Part-of-Speech Tagging with Two Sequential Transducers

André Kempe
Xerox Research Centre Europe

Abstract

The article presents a method of constructing and applying a cascade consisting of a left- and a right-sequential finite-state transducer, \( T_1 \) and \( T_2 \), for part-of-speech disambiguation. In the process of POS tagging, every word is first assigned a unique ambiguity class that represents the set of alternative tags that this word can occur with. The sequence of the ambiguity classes of all words of one sentence is then mapped by \( T_1 \) to a sequence of reduced ambiguity classes where some of the less likely tags are removed. That sequence is finally mapped by \( T_2 \) to a sequence of single tags. Compared to a Hidden Markov model tagger, this transducer cascade has the advantage of significantly higher processing speed, but at the cost of slightly lower accuracy. Applications such as Information Retrieval, where the speed can be more important than accuracy, could benefit from this approach.

1 Introduction

We present a method of constructing and applying a cascade consisting of a left- and a right-sequential finite-state transducer (FST), \( T_1 \) and \( T_2 \), for part-of-speech (POS) disambiguation.

In the process of POS tagging, we first assign every word of a sentence a unique ambiguity class \( c_i \) that can be looked up in a lexicon encoded by a sequential FST. Every \( c_i \) is denoted by a single symbol, e.g. “[ADJ NOUN]”, although it represents a set of alternative tags that a given word can occur with. The sequence of the \( c_i \) of all words of one sentence is the input to our FST cascade (Fig. 1). It is mapped by \( T_1 \), from left to right, to a sequence of reduced ambiguity classes \( r_i \). Every \( r_i \) is denoted by a single symbol, although it represents a set of alternative tags. Intuitively, \( T_1 \) eliminates the less likely tags from \( c_i \), thus creating \( r_i \). Finally, \( T_2 \) maps the sequence of \( r_i \), from right to left, to an output sequence of single POS tags \( t_i \). Intuitively, \( T_2 \) selects the most likely \( t_i \) from every \( r_i \) (Fig. 1).

Compared to a Hidden Markov model (HMM) (Rabiner 1990), this FST cascade has the advantage of significantly higher processing speed, but at the cost of slightly lower accuracy. Applications such as Information Retrieval, where the speed can be more important than accuracy, could benefit from this approach.

Although our approach is related to the concept of bimachines (Schützenberger 1961) and factorization (Elgot and Mezei 1965), we proceed differently in that we build two sequential FSTs directly and not by factorization.

This article is structured as follows. Section 2 describes how the ambiguity classes and reduced ambiguity classes are defined based on a lexicon and a training corpus. Then, Section 3 explains how the probabilities of these classes in the context of other classes are calculated. The construction of \( T_1 \) and \( T_2 \) is shown in Section 4. It makes use of the previously defined classes and their probabilities.
Section 5 describes the application of the FSTs to an input text, and Section 6 finally compares the FSTs to an HMM tagger, based on experimental data.

2 Definition of Classes

Instead of dealing with lexical probabilities of individual words (Church 1988), many POS taggers group words into ambiguity classes and deal with lexical probabilities of these classes (Cutting, Kupiec, Pedersen and Sibun 1992, Kupiec 1992). Every word belongs to one ambiguity class that is described by the set of all POS tags that the word can occur with. For example, the class described by \{NOUN, VERB\} includes all words that could be analyzed either as noun or verb depending on the context. We follow this approach.

Some approaches make a more fine-grained word classification (Daelemans, Zavrel, Berck and Gillis 1996, Tzoukermann and Radev 1996). Words that occur with the same alternative tags, e.g., NOUN and VERB, can here be assigned different ambiguity classes depending on whether they occur more frequently with one or with the other tag. Although this has proven to increase the accuracy of HMM-based POS disambiguation, it did not significantly improve our method. After some investigations in this direction, we decided to follow the simpler classification above.

Before we can build the FST cascade, we have to define ambiguity classes, that will constitute the input alphabet of \( T_1 \), and reduced ambiguity classes, that will form the intermediate alphabet of the cascade, i.e., the output of \( T_1 \) and the input of \( T_2 \).

Ambiguity classes \( c_i \) are defined from the training corpus and lexicon, and are
Part-of-Speech Tagging with Two Sequential Transducers

Each described by a pair consisting of a tag list $\hat{t}(c_i)$ and a probability vector $\bar{p}(c_i)$:

$$\hat{t}(c_i) = \langle t_{i1}, t_{i2}, \ldots, t_{in} \rangle \quad \bar{p}(c_i) = \begin{bmatrix} p(t_{i1}|c_i) \\ p(t_{i2}|c_i) \\ \vdots \\ p(t_{in}|c_i) \end{bmatrix}$$ (1)

For example:

$$\hat{t}(c_1) = \langle \text{ADJ, NOUN, VERB} \rangle \quad \bar{p}(c_1) = \begin{bmatrix} 0.29 \\ 0.60 \\ 0.11 \end{bmatrix}$$ (2)

which means that the words that belong to $c_1$ are tagged as ADJ in 29%, as NOUN in 60%, and as VERB in 11% of all cases in the training corpus.

When all $c_i$ are defined, a class-based lexicon, that maps every word to a single class symbol, is constructed from the original tag-based lexicon, that maps every word to a set of alternative tag symbols. In the class-based lexicon, the above $c_1$ (Eq. 2) could be represented, e.g., by the symbol "[ADJ NOUN VERB].

We describe a reduced ambiguity classes $r_i$ also by a pair consisting of a tag list $\hat{t}(r_i)$ and a probability vector $\bar{p}(r_i)$. Intuitively, an $r_i$ can be seen as a $c_i$ where some of the less likely tags have been removed. Since at this point we cannot decide which tags are less likely, all possible subclasses of all $c_i$ are considered. To generate a complete set of $r_i$, all $c_i$ are split into all possible subclasses $s_{ij}$ that are assigned a tag list $\hat{t}(s_{ij})$ containing a subset of the tags of $\hat{t}(c_i)$, and an (un-normalized) probability vector $\bar{p}(s_{ij})$ containing only the relevant elements of $\bar{p}(c_i)$. For example, the above $c_1$ (Eq. 2) is split into seven subclasses $s_{1j}$:

$$\hat{t}(s_{1,0}) = \langle \text{ADJ, NOUN, VERB} \rangle \quad \bar{p}(s_{1,0}) = \begin{bmatrix} 0.29 \\ 0.60 \\ 0.11 \end{bmatrix}$$

$$\hat{t}(s_{1,1}) = \langle \text{NOUN, VERB} \rangle \quad \bar{p}(s_{1,1}) = \begin{bmatrix} 0.60 \\ 0.11 \end{bmatrix}$$ (3)

$$\hat{t}(s_{1,2}) = \langle \text{ADJ, VERB} \rangle \quad \bar{p}(s_{1,2}) = \begin{bmatrix} 0.29 \\ 0.11 \end{bmatrix}$$

etc.

Different $c_i$ can produce a $s_{ij}$ with the same tag list $\hat{t}(s_{ij})$ but with different probability vectors $\bar{p}(s_{ij})$: e.g., the classes with the tag lists $\langle \text{ADJ, NOUN, VERB} \rangle$, $\langle \text{NOUN, VERB} \rangle$, and $\langle \text{ADJ, ADV, NOUN, VERB} \rangle$ can all produce a subclass with the tag list $\langle \text{NOUN, VERB} \rangle$. To reduce the total number of subclasses, all $s_{ij}$ with the same tag list $\hat{t}(s_{ij})$ are clustered, based on the centroid method (Romesburg 1989, p. 136), using the vector cosine as the similarity measure between clusters (Salton and McGill 1983, p. 201). Each final cluster constitutes a reduced ambiguity class
If we obtain, e.g., three \( r_y \) with the same tag list \( \tilde{t}(r_y) = \langle \text{NOUN, VERB} \rangle \) but with different (re-normalized) probability vectors:

\[
\begin{align*}
\tilde{p}(r_1) &= \begin{bmatrix} 0.89 \\ 0.11 \end{bmatrix} \\
\tilde{p}(r_2) &= \begin{bmatrix} 0.57 \\ 0.43 \end{bmatrix} \\
\tilde{p}(r_3) &= \begin{bmatrix} 0.09 \\ 0.91 \end{bmatrix}
\end{align*}
\]  

(4)

we represent them in an FST by three different symbols, e.g., “[NOUN VERB]\_1”, “[NOUN VERB]\_2”, and “[NOUN VERB]\_3”.

3 Contextual Probabilities

\( T_1 \) will map a sequence of \( c_i \), from left to right, to a sequence of \( r_i \). Therefore, the construction of \( T_1 \) requires estimating the most likely \( r_i \) in the context of both the current \( c_i \) and the previous \( r_{i-1} \) (wrt. the current position \( i \) in a sequence). To determine this \( r_i \), a probability \( P_{T_1}(t_{ij}) \) is estimated for every POS tag \( t_{ij} \) in \( c_i \).

In the initial position, \( P_{T_1}(t_{ij}) \) depends on the preceding sentence boundary \( \#_{i-1} \) and the current \( c_i \) which are assumed to be mutually independent:

\[
P_{T_1}(t_{ij}) = p(t_{ij}|\#_{i-1} c_i)
\]

\[
= \frac{p(t_{ij}|\#_{i-1} c_i)}{p(\#_{i-1} c_i)}
\]

\[
= \frac{p(\#_{i-1} c_i|t_{ij}) \cdot p(t_{ij})}{p(\#_{i-1} c_i)}
\]

\[
\approx \frac{p(\#_{i-1} c_i|t_{ij}) \cdot p(t_{ij})}{p(\#_{i-1}) \cdot p(c_i)}
\]

\[
= \frac{p(\#_{i-1} t_{ij}) \cdot p(c_i|t_{ij}) \cdot p(t_{ij})}{p(\#_{i-1}) \cdot p(c_i)}
\]

\[
= \frac{p(\#_{i-1} t_{ij}) \cdot p(t_{ij} c_i)}{p(t_{ij}) \cdot p(\#_{i-1} \cdot p(c_i)}
\]

\[
= \frac{p(t_{ij}|\#_{i-1}) \cdot p(t_{ij}|c_i)}{p(t_{ij})}
\]

\]

(5)

The latter \( p(t_{ij}|c_i) \) can be extracted from the probability vector \( \tilde{p}(c_i) \), and \( p(t_{ij}|\#_{i-1}) \) and \( p(t_{ij}) \) can be estimated from the training corpus.

In another than the initial position, \( P_{T_1}(t_{ij}) \) depends on the preceding \( r_{i-1} \) and the current \( c_i \) which are assumed to be mutually independent:

\[
P_{T_1}(t_{ij}) = p(t_{ij}|r_{i-1} c_i) \approx \frac{p(t_{ij}|r_{i-1}) \cdot p(t_{ij}|c_i)}{p(t_{ij})}
\]  

(6)
The latter $p(t_{ij} | r_{i-1})$ is estimated by:

$$
p(t_{ij} | r_{i-1}) = \sum_k p(t_{ij} | t_{i-1,k}) \cdot p(t_{i-1,k} | r_{i-1})$$

(7)

with $t_{ij} \in \tilde{t}(c_i) ; \ t_{i-1,k} \in \tilde{t}(r_{i-1})$

where $p(t_{ij} | t_{i-1,k})$ can be estimated from the training corpus, and $p(t_{i-1,k} | r_{i-1})$ can be extracted from the probability vector $\hat{p}(r_{i-1})$ of the preceding $r_{i-1}$.

To evaluate all tags of the current $c_i$, a list $\mathcal{P}(c_i)$ containing pairs $\langle t_{ij}, P_{t_i}(t_{ij}) \rangle$ of all tags $t_{ij}$ of $c_i$ with their probabilities $P_{t_i}(t_{ij})$ (Eq.s 5, 6), is created:

$$\mathcal{P}(c_i) = \left( \begin{array}{l}
\langle t_{i,1}, P_{t_i}(t_{i,1}) \rangle \\
\langle t_{i,2}, P_{t_i}(t_{i,2}) \rangle \\
\vdots \\
\langle t_{i,j}, P_{t_i}(t_{i,j}) \rangle \\
\end{array} \right)$$

(8)

Every tag $t_{ij}$ in $\mathcal{P}$ is compared to the most likely tag $t_{i,m}$ in $\mathcal{P}$. If the ratio of their probabilities is below a threshold $\tau$, $t_{ij}$ is removed from $\mathcal{P}$:

$$\frac{P_{t_i}(t_{ij})}{P_{t_i}(t_{i,m})} < \tau$$

(9)

Removing less likely tags leads to a reduced list $\mathcal{P}_r(c_i)$ that is then split into a reduced tag list $t_r(c_i)$ and a reduced probability vector $\hat{p}_r(c_i)$ that jointly describe a reduced ambiguity class $r_y$. From among all predefined $r_i$ (cf. e.g. Eq. 4), we select the one that has the same tag list $t(r_i)$ as the “ideal” reduced class $r_y$ and the most similar probability vector $\hat{p}(r_i)$ according to the cosine measure. This $r_i$ is considered to be the most likely among all predefined $r_i$ in the context of both the current $c_i$ and the previous $r_{i-1}$.

$T_2$ will map a sequence of $r_i$, from right to left, to a sequence of tags $t_i$. Therefore, the construction of $T_2$ requires estimating the most likely $t_i$ in the context of both the current $r_i$ and the following $t_{i+1}$. To determine this $t_i$, a probability $P_{t_2}(t_{ij})$ is estimated for every tag $t_{ij}$ of the current $r_i$. In the final position, $P_{t_2}(t_{ij})$ depends on the current $r_i$ and on the following sentence boundary $\#_{i+1}$:

$$P_{t_2}(t_{ij}) = p(t_{ij} | r_i \#_{i+1}) \approx \frac{p(t_{ij} | \#_{i+1}) \cdot p(t_{ij} | r_i)}{p(t_{ij})}$$

(10)

In another than the final position, $P_{t_2}(t_{ij})$ depends on the current $r_i$ and the following tag $t_{i+1}$:

$$P_{t_2}(t_{ij}) = p(t_{ij} | r_i t_{i+1}) \approx \frac{p(t_{ij} | t_{i+1}) \cdot p(t_{ij} | r_i)}{p(t_{ij})}$$

(11)
The latter \( p(t_{ij}) \), \( p(t_{ij}|\#_{i+1}) \), and \( p(t_{ij}|r_{i}) \) are estimated from the training corpus, and \( p(t_{ij}|r_{i}) \) is extracted from the probability vector \( \tilde{p}(r_{i}) \).

The \( t_{i} \) with the highest probability \( P_{T_{1}}(t_{i}) \) is the most likely tag in the context of both the current \( r_{i} \) and the following \( t_{i+1} \) (Eq.s 10, 11).

### 4 Construction of the FSTs

The construction of \( T_{1} \) is preceded by defining all \( c_{i} \) and \( r_{i} \), and estimating their contextual probabilities. In this process, all words in the training corpus, that are initially annotated with POS tags, are in addition annotated with ambiguity classes \( c_{i} \).

In \( T_{1} \), one state is created for every \( r_{i} \) (output symbol), and is labeled with this \( r_{i} \) (Fig. 2a). An initial state, not corresponding to any \( r_{i} \), is created in addition. From every state, one outgoing arc is created for every \( c_{i} \) (input symbol), and is labeled with this \( c_{i} \). The destination of every arc is the state of the most likely \( r_{i} \) in the context of both the current \( c_{i} \) (arc label) and the preceding \( r_{i-1} \) (source state label) which is estimated as described above. All arc labels are then changed from simple symbols \( c_{i} \) to symbol pairs \( c_{i}\;r_{i} \) (mapping \( c_{i} \) to \( r_{i} \) that consist of the original arc label and the destination state label. All state labels are removed (Fig. 2b). Those \( r_{i} \) that are unlikely in any context disappear from \( T_{1} \) because the corresponding states have no incoming arcs. \( T_{1} \) accepts any sequence of \( c_{i} \) and maps it, from left to right, to the sequence of the most likely \( r_{i} \) in the given left context.

![Figure 2: Two stages in the construction of \( T_{1} \)](a)

The construction of \( T_{2} \) is preceded by annotating the training corpus in addition with reduced ambiguity classes \( r_{i} \), by means of \( T_{1} \). The probability vectors \( \tilde{p}(r_{i}) \) of all \( r_{i} \) are then re-estimated. The contextual probabilities of tags, are estimated only at this point (Eq.s 10, 11).

In \( T_{2} \), one state is created for every \( t_{i} \) (output symbol), and is labeled with this \( t_{i} \) (Fig. 3a). An initial state is added. From every state, one outgoing arc is created for every \( r_{i} \) (input symbol) that occurs in the output language of \( T_{1} \), and is labeled with this \( r_{i} \). The destination of every arc is the state of the most likely \( t_{i} \) in the context of both the current \( r_{i} \) (arc label) and the following \( t_{i+1} \) (source state label) which is estimated as described above. Note, this is the following tag, rather than the preceding, because \( T_{2} \) will be applied from right to left. All arc labels are then changed into symbol pairs \( r_{i}\;t_{i} \) and all state labels are removed (Fig. 3b), as was done in \( T_{1} \). \( T_{2} \) accepts any sequence of \( r_{i} \), generated by \( T_{1} \), and maps it, from
right to left, to the sequence of the most likely \( t_i \) in the given right context.

Both \( T_1 \) and \( T_2 \) are sequential. They can be minimized with standard algorithms. Once \( T_1 \) and \( T_2 \) are built, the probabilities of all \( t_i, r_i, \) and \( c_i \) are of no further use. Probabilities do not explicitly occur in the FSTs, and are not directly used at run time. They are, however, “reflected” by the structure of the FSTs.

5 Application of the FSTs

Our FST tagger uses the above described \( T_1 \) and \( T_2 \), a class-based lexicon, and possibly a guesser to predict the ambiguity classes of unknown words (possibly based on their suffixes). The lexicon and guesser are also sequential FSTs, and map any word that they accept to a single symbol \( c_i \) representing an ambiguity class (Fig. 1). If a word cannot be found in the lexicon, it is analyzed by the guesser. If this does not provide an analysis either, the word is assigned a special \( c_i \) for unknown words that is estimated from the \( m \) most frequent tags of all words that occur only once in the training corpus.

The sequence of the \( c_i \) of all words of one sentence is the input to our FST cascade (Fig. 1). It is mapped by \( T_1 \), from left to right, to a sequence of reduced ambiguity classes \( r_i \). Intuitively, \( T_1 \) eliminates the less likely tags from \( c_i \), thus creating \( r_i \). Finally, \( T_2 \) maps the sequence of \( r_i \), from right to left, to an output sequence of single POS tags \( t_i \). Intuitively, \( T_2 \) selects the most likely \( t_i \) from every \( r_i \).

6 Results

We compared our FST tagger on English, German, and Spanish with a commercially available (foreign) HMM tagger (Table 1). The comparison was made on the same non-overlapping training and test corpora for both taggers (Table 3). The FST tagger was on average 10 times as fast but slightly less accurate than the HMM tagger (45 600 words/sec and 96.97% versus 4 360 words/sec and 97.43%). In some applications such as Information Retrieval a significant speed increase could be worth the small loss in accuracy.
|                | English | German | Spanish | Average  |
|----------------|---------|--------|---------|----------|
| Speed (words/sec) | $T_1+T_2$ | 47 600 | 42 200 | 46 900 | 45 600 |
|                | HMM     | 4 110  | 3 620  | 5 360  | 4 360  |
| Accuracy (%)   | $T_1+T_2$ | 96.54 | 96.79 | 97.05 | 96.97 |
|                | HMM     | 96.80  | 97.55  | 97.95  | 97.43  |

*Computer: SUN Workstation, Ultra2, with 1 CPU*

Table 1: Processing speed and accuracy of the FST and the HMM taggers

|                | English | German | Spanish | Average  |
|----------------|---------|--------|---------|----------|
| # States       |         | 615    | 496     | 353      | 488      |
| # Arcs         |         | 209 000| 197 000 | 96 000   | 167 000  |
| # Tags         |         | 76     | 67      | 56       | 66       |
| # Ambiguity classes |       | 349    | 448     | 265      | 354      |
| # Reduced ambiguity classes | | 724    | 732     | 465      | 640      |

Table 2: Sizes of the FST cascades and their alphabets

|                | English | German | Spanish | Average  |
|----------------|---------|--------|---------|----------|
| Training corpus size (words) | 20 000 | 91 000 | 16 000 | 42 000 |
| Test corpus size (words)      | 20 000 | 40 000 | 15 000 | 25 000 |

Table 3: Sizes of the training and test corpora

**References**

Church, K. W. (1988), A stochastic parts program and noun phrase parser for unrestricted text, *Proceedings of the 2nd Conference on Applied Natural Language Processing (ANLP)*, Association for Computational Linguistics, Austin, TX, USA, pp. 136–143.

Cutting, D., Kupiec, J., Pedersen, J. and Sibun, P. (1992), A practical part-of-speech tagger, *Proceedings of the 3rd Conference on Applied Natural Language Processing (ANLP)*, Association for Computational Linguistics, Trento, Italy, pp. 133–140.

Daelemans, W., Zavrel, J., Berck, P. and Gillis, S. (1996), MBT: A memory-based part-of-speech tagger-generator, *Proceedings of the 4th Workshop on Very Large Corpora*, Special Interest Group for Linguistic Data and Corpus-based Approaches (SIGDAT) of the ACL, Copenhagen, Denmark, pp. 14–27.
Elgot, C. C. and Mezei, J. E.(1965), On relations defined by generalized finite automata, *IBM Journal of Research and Development* pp. 47–68.

Kupiec, J. M.(1992), Robust part-of-speech tagging using a hidden markov model, *Computer, Speech, and Language* **6**(3), 225–242.

Rabiner, L. R.(1990), A tutorial on hidden markov models and selected applications in speech recognition, *in* A. Waibel and K.-F. Lee (eds), *Readings in Speech Recognition*, Morgan Kaufmann, pp. 267–296.

Romesburg, H. C.(1989), *Cluster Analysis for Researchers*, Krieger Publishing Company, Malabar, FL, USA.

Salton, G. and McGill, M. J.(1983), *Introduction to Modern Information Retrieval*, McGraw-Hill Advanced Computer Science Series, McGraw-Hill Publishing Company, New York, USA.

Schützenberger, M. P.(1961), A remark on finite transducers, *Information and Control* **4**, 185–187.

Tzoukermann, E. and Radev, D. R.(1996), Using word class for part-of-speech disambiguation, *Proceedings of the 4th Workshop on Very Large Corpora*, Special Interest Group for Linguistic Data and Corpus-based Approaches (SIGDAT) of the ACL, Copenhagen, Denmark, pp. 1–13.