Part-of-Speech Tagging Without Training

Stéphane Bressan, and Lily Suryana Indradjaja

School of Computing, National University of Singapore
{go303015, steph}@nus.edu.sg

Abstract. The development of the Internet and the World Wide Web can be either a threat to the survival of indigenous languages or an opportunity for their development. The choice between cultural diversity and linguistic uniformity is in our hands and the outcome depends on our capability to devise, design and use tools and techniques for the processing of natural languages. Unfortunately natural language processing requires extensive expertise and large collections of reference data. Our research is concerned with the economical and therefore semi-automatic or automatic acquisition of such linguistic information necessary for the development of indigenous or multilingual information systems. In this paper, we propose new methods and variants of existing methods for part-of-speech tagging. We comparatively and empirically analyze the proposed methods and existing reference methods using the Brown English language corpus and we present some preliminary remarks on experiments with an Indonesian language Corpus.

1 Introduction

The development of the Internet and the World Wide Web can be either a threat to the survival of indigenous languages or an opportunity for their development. The choice between cultural diversity and linguistic uniformity is in our hands and the outcome depends on our capability to devise, design and use tools and techniques for the processing of natural languages. Unfortunately natural language processing requires extensive expertise and large collections of reference data. Our research is concerned with the economical and therefore semi-automatic or automatic acquisition of such linguistic information necessary for the development of other-than-English indigenous or multilingual information systems.

In this paper we are concerned with automatic part-of-speech tagging. Part-of-speech tagging is the task of assigning the correct class (part-of-speech) to each word in a sentence. A part-of-speech can be a noun, verb, adjective, adverb etc. It should not be confused with the role/position in a sentence, which includes subject, predicate and object. Different word classes may occupy the same position, and similarly, a part-of-speech can take on different roles in a sentence. Automatic part-of-speech tagging is therefore the assignment of a part-of-speech class (or tag) to terms in a document. In this paper, we present and evaluate several methods for the fully automatic acquisition of the knowledge necessary for part-of-speech tagging.

A. Aagesen et al. (Eds.): INTELLCOMM 2004, LNCS 3283, pp. 112–119, 2004.
© IFIP International Federation for Information Processing 2004

We try and devise a method that relies on the general statistical properties that characterize languages yet that can be learned independently of particular knowledge about a particular language.

In the next section, we discuss the research efforts and results that represent the state of the art in the area of part-of-speech tagging. In section 3 we present the proposed methods and their detailed description. In section 4, we comparatively and empirically analyze the proposed methods and existing reference methods using the Brown English language corpus and we present some preliminary remarks on experiments with an Indonesian language Corpus. We conclude in section 5.

2 Related Work

The definition of part-of-speech and their association to words and units in a sentence or a text is an essential and possibly intrinsic component of a linguistic system. In information retrieval and computation linguistic, the applications of part-of-speech tagging are numerous and ubiquitous. Part-of-speech tagging can be a tool for tasks as various as word-sense disambiguation, indexing, stemming, and, of course, parsing.

The simplest part-of-speech taggers are based on n-gram (bigram or trigram) [5,4]. Taggers which implement n-gram models require a relatively large tagged training corpus. Transformation-based tagging as introduced by Brill in [3] also requires a tagged corpus for training. No pre-tagged corpus is necessary for Hidden Markov Models [12,6,14]. However, a lexicon that specifies the possible parts of speech for every word is still needed.

Several researchers have worked on learning grammatical properties of words. Elman (1990) trained a connectionist net to predict words, a process that generates internal representations that reflect grammatical categories. Brill et al. in [2] tried to infer grammatical category from bigram statistics. Finch and Chater in [9] and Finch in [10] used vector models in which words are clustered according to the similarity of their close neighbors in a corpus. Kneser and Ney in [13] presented a probabilistic model for entropy maximization that also relies on the immediate neighbors of words in a corpus. Biber in [1] applied factor analysis to collocations of two target words ("certain" and "right") with their immediate neighbors.

Schutze in [18] is the first to have presented an algorithm for tagging words whose part-of-speech properties are unknown. He hypothesized that syntactic behavior is reflected in co-occurrence patterns. The degree of similarity between two words is determined by the degree to which they share the same left and right neighbors as reflected by the cosine similarity in a vector space. The context vectors of all the words are then clustered using Buckshot clustering algorithm [7]. Schutze conducted two types of experiments: one that classifies word types, and one that categorizes individual word occurrences, i.e. different occurrences of the same word may be assigned different tags. We will subsequently refer to these two types of classification as *Schutze1* and *Schutze2*, respectively.

3 Proposed Methods

3.1 N-Gram Method and Sentence Splitting

The first method we propose is based on n-grams. It hypothesizes that the more examples of sentences or sequences of words that are identical but for a word in each sentence at the same position, the more these two words are likely to be of the same part-of-speech category. We consider all the sequences of n words (n-grams) that are identical in all but for one word. We then count the number of such occurrences in the whole corpus for each pair of words and form a similarity matrix. For example, if we have two trigrams (w1 w2 w3) and (w1 w4 w3), we increment by one the count in the entry (2, 4) of the matrix. We refer to such occurrence as the association between w2 and w4.

The sentence splitting approach tries and identifies words that frequently serve as pivots. Let us consider for instance two sentences that have three words (w1 w2 w3) and (w4 w2 w5). The method hypothesizes that the more such examples, the more likely w1 and w4, and similarly w3 and w5, are to be of the same part-of-speech category. The idea can be applied recursively by splitting sentences around pivots into sub-sentences until we find individual words. We then count such occurrences over the entire corpus and thus form a similarity matrix accordingly.

3.2 Clustering

After the similarity matrix in each of both the methods described above is formed we group the different words into classes using a clustering algorithm. The choice of clustering algorithm has a significant impact on the quality of the clustering results. In this paper, depending on its applicability to the different methods, we consider one of or both the Markov Clustering (MCL) algorithm and the k-means clustering algorithm. In the MCL algorithm, the words and their associations are viewed as the vertices and edges of a weighted graph, respectively. The algorithm partition the graph into connected regions, i.e. with many edges connecting vertices inside the partition and few connecting vertices inside the region with vertices outside the region. The number of edges connecting two vertices corresponds to the degree of similarity of the two vertices. By simulating flows (random walks) within this structure, it further strengthens the flow where the current (connection) is already strong, and weakens the flow where the current is weak, hence placing the vertices with higher degree of similarity in the same cluster and those with lower degree of similarity in different clusters. The K-means clustering algorithm, on the other hand, views the similarity between two words as the inverse of the distance of the two vectors representing the words. This algorithm first forms the desired number of clusters (k) and then proceeds by assigning components to the cluster to which their distance is the minimum.

3.3 Extended Schutze's Approach

Schutze in [18] measured the similarity between two words with respect to their syntactic behavior by the degree to which they share the same immediate neighbors on the left and right. The counts of neighbors are assembled into a vector, with one dimension for each neighbor. We refer to the vector of left neighbors of a word as its left context vector, and to the vector of right neighbors as its right context vector. For example, if w1 and w2 are the neighbors to be considered, the left and right context vectors of a word w – assuming that it only occurs in the phrase (w1 w w2) – are (1 0) and (0 1) respectively. Each of the context vectors in Schutze's approach consists of 250 entries, corresponding to the 250 most frequent words in the Brown corpus.

Schutze acknowledges in his paper that the decision to consider only immediate neighbors may lead to an error when two words of different classes are categorized into the same class because they share the same left and right neighbors. For example, the adverb "currently" in the sentence "Hester, currently Dean of …" and the conjunction "if" in "to add that, if United States policies …" have similar neighbors (comma, noun phrase). To address this problem, we extend Schutze's algorithm by considering a broader context. In addition to the left and right context vectors characterizing a word w, we form two more context vectors: one for the word preceding its immediate left neighbor one for the word following its immediate right neighbor. We will refer to these two additional context vectors as the secondary left and secondary right context vectors of w. Hence, the degree of similarity between two words is determined by the degree to which they share the same two neighbors on the left and right, respectively.

With each context vector consisting of 250 entries, each word in the extended Schutze's approach is characterized by a vector of 1,000 entries. We form a matrix of size n-by-250 for each type of context vector, where n is the number of words to be clustered. To allow us to work with smaller matrices, we reduce the dimensionality of the matrix by using the Singular Value Decomposition (see for instance [15]). Each of the matrices is reduced to its 50 main dimensions. We then concatenate the four reduced matrices to form a matrix of size n-by-200. The k-means clustering algorithm is used for clustering in the case of Schutze's method and its variants.

4 Experiments and Results

We evaluate the proposed methods using the Brown corpus [11] tagged by the Penn Treebank Project [16]. The Brown corpus consists of 500 texts in English, each consisting of just over 2,000 words. The texts were sampled from 15 different text categories, ranging from press editorials to fictions and humors. The corpus is made up of 52,108 sentences, constituting over 1 million words, with 47,649 of them being distinct. We form 16 classes of tags, with each tag associated with one or more tags from the Penn Treebank. We label each cluster found with the tag that occurs the most frequently in that cluster. We say a word is correctly tagged if it is in a cluster whose associated tag is the same as the one associated to that word in the tagged corpus. For each tag t from the Penn Treebank, we count the occurrences of t in the corpus, the number of correctly tagged words and the number of incorrectly tagged words. We use three performance metrics in the evaluation: the average precision of the 16 tags, the average recall, and the percentage of the number of words correctly clustered, i.e. the sum of the correct cases for all the 16 tags over the size of the corpus. The algorithms are implemented in Java. SVD is implemented using the JAMA package [17]. The k-means clustering algorithm implementation is our own, while the MCL algorithm is from [19].

4.1 Experiments on a Subset of the Brown Corpus

The first set of experiments comparatively evaluates the performance of the combination of the two trigram and sentence-splitting methods with the two clustering algorithms. We have randomly chosen a small subset of the Brown corpus, containing a total of 89,397 words with 12,664 of them being distinct in order to be able to obtain the clustering results in a reasonable time. With the k-means clustering algorithm, we present the best results obtained for k=200 clusters (recall from the previous section that MCL algorithm determines the number of optimal classes, so we do not need to specify the desired number of classes). Table1 summarizes the results of evaluating the sentence splitting and trigram approach on this corpus using both clustering algorithms.

Table 1. The results of clustering the words in a subset of the Brown corpus, using different combinations of sentence-splitting and trigram approaches with MCL and k-means clustering algorithms

Method	Clustering Algorithm	#Clusters	Average Precision	Average Recall	% Correct
Sentence splitting	MCL	498	0.74	0.43	52%
Sentence splitting	K-means	200	0.76	0.51	63%
Trigram	MCL	655	0.82	0.47	57%
Trigram	K-means	200	0.87	0.68	71%

We see that the trigram method consistently outperform the sentence-splitting methods in both recall and precision for both types of clustering. For an optimal value of k (=200) the best results in recall and precision are obtained by the k-mean algorithm for both methods. Yet the reader must remember that MCL does not require that the programmer anticipates the number of clusters. The reader notices that, in this context, the loss of performance of the trigram method with MCL over the trigram method with k-mean is mainly in precision.

4.2 Experiments on the Whole Brown Corpus

On the basis of the results obtained in the previous sub-section, we now compare the trigram and extended Schutze's methods. The two methods use the k-mean clustering. For the trigram method we reduce the matrix using the 300, 500, and 7000 most frequent words, successively. We use k=200. The experiments are run on the complete Brown corpus. The results for Schutze1 and Schutze2 are reported from [18] in which they have been described to be run with the same parameters on the same corpus. Table 2 shows the results obtained with these approaches. The extended Schutze's method, which we have proposed, consistently outperforms all other methods.

Table 2. Results of the trigram and extended Schutze's methods on the whole Brown corpus, using the k-means clustering algorithm. Words are classified into 200 clusters. The number in the brackets with the trigram approach indicates the number of most frequent words used for association

Method	Average Precision	Average Recall	% Correct
Trigram (300)	0.70	0.60	64%
Trigram (500)	0.74	0.62	66%
Trigram (700)	0.76	0.62	67%
Extended Schutze's	**0.90**	**0.72**	**81%**
Schutze1	0.53	0.52	65%
Schutze2	0.78	0.71	80%

The trigram method manages to find 31,838 words that are associated with at least one word in the corpus before the reduction to the 300, 500, and 700 most frequent words respectively. From the three sets of results, and as intuition suggests, we can anticipate that the gradient of improvement decreases as the number of the kept most frequent words increases. Recall from section 3 that the extended Schutze's approach counts the co-occurrence of words using four context vectors, each consisting of 250 entries corresponding to the 250 most frequent words in the corpus. With the extended Schutze's method and the 250 most frequent words use to compose the context vectors, we obtain 47,553 words that co-occur at least once with one of the 250 most frequent words. We observe from Table 3 that this method performs better (average precision/recall of 0.90/0.72) than the trigram approach. This is probably because with the trigram approach we only manage to associate 31,838 words with each other. The rest of the words not associated with any word in the corpus are characterized by a zero vector; hence we practically cluster only 31,838 words with the trigram approach, which is less than 70% of the number of words clustered with the extended Schutze's method. This extended method also performs better than the original Schutze's methods (Schutze1 and Schutze2, average precision/recall of 0.53/0.52 and 0.78/0.71 respectively). Recall from the end of section 2 that the method Schutze2 classifies individual word occurrences, i.e. different occurrences of the same word may be assigned different tags, while the extended Schutze's approach classifies word types only. While it seems that Schutze2 should be able to achieve higher precision in general, it is also possible that it assigns too many different tags to a word.

4.3 Experiments on the Indonesian Corpus

We applied the extended Schutze's method to a corpus of documents in the Indonesian language. Since no pre-tagged corpus is available for this language, we are not able to measure the precision and recall. Nevertheless, manual inspection of the clusters formed can show several interesting properties. In particular it shows that words with the same affixes tend to be in the same cluster. This is particularly interesting since the Indonesian language is a morphologically rich language with a predominant derivational morphology. Examples are given in table3. These observation suggest to use the methods proposed to guide and improve stemming algorithms for morphologically rich algorithms with a predominant derivational morphology or a inflectional morphology that differentiates parts-of-speech (and position in the sentence).

Table 3. Examples of clusters found from the Indonesian corpus with their associated affixes

Tag	Affix	Examples
Verb	me-i	menangani, mengatasi, mengulangi
Verb	di-kan	diperlukan, disiagakan, dipertemukan, disempurnakan
Noun	pe-	pengusaha, pejuang, pejabat, peneliti, pendiri
Noun	ke-an	keselamatan, kepentingan, keutuhan
Noun	-nya	kontraknya, strateginya, rambutnya

Finally, the methods achieve a finer granularity of clustering that suggest a semantic significance to the results beyond the part-of-speech tagging by grouping in clusters such as days, places, and people. This suggests that similar methods can also be used for name-entity recognition.

5 Conclusion

We have presented in this paper several effective methods for part-of-speech tagging in the absence of both a tagged corpus, a lexicon, and without a set of predefined tags. We evaluated the performance of the methods we proposed and their variants and compared them with the state of the art reference, the Schutze's methods (of which one of our methods is an extension). We showed that the extended Schutze's method that we have proposed yields a consistent improvement over all other methods with an average precision of 90%, and tagging correctly over 80% of the words in the corpus.

Further manual inspection of the clusters obtained during the experiments of various corpora suggested applications or combination of our approach to other domains such as stemming and name-entity recognition.

References

1. Biber, Douglas (1993). Co-occurrence Patterns among Collocations: A Tool for Corpus-based Lexical Knowledge Acquisition. Computational Linguistics, 19(3):531-538.
2. Brill, Eric; Magerman, David; Marcus, Mitch; and Santorini, Beatrice (1990). Deducing Linguistic Structure from the Statistics of Large Corpora. In Proceedings of the DARPA Speech and Natural Language Workshop, pages 275-282.

3. Brill, Eric (1993). Automatic Grammar Induction and Parsing Free Text: A Transformation-based Approach. In Proceedings of ACL 31. Columbus OH.
4. Charniak, Eugene; Hendrickson, Curtis; Jacobson, Neil; and Perkowitz, Mike (1993). Equations for Part-of-speech Tagging. In Proceedings of the Eleventh National Conference on Artificial Intelligence, pages 784-789.
5. Church, Kenneth W. (1989). A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In Proceedings of ICASSP-S9. Glasgow, Scotland.
6. Cutting, Doug; Kupiec, Julian; Pedersen, Jan; and Sibun, Penelope (1991). A Practical Part-of-Speech Tagger. In The 3rd Conference on Applied Natural Language Processing. Trento, Italy.
7. Cutting, Douglas R; Pedersen, Jan O; Karger, David; and Tukey, John W. (1992). Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In Proceedings of SIGIR 'g2, pages 318-329.
8. Elman, Jeffrey L. (1990). Finding Structure in Time. Cognitive Science, pages 179-211.
9. Finch, Steven and Chater, Nick (1992). Bootstrapping Syntactic Categories using Statistical Methods. In Walter Daelemans and David Powers, editors, Background and Experiments in Machine Learning of Natural Language, pages 229-235. Tilburg University. Institute for Language Technology and AI.
10. Finch, Steven (1993). Finding Structure in Language. Ph.D. Thesis, University of Edinburgh.Scotland.
11. Francis, W.N. and Kucera, F. (1982). Frequency Analysis of English Usage. Houghton Mifflin, Boston.
12. Jelinek, F (1985). Robust Part-of-speech Tagging using a Hidden Markov Model. Technical Report. IBM, T.J. Watson Research Center.
13. Kneser, Reinhard and Ney, Hermann (1993). Forming Word Classes by Statistical Clustering for Statistical Language Modelling. In Reinhard Kohler and Burghard B. Rieger, editors, Contributions to Quantitative Linguistics, pages 221-226. Dordrecht, The Netherlands.
14. Kupiec, Julian (1992). Robust Part-of-Speech Tagging using a Hidden Markov Model. Computer Speech and Language, 6:225-242.
15. Manning, C. D. and Schutze, Hinrich (1999). Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA.
16. Marcus, M., Kim, G., Marcinkiewicz, M., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The Penn Treebank: Annotating Predicate Argument Structure. In ARPA Human Language Technology Workshop.
17. Miller, Bruce (2000). http://math.nist.gov/javanumerics/jama/. National Institute of Standards and Technology.
18. Schutze, Hinrich (1999). Distributional Part-of-speech Tagging. In EACL7, pages 141-148.
19. van Dongen, Stijn (2000). Graph Clustering by Flow Simulation. Ph.D. Thesis, University of Utrecht. The Netherlands.