Integrating External Dictionaries into Stochastic Part-of-Speech Taggers*

Jorge Graña Gil, Jean-Cédric Chappelier and **Manuel Vilares Ferro**
Computer Science Department, University of Corunna (Spain)
{grana,vilares}@dc.fi.udc.es
Computer Science Department, Swiss Federal Institute of Technology in Lausanne (Switzerland)
Jean-Cedric.Chappelier@epfl.ch

Abstract

The highest performances in part-of-speech tagging have been obtained by using stochastic methods, such as hidden Markov models. The running parameters of a hidden Markov model for tagging can be estimated from tagged corpora. However, the current situation in the automatic processing of some languages is very short training texts, but very large dictionaries. These dictionaries can provide very useful information for improving the treatment of unknown words. In this paper we present new strategies for integrating external dictionaries into a stochastic tagging framework. Instead of the most intuitive Adding One method, we propose the use of the Good-Turing formulas, which produce less distortion of the model we are estimating. This technique guarantees good performances in the automatic processing of languages for which reference texts hardly exist.

1 Introduction

The ultimate goal of research on Natural Language Processing is to parse and understand human languages. Currently, we are still far from achieving this goal. For this reason, much research in computational linguistics has focussed on intermediate tasks that make sense of some of the structure inherent in language without requiring complete understanding. One such task is part-of-speech tagging, or simply tagging.

Elimination of lexical ambiguities is a crucial task during the process of tagging a text in natural language. If we take in isolation, for instance, the word `time`, we can see that it has several possible tags in English: noun, adjective or verb. However, if we examine the context in which the word appears, on most occasions only one of the tags is possible. In addition, we are also interested in being able to give a tag to all the words that appear in a text, but are not present in our dictionary, and to guarantee somehow that this tag is the correct one. A good performance at this stage improves the viability of syntactic and semantic analysis.

Traditionally, the starting point for tagging is linguistic resources like dictionaries and written texts, previously tagged or not. This research line is called corpus-based linguistics. These corpora are used to tune the running parameters of the taggers. This tuning process is called training. The highest performances have been obtained by using stochastic methods, such as hidden Markov models (Brants 2000), and taking tagged corpora as training data.

Many other non-quantitative methods have been applied to tagging. One of them is EngCG (English Constraint Grammar), which performs better than Markov model taggers, especially if training and application corpora are not from the same source (Samuelsson & Voutilainen 1997). The accuracy figures for EngCG in that paper are better than 99% with a set of 1,000 constraint rules vs. better than 95% for a Markov model tagger, but comparison is difficult since some ambiguities are not resolved by EngCG. EngCG returns a set of more than one tag in some cases. Moreover, the following aspects must be considered:

- The success of this approach is partly due to integrating a large dictionary with the tagger, which is precisely the theme of the present work.
- There are many cases where using this kind of rule-based tagger is not possible.
- This methodology amounts to writing a small expert system for tagging. The claim has been made that for somebody who is familiar with the methodology, writing this type of tagger takes no more effort than building a Markov model tagger (Chanod & Tapanainen 1995), thought it could be ar-

* This work has been partially supported by the European Union (under FEDER project 1FD97-0047-C04-02), by the Spanish Government (under project TIC2000-0370-C02-01), and by the Galician Government (under project PGIDT99XI10502B).

gued that the methodology for Markov model tagging is more easily accessible.

For all these reasons, the discussion has focussed on integrating external dictionaries into a stochastic tagging framework.

A hidden Markov model for tagging needs two sets of running parameters: the n-grams probabilities for the transitions between tags, and the emission probabilities for words. Both can be estimated from a tagged corpus. However, sometimes we have to deal with languages with very few available linguistic resources, such as Spanish. Currently, the typical situation in Spanish processing is very short training texts, since they are not common resources, but very large dictionaries, since the morphology of the language is well-known and the effort made to formalize it has been much greater.

The standard approach to improve the model in these cases consists of applying on raw data unsupervised learning techniques, such as the Baum-Welch algorithm (Baum 1972; Dempster *et al.* 1977). Due to the difficulties to control these techniques, the alternative method addressed in this paper is to base the improvement on the use of dictionaries. External dictionaries should not be ignored anyway, because they provide very useful information for improving the treatment of unknown words. Therefore, the goal of the present work is to implement new strategies for integrating external dictionaries into a stochastic tagging framework, and to verify the way in which the use of these dictionaries can help to increase the performance of the tagging process, especially when the training corpus is small.

The most intuitive way to perform this integration is the Adding One method (Church 1988), which consists of using the dictionary as an additional tagged corpus where a frequency of 1 is assigned to each word-tag pair. However, this integration does not produce a coherent representation of the model we are estimating, and it can produce important alterations in the working parameters. This leads us to consider another method based on the Good-Turing formulas (Church & Gale 1991; Jelinek 1997). Every word-tag pair present only in the external dictionary can be seen as an event with null frequency in the training corpus, and the Good-Turing formulas are themselves a method able to assign probabilities greater than 0 to these rare but existing events. In addition, this technique produces less distortion of the model.

2 Architecture of the tagger

Our system uses a second order Markov model for part-of-speech tagging. In this section, we describe in detail the elements of the model and the procedures to estimate its working parameters.

2.1 The underlying model

The states of the model represent pairs of tags, and outputs represent the words. Transition probabilities depend on the states, thus pairs of tags. Output probabilities only depend on the most recent category. To be explicit, we calculate

$$\arg\max_{t_1 \ldots t_n} \prod_{i=1}^{n} [P(w_i|t_i) \times P(t_i|t_{i-2}, t_{i-1})]$$

for a given sentence of words $w_1 \ldots w_n$ of length n, where $t_1 \ldots t_n$ are elements of the tagset.

Transition and output probabilities are estimated from a tagged corpus. As a first step, we use the maximum likelihood probabilities $\hat{P}$ which are derived from the relative frequencies:

$$\begin{aligned} \text{Unigrams}: & \quad \hat{P}(t_3) = \tfrac{C(t_3)}{N} \\ \text{Bigrams}: & \quad \hat{P}(t_3|t_2) = \tfrac{C(t_2,t_3)}{C(t_2)} \\ \text{Trigrams}: & \quad \hat{P}(t_3|t_1,t_2) = \tfrac{C(t_1,t_2,t_3)}{C(t_1,t_2)} \\ \text{Lexical}: & \quad \hat{P}(w_3|t_3) = \tfrac{C(w_3,t_3)}{C(t_3)} \end{aligned}$$

for all t_1, t_2, t_3 in the tagset and w_3 in the training corpus. $C(x)$ is the number of times the event x occurs in the training corpus, and N is the total number of tokens in the training corpus. We define a maximum likelihood probability to be zero if the corresponding nominator is null. As a second step, contextual frequencies are smoothed and lexical frequencies are completed by handling words that do not appear in the training corpus but are present in external dictionaries of the corresponding language.

Note that the lexical probabilities, which we will call *output* or *emission probabilities*, do not come from more intuitive percentages such as: 70% of the occurrences of the word `time` are as noun, 20% are as adjective and the remaining 10% are as verb. The corresponding probabilities (0.7, 0.2 and 0.1) would be $P(t|w)$ (note that $\sum_t P(t|w) = 1$), but the model uses precisely the opposite ones, i.e. $P(w|t)$.

To understand these later probabilities in a natural way, we resort to the simulation of a hidden Markov model by an urn-and-ball system (Rabiner & Juang 1993). In this system, we have a different urn for each tag t in the tagset. For each pair (w,t) in the training corpus, the word w is added to the urn t in a such a way that the same word could be in different urns (due to the ambiguities) and it also could be in the same urn more than once (allowing the words in the same urn to have different emission probabilities). Therefore, $P(w|t)$ is the probability of extracting the word w from the urn that contains all the words that have been seen with the tag t (in this case, $\sum_w P(w|t) = 1$).

Other taggers also use the second-order hypothesis for emission probabilities (Thede & Harper 1999), by considering each word in the training corpus as an output symbol of the state (t_i, t_j), where t_j is the tag of the word and t_i the tag of the previous word. This technique reports a slight improvement in the handling of unknown words. However, as we will see later, the integration of external dictionaries into this model would be particularly difficult, since each word in a dictionary is attached to individual tags, not to pairs of tags.

2.2 Smoothing

Trigram probabilities generated from a corpus cannot be used directly because of the sparse-data problem, which means that there are insufficient instances for each trigram to reliably estimate the probability. Furthermore, setting a probability to zero because the corresponding trigram never occurred in the corpus has an undesired effect. It causes the probability of a complete sequence to be set to zero if its use is necessary for a new text sequence, thus making it impossible to rank different sequences containing a zero probability.

The smoothing paradigm that delivers the best results is linear interpolation of unigrams, bigrams and trigrams. Therefore, we estimate a trigram probability as follows:

$$P(t_3|t_1,t_2) = \lambda_3\,\hat{P}(t_3|t_1,t_2) + \lambda_2\,\hat{P}(t_3|t_2) + \lambda_1\,\hat{P}(t_3)$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$, so P again represents probability distributions.

We use the context-independent variant of linear interpolation, i.e. the values of the λs do not depend on the particular trigram. Contrary to intuition, this yields better results than the context-dependent variant, because, due again to sparse-data problems, one usually cannot estimate an adequate different set of λs for each trigram.

The values of λ_1, λ_2 and λ_3 are estimated by deleted interpolation. This technique successively removes each trigram from the training corpus and estimates best values for the λs from all other n-grams in the corpus. Given the frequency counts for unigrams, bigrams and trigrams, the weights can be very efficiently determined with a processing time which is linear in relation to the number of different trigrams. The algorithm (Brants 2000) is:

1. Set $\lambda_1 = \lambda_2 = \lambda_3 = 0$
2. For each trigram $t_1\,t_2\,t_3$ with $C(t_1,t_2,t_3) > 0$, locate the maximum of the following three values and perform the corresponding action:
   - $\frac{C(t_1,t_2,t_3)-1}{C(t_1,t_2)-1}$: increment λ_3 by $C(t_1,t_2,t_3)$
   - $\frac{C(t_2,t_3)-1}{C(t_2)-1}$: increment λ_2 by $C(t_1,t_2,t_3)$
   - $\frac{C(t_3)-1}{N-1}$: increment λ_1 by $C(t_1,t_2,t_3)$
3. Normalize λ_1, λ_2 and λ_3

If the denominator in one of the expressions is 0, we define the result of that expression to be 0. Note that subtracting 1 means taking unseen data into account. Without this subtraction the model would overfit the training data, generating $\lambda_3 = 1$ and $\lambda_1 = \lambda_2 = 0$, and would yield worse results.

2.3 Handling of unknown words

Currently, the method of handling unknown words that seems to work best for inflected languages is suffix analysis. Given an unknown word, its candidate tags and their probabilities are set according to the ending of the word in question.

The probability distribution of a particular suffix is generated from all words in the training set that share the same suffix of some predefined maximum length. The term suffix as used here means "final sequence of characters of a word" which is not necessarily a linguistically meaningful suffix.

Probabilities are smoothed by successive abstraction (Samuelsson 1993). This calculates the probability of a tag t given the last m letters l_i of an n letter word: $P(t|l_{n-m+1},\ldots,l_n)$. The sequence of increasingly more general contexts omits more and more characters of the suffix, so that $P(t|l_{n-m+2},\ldots,l_n)$, $P(t|l_{n-m+3},\ldots,l_n)$, $\ldots$, $P(t)$ are used for smoothing. The recursion formula is

$$P(t|l_{n-i+1},\ldots,l_n) = \frac{\hat{P}(t|l_{n-i+1},\ldots,l_n) + \theta_i\, P(t|l_{n-i+2},\ldots,l_n)}{1+\theta_i}$$

for $i = m,\ldots,1$, using the maximum likelihood for a suffix of length i derived from corpus frequencies by

$$\hat{P}(t|l_{n-i+1},\ldots,l_n) = \frac{C(t,l_{n-i+1},\ldots,l_n)}{C(l_{n-i+1},\ldots,l_n)}$$

also using weights θ_i, and the initialization $P(t) = \hat{P}(t)$. Of course, as parameters for the Markov model, we need the inverse conditional probabilities $P(l_{n-i+1},\ldots,l_n|t)$ which are obtained by simple Bayesian inversion.

As it can be seen, the definition of this method is not entirely exact and leaves room for interpretation:

- One has to identify an appropriate value for m, the longest suffix used. We use the longest suffix that we can find in the training set, but at most 10 characters.

- Following (Brants 2000), we use a context-independent approach for θ_i (as we did for the contextual weights λ_i), i.e. we set all θ_i to the standard deviation of the unconditioned maximum likelihood probabilities of the tags in the training corpus:

  $$\theta_i = \frac{1}{s-1}\sum_{j=1}^{s}(\hat{P}(t^j) - \bar{P})^2$$

  for all $i = m,\ldots,1$, where s is the cardinal of the tagset and $\bar{P}$ the average

  $$\bar{P} = \frac{1}{s}\sum_{j=1}^{s}\hat{P}(t^j)$$

- Another freedom concerns the choice of the words in the lexicon that should be used for suffix handling. Accepting that unknown words are most probably infrequent, one can argue that using suffixes of infrequent words in the lexicon is a better approximation for unknown words than using suffixes of frequent words. Therefore, we restrict the procedure of suffix handling to words with a frequency lower than or equal to some threshold value (empirically, 10 seems to be a good choice).

In conclusion, this method obtains a proper probability distribution for each tag and for each suffix length, as needed by the hidden Markov model. The guesser can be implemented as a tree of letters containing the final characters of the words. Then, an unknown word is processed letter by letter from right to left going as deep in the tree as possible. The node in which the process stops will determine the list of candidate tags and their probabilities.

Other approaches to guessing combine the probabilities of the words in the lexicon with the probabilities of unknown words, i.e. given a tag t, they allow $\sum_w P(w|t) = S < 1$, where $1-S$ would be the amount of probability devoted to unknown words, if t is known. The estimation of an appropriate value for S is precisely the weakest point of these methods. Fortunately, in our tagger, the sets of probabilities for suffix are separated from the emission probabilities of the words in the lexicon, which also constitutes an advantage when integrating external dictionaries, as we will see in the following section.

3 Integrating external dictionaries

A problem similar to the sparse-data phenomenon for states arises for words as well. Initially, we could assume that the emission probabilities are the only thing that can be perfectly estimated from a tagged corpus. However, in practice, we can find words that are not present in training texts, but that appear in a dictionary[1], i.e. they are words for which the corresponding tags are known, but they do not have emission probabilities, since they have not been seen during training. Once more, it is not convenient to leave these probabilities as zero.

Therefore, it is necessary to use methods to integrate the information provided by the external dictionary within the stochastic tagging framework. To perform this task, instead of the more

[1] In this context, a dictionary is simply a list of words, each word being attached to the enumeration of its possible tags. Contrary to what could be expected, no semantic information is stored in this kind of dictionaries.

general Adding One method, we propose the use of the Good-Turing formulas, as we justify below.

3.1 Adding One method

The most intuitive way to perform integration of dictionaries is the Adding One method, which consists of using the dictionary as an additional tagged corpus where a frequency of 1 is assigned to each word-tag pair. Of course, this new corpus is used to estimate only emission probabilities of words, but not n-grams probabilities, since the order of words and tags in the dictionary has nothing to do with the order of words and tags in real sentences.

Then, for each pair (w_i, t_j) in the dictionary, the emission probability is estimated by

$$\hat{P}(w_i|t_j) = \frac{C(w_i, t_j) + 1}{C(t_j) + K_j}$$

where K_j is the number of words tagged with t_j in the dictionary. For the rest of the word-tag pairs appearing in the training corpus but not in the dictionary, the estimation is performed with the same formula but without adding 1 in the nominator. Intuitively, taking the urn-and-ball system again, this method operates as if all the "balls" or words in the dictionary were put in their corresponding "urns" or tags only once, previously to the estimation of the emission probabilities. With this, the desired effect of having output probabilities for the words in the dictionary other than zero is achieved, even if they do not appear in the training corpus.

At this point, it is important to remember that the success of a tagger on a new application text will be greater if its style is close to the style of the training corpus. The set of words in a general dictionary must be still considered, because it can increase the tagger's coverage. But the style of the training text is defined by the words that appear in it, and these words should have more importance than other external words.

With the Adding One integration, given two words with the same tag, w_i and w_j, the first one present only in the dictionary, and the second one present only once in the training corpus, we find that both have the same frequency and then the same emission probability, which does not produce a coherent representation of the model we are estimating. That is, an external set of words always produce an important alteration in the model parameters, especially when this set is great. This leads us to consider another kind of methods.

3.2 Good-Turing formulas

Every word-tag pair present only in the external dictionary can be seen as a unobserved event, that is, a rare event with frequency 0 (*unobserved zero* event), while a non-possible event is any word-tag pair that is present neither in the dictionary nor in the training corpus (*real zero* event)[2]. The Good-Turing formulas constitute an estimation method that is not completely based on relative frequencies, but which is able to assign probabilities other than 0 to the unobserved events.

In addition, in cases like the above-mentioned one, this technique guarantees that the emission probability for w_i will always be less than that for w_j, and this produces less distortion of the model.

For a collection of events x in the training corpus, we are interested in determining our estimates $\hat{P}(x)$ of the probabilities $P(x)$ with the following structure (Jelinek 1997):

$$\hat{P}(x) = \begin{cases} q_i & \text{for all } x \text{ for which } C(x) = i, \\ & \text{and } i = 0, 1, \ldots, M, \\ \alpha f(x) & \text{for all } x \text{ for which } C(x) > M, \\ & \text{where } f(x) = \frac{C(x)}{N}, \end{cases}$$

where N is the size of the training set. The basic intuition under this structure is that all events observed the same number of times $i \in \{0, 1, \ldots, M\}$ should have the same probability. In the second case, i.e. when $C(x)$ is greater than a certain threshold M, we rely on the corresponding smoothing of the relative frequencies.

The Good-Turing formulas, which can be derived by a variety of methods (Nadas 1991; Ney *et al.* 1995), uses the training set to find the optimal values of the parameters q_i, α and M. Being n_i the number of different symbols x for which $C(x) = i$, we have:

$$q_i = \frac{n_{i+1}}{n_i} \frac{i+1}{N}, \qquad i = 0, 1, \ldots, M \qquad (1)$$

We obtain the value of α by normalization $\sum \hat{P}(x) = 1$, that is setting

$$\sum_{i=0}^{M} q_i n_i + \alpha \sum_{i>M} \frac{i}{N} n_i = 1$$

[2] For example, unknown words are considered real zeros, and, as we have seen before, these null probabilities are solved by the guesser, which obtains the corresponding emission probabilities related to morphology of words.

Integration Method	Training Corpus Size (in sentences)				
	1,000	2,000	3,000	4,000	5,000
no external dictionary	94.14	95.91	96.67	97.22	97.56
Adding One	94.76 (+0.62)	96.26 (+0.35)	96.90 (+0.23)	97.36 (+0.14)	97.66 (+0.10)
Good-Turing	95.26 (+1.12)	96.57 (+0.66)	97.03 (+0.36)	97.44 (+0.22)	97.74 (+0.18)

Table 1: Adding One vs. Good-Turing with the GALENA dictionary on the ITU corpus

Using (1) we obtain

$$\alpha = \frac{\sum\limits_{i>M+1} i\, n_i}{\sum\limits_{i>M} i\, n_i} \tag{2}$$

We have thus in (1) and (2) the well-known Good-Turing formulas, and the natural monotonicity constraint $q_{i-1} < q_i$ now requires the choice of M to satisfy

$$(n_i)^2 < \frac{i+1}{i} n_{i-1}\, n_{i+1}, \qquad i = 1, 2, \ldots, M,$$

and

$$\frac{n_{M+1}}{n_M} < \frac{\sum\limits_{i>M+1} i\, n_i}{\sum\limits_{i>M} i\, n_i}$$

It should be noted that although the occurrence numbers n_i for $i = 1, 2, \ldots, M+1$ are those actually observed in the training data, n_0 is different. It is an inferred number equal to the size of the total lexicon minus the size of the sublexicon actually observed in the training corpus.

It is also interesting to note from (1) that the total probability mass $q_0\, n_0$ assigned by the formulas to all the unobserved events is equal to n_1/N, i.e. the total probability that would have been assigned to singleton events by a relative frequency formula.

With the use of the Good-Turing integration, we have observed improvements in performance. However, it is important to remember that, in our case, this kind of integration has been performed considering isolated sets of words over each different tag (Graña 2000), and only when $n_0 \gg n_1 \gg n_2$, that is, when there are many unobserved events and the data is really sparse (typically, for the most populated categories in the external dictionary: nouns, adjectives, verbs, ...). When this situation does not apply (typically, for the least populated categories in the external dictionary: articles, prepositions, conjunctions, ...), the method is not always applicable and there is the danger of using it incorrectly. In those cases, the Adding One integration should be used.

4 Evaluation

The evaluation experiments have been performed with:

- The ITU[3] corpus, a free available Spanish corpus which has 14,919 sentences and 486,073 tokens. The dictionary formed by the words that appear in the corpus has 17,138 different words, with 18,917 possible taggings.

- The GALENA[4] dictionary, a Spanish dictionary which has 291,604 words with 354,007 possible taggings.

The intersection between both dictionaries involves 6,594 words, which represent 78.73% of the tokens that appear in the ITU corpus. In this case, the contribution of the external dictionary is clear, but its integration must be performed carefully because its great size in comparison with that of the dictionary extracted from the training corpus could lead to important distortions in the model.

The steps of the strategy to perform the experiments are: to build a training corpus formed by sentences randomly taken from the ITU corpus, to train the tagger, to retag the remaining portion of the corpus, and to compare it with the original one. Obviously, this must be done with different sizes of training corpus (we have chosen five different sizes from 1,000 to 5,000 sentences), and in all the possible situations regarding the external dictionary (with no external dictionary, with the GALENA dictionary integrated by the Adding One method, and with the same dictionary integrated by the Good-Turing method).

[3] *International Telecommunications Union CCITT Handbook* (CRATER 1993).

[4] *Generation of Natural Language Analyzers.* See `http://coleweb.dc.fi.udc.es` for more information about this project.

Table 1 shows the percentages of words correctly tagged, and the improvement produced by the GALENA dictionary (when it is integrated by the methods under consideration). We observed much better performances in the Good-Turing integration, especially when the training corpus is small.

The only negative aspect, which is due to the greater complexity in the calculus of the Good-Turing method, is a slight increase in training times. Nevertheless, tagging times remain unchanged.

5 Conclusion

Our contribution, the integration of external dictionaries into a stochastic tagger by the use of the Good-Turing formulas, instead of the Adding One method, produces improvements in performance, especially when dictionaries are great and training texts are small. This kind of situation defines the state-of-the-art in automatic processing of Spanish. As an important conclusion, this technique will guarantee a good performance not only in the automatic processing of this language, but also that of others for which reference texts hardly exist.

References

Baum, L.E. (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, vol. 3, pp. 1-8.

Brants, T. (2000). TNT - A statistical part-of-speech tagger. In *Proceedings of the Sixth Applied Natural Language Processing Conference (ANLP-2000)*, Seattle, WA.

Chanod, J.-P.; Tapanainen, P. (1995). Tagging French - comparing a statistical and a constraint-based method. In *Proceedings of the 7th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 149-156.

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

Church, K.W.; Gale, W.A. (1991). A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, vol. 5, pp. 19-54.

CRATER project. (1993). Corpus Resources And Terminology ExtRaction (MLAP-93/20), creation of a set of tools and resources for multilingual corpus linguistics work. Lancaster University, UK; Computers, Communications and Visions, France; Universidad Autónoma de Madrid, Spain. `http://www.lllf.uam.es/~fernando/projects/es_corpus.html`

Dempster, A.P.; Laird, N.M.; Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc.*, vol. 39(1), pp. 1-38.

Graña Gil, J. (2000). Robust parsing techniques for natural language tagging (in Spanish). *PhD. Thesis*, Departamento de Computación, Universidad de La Coruña (Spain).

Jelinek, F. (1997). Statistical methods for speech recognition. *The MIT Press*, Cambridge, MA.

Nadas, A. (1991). Good, Jelinek, Mercer and Robbins on Turing's estimate of probabilities. *American Journal of Mathematical and Management Sciences*, vol. 11(3-4), pp. 229-308.

Ney, H.; Essen, U.; Kneser, R. (1995). On the estimation of "small" probabilities by leaving-one-out. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17(12) (December), pp. 1202-1212.

Rabiner, L.; Juang, B.H. (1993). Fundamentals of speech recognition. *Prentice-Hall*, Englewood Cliffs, NJ.

Samuelsson, C. (1993). Morphological tagging based entirely on Bayesian inference. In *Proceedings of the 9th Nordic Conference on Computational Linguistics*, Stockholm University, Stockholm, Sweden.

Samuelsson, C.; Voutilainen, A. (1997). Comparing a linguistic and a stochastic tagger. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics/8th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 246-253.

Thede, S.M.; Harper, M.P. (1999). A second-order hidden Markov model for part-of-speech tagging. In *Proceedings of the 37th Meeting of the Association for Computational Linguistics*, pp. 175-182.