%	28%	100%

As shown, the proper noun assignment (NN) worked exceptionally well for the cases which failed to be handled by the guessers. For words longer than three characters (len>3) the NN assignment scored above 90%, and for words less than four characters length (len<4) the score was 100%. The real accuracy of unknown word tagging with the Xerox guesser was 69.2% in contrast to 85.6% of the real joint accuracy of tagging with the cascading guesser. This difference of more than 16% is due to the difference in precision between the Xerox guesser and the cascading guesser. The cascading guesser also handled 10% more words than the Xerox guesser but this was partially compensated for by the exceptional performance of the NN assignment.

Similar results (Table 5) were detected with the Brill tagger which in general was slightly more accurate than the HMM one[18]. The cascading guesser performed better than the original Brill guesser by about 8%, boosting the performance on the unknown words from 84.5%[19] to 92.2%. Another interesting thing to notice is that the Brill tagger utilised the cascading guessing rules better than the HMM tagger. This can partly be accounted for by the fact that the cascading guessing rules for

[18] This, however, was not an entirely fair comparison because of the differences in the tag-sets in use by the taggers. The HMM tagger was trained on the original Brown Corpus tag-set which makes more distinctions between categories than the Penn Brown Corpus tag-set.

[19] Which agrees with quoted by Brill (Brill, 1994) 85%.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Table 6. *Results of tagging the same as in Table 5 text by four different combinations of two taggers with three word-guessing modules using the small lexicon (closed-class and short words).*

Tagger	Guessing strategy	Total words	Unkn. words	Total mistag.	Unkn. mistag.	Total Score (%)	Unkn. Score (%)
HMM	Xerox	5970	2215	556	516	90.7	76.7
HMM	$P_{80}+S_{60}+E_{75}$	5970	2215	332	309	94.44	86.05
Brill	Brill	5970	2215	464	410	93.1	81.5
Brill	$P_{80}+S_{60}+E_{75}$	5970	2215	327	287	94.52	87.45

the Brill tagger listed the most likely parts-of-speech for unknown words first[20] as required by the Brill tagger whereas the HMM tagger was not provided with such preferences. When the 347 unknown words were added to the lexicon both taggers scored 98.5% on these words.

In the second experiment we tagged the same text using only the small lexicon. Out of 5970 words of the text, 2215 were unknown to the small lexicon. The results of this tagging are summarised in Table 6. The accuracy of the taggers on the 2215 unknown words when they were made known to the lexicon was much lower than in the previous experiment — 90.3% for the HMM tagger and 91.5% for Brill's tagger. Naturally, the performance of the guessers was also lower than in the previous experiment plus the fact that many 'semi-closed' class adverbs like 'however', 'instead', etc., were missing from the small lexicon. The accuracy of the tagging on unknown words dropped by about 5% in general. The best results on unknown words were again obtained on the cascading guesser (86–87.45%) and Brill's tagger again did better than the HMM one by about 1%.

To evaluate the guessing components with a highly specialised language model we performed a similar tagging experiment on the Map Task Corpus (Anderson *et al.*1991). This is a corpus of problem solving spontaneous dialogues which consists of separate transcriptions for each speaker. There was trained data available only for the HMM tagger, so we did not evaluate the guesser with the Brill tagger in this experiment. Since the Map Task corpus did not include words unknown to the tagger's lexicon, we tagged the text with the small lexicon of closed-class and short words. It is well known that optimisation of an HMM tagger requires significant amounts of work in changing the lexicon and initial values for the probability matrices. Indeed, the lexicon for the Map Task Corpus appeared to be highly specialised and listed only 4600 words with very restricted sets of POS categories. Due to the small size of the lexicon we could not train the guessing component for that corpus. So we decided to reuse the guessing rules developed for the Brown Corpus which naturally provided more general categories than were listed in the

[20] This was done by a very simple heuristic to put the NN (noun) category first in all POS-classes which included this category, and by arranging the JJ VBD VBN class as VBN VBD JJ.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Table 7. *Results of tagging the Map Task corpus by the HMM tagger with the Xerox guesser and the cascading guesser.*

Tagger	Guessing strategy	Total words	Unkn. words	Total mistag.	Unkn. mistag.	Total Score (%)	Unkn. Score (%)
HMM	Xerox	183,340	21,537	7687	4859	95.8	77.44
HMM	$P_{80}+S_{60}+E_{75}$	183,340	21,537	5757	3026	96.86	85.95

lexicon. For example, the word 'abandoned' was marked only as an adjective (JJ) in the Map Task lexicon but was guessed as adjective (JJ), past verb (VBD) and participle (VBN).

The results of tagging the Map Task corpus by the HMM tagger with the small lexicon are summarized in Table 7. The Map Task corpus consists of 183,340 words from which only 21,537 did not belong to the closed-class and short words. After we tagged the corpus using the cascading guesser we mis-tagged 5757 words from which 3026 were unknown to the lexicon. This corresponds to 85.95% of the tagging accuracy on unknown words and 96.86% of the overall tagging accuracy. Using the Xerox guesser the accuracy on the unknown words was 77.44% and the overall tagging accuracy 95.8%. These results were consistent with the results obtained for the HMM tagger in the previous experiment. Tagging accuracy on the unknown words when they were made known to the lexicon was 99.9% which means that many of them were listed in the lexicon as unambiguous. Such specialisation of the lexicon can be seen, in fact, as a limitation of the specialised sublanguage approach to tagging – achieving good results on seen data it is not suitable for scaling up.

There were three types of mis-taggings on unknown words detected in our experiments. Mis-tagging of the first type occured when the guesser provided broader POS-classes for unknown words and the tagger had difficulties with the disambiguation of such broader classes. This is especially the case with words which are guessed as a noun and an adjective simultaneously but are listed only as one of them in the lexicon as, for example, the word 'brain'. Another highly ambiguous group of words are the 'ing' words which, in general, can act as nouns, adjectives and gerunds and only the direct lexicalisation can restrict the search space, as in the case with the word 'going', which cannot be an adjective but only a noun and a gerund. The second type of mis-tagging was caused by the wrong assignments of POSs by the guesser. Usually this was the case with irregular words such as 'cattle' or 'data' which were wrongly guessed as singular nouns (NN), but in fact were plural nouns (NNS). And the third type of the mis-tagging occured when the word-guesser assigned the correct POS-class to a word, but the tagger still disambiguated this class wrongly. Considering the accuracy of tagging on the unknown words when they all were made known to the lexicon of 90–92% and tagging accuracy of 86–88% with these words unknown, this is the most frequent type of errors which account for more than 60% of the mis-taggings. Naturally, this cannot be considered as a fault of the guessers.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

5 Discussion and conclusion

We presented a technique for fully unsupervised statistical acquisition of rules which guess possible parts of speech for words unknown to the lexicon. This technique does not require specially prepared training data and uses for training the lexicon of the tagger and word frequencies collected from a raw corpus. Using these training data three types of guessing rules are induced: prefix morphological rules, suffix morphological rules and ending guessing rules. To select best performing guessing rule-sets we suggested an evaluation methodology, which is solely dedicated to the performance of part-of-speech guessers.

Evaluation of tagging accuracy on unknown words using texts unseen by the guessers and the taggers 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%). The cascading guesser outperformed the guesser supplied with the Xerox tagger by about 8–9% and the guesser supplied with Brill's tagger by about 6–7%. Tagging accuracy on unknown words using the cascading guesser was detected at 90–92% when tagging with the full-fledged lexicon and 86–88% when tagging with the closed-class and short-word lexicon. When the unknown words were made known to the lexicon the accuracy of tagging was detected at 96–98% and 90–92%, respectively. This 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 morphological guessing rules do improve the guessing performance. Since they are more accurate than ending-guessing rules they are applied first and improve the precision of the guessings by about 5%. This, actually, results in about 2% higher accuracy of tagging on unknown words.

The acquired guessing rules employed in our cascading guesser are, in fact, of a standard nature and in that form or another are used in other POS-guessers. There are, however, a few points which make the rule-sets acquired by the presented here technique more accurate:

- the acquisition of such rules is done from a general purpose lexicon rather than tagged corpus, because the guesser's task is akin to the lexicon lookup;
- there is a well-tuned statistical scoring function which accounts for word features and frequency distribution;
- there is an empirical way to determine an optimum collection of rules, since acquired rules are subject to rigorous direct evaluation in terms of precision, recall and coverage;
- rules are applied cascadingly using the most accurate rules first.

Another important advantage of the proposed technique is that the development of data for training is considerably simpler than the development of the training examples or a correctly tagged corpus required by other approaches. The quantitative requirements are less demanding too. In our approach, guessing rules are induced from the lexicon, and the actual corpus frequencies of word-usage then allow for 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. Thus the

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

major factor in the learning process is the lexicon. Since guessing rules are meant to capture general language regularities, the lexicon should be as general as possible (list *all* possible POS-tags for a word) and as large as possible. The corresponding corpus should include most of the words from the lexicon and be large enough to obtain reliable estimates of word-frequency distribution. Our experiments with the lexicon and word frequencies derived from the Brown Corpus, which can be considered as a general model of English, resulted in guessing rule-sets which proved to be domain and corpus independent[21], producing similar results on test texts of different origins.

Although in general the performance of the cascading guesser was only 6% lower than the lookup of a general language lexicon, there is room for improvement. First, in the extraction of the morphological rules we did not attempt to model non-concatenative cases. In English, however, since most of letter mutations occur in the last letter of the stems it is possible to account for it. So our next goal is to extract morphological rules with one letter mutations at the end. This would account for cases like 'try - tries', 'reduce - reducing', 'advise - advisable'. We expect it to increase the coverage of the suffix morphological rules and hence contribute to the overall guessing accuracy. Another avenue for improvement is to provide the guessing rules with the probabilities of emission of POS-tags from their resulting POS-classes (*R*-classes). This information can be compiled automatically and also might improve the accuracy of unknown word tagging.

The described rule acquisition and evaluation methods are implemented as a modular set of C++ and AWK tools, and the guesser is easily extendable to sub-language specific regularities and retrainable to new tag-sets and other languages, provided that these languages have affixational morphology.

Acknowledgements

I would like to thank Chris Brew for helpful discussions on the issues related to this paper.

References

Anderson, A.H., Bader, M., Bard, E.G., Boyle, E., Doherty, G., Garrod, S., Isard, S., Kowtko, J., McAllister, J., Miller, J., Sotillo, C., Thompson, H. and Weinert, R. (1991) The HCRC Map Task Corpus. *Language and Speech*, **34**(4): 351–366.

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

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

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

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

[21] But tag-set dependent.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press

Cutting, D., Kupiec, J., Pedersen, J. and Sibun. P. (1992) A practical part-of-speech tagger. *Proceedings of the Third Conference on Applied Natural Language Processing (ANLP-92)*, Paris, France, pp. 133–140.

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

Finch, S. and Chater, N. (1993) Learning syntactic categories: a statistical approach. In M. R. Oaksford & G. D. A. Brown (eds), *Neurodynamics and Psychology*, London:Harcourt Brace & Co. pp. 295–322.

Francis, W. and Kucera, H. (1982) *Frequency Analysis of English Usage*. Boston, MA: Houghton Mifflin.

Good, I. J. (1953) The population frequencies of species and the estimation of population parameters. *Biometrika*, **40**(3, 4): 237–264.

Hayslett, H. (1981) *Statistics Made Simple*. Heinemann: London.

Katz, S. M. (1987) Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans, Acoustic, Speech, and Signal Processing*, **35**(3): 400–401.

Knuth, D. (1973) *The Art of Computer Programming. Volume 3. Sorting and Searching. Reading*, MA: Addison-Wesley

Kupiec, J. (1992) Robust part-of-speech tagging using a Hidden Markov Model. *Computer Speech and Language*.

Lewis, D. (1991) Evaluating text categorization. *Speech and Natural Language: Proceedings of a Workshop held at Pacific Grove CA*, Los Altos, CA: Morgan Kaufmann

Marcus, M., Marcinkiewicz, M.A. and Santorini, B. (1993) Building a large annotated corpus of English: Penn Treebank. *Computational Linguistics*, **19**(2): 313–329.

Mikheev, A. (1996) Unsupervised learning of word-category guessing rules. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-96)*, Santa Cruz, USA, pp. 327–334.

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

Salton, G. (1985) *Automatic Text Processing. The Transformation, Analysis, and Retrieval of Information by Computer*. Cornell University: Addison-Wesley.

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

Tzoukermann, E., Radev, D. R. and Gale, W. A. (1995) Combining linguistic knowledge and statistical learning in French part of speech tagging. *EACL SIGDAT Workshop*, Dublin, Ireland, pp. 51–59.

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

Weischedel, R., Meteer, M., Schwartz, R., Ramshaw, L. and Palmucci, J. (1993) Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics*, **19**(2): 359–382.

Byoung-Tak Zhang 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)*, Helsinki, Finland, 431–435.

https://doi.org/10.1017/S1351324996001271 Published online by Cambridge University Press