Automatic Part of Speech Tagging for Arabic: An Experiment Using Bigram Hidden Markov Model

Mohammed Albared, Nazlia Omar, Mohd. Juzaiddin Ab Aziz, and Mohd Zakree Ahmad Nazri

University Kebangsaan Malaysia, Faculty of Information Science and Technology, Department of Computer Science
mohammed_albared@yahoo.com,{no,din}@ftsm.ukm.my
http://www.ukm.my

Abstract. Part Of Speech (POS) tagging is the ability to computationally determine which POS of a word is activated by its use in a particular context. POS tagger is a useful preprocessing tool in many natural languages processing (NLP) applications such as information extraction and information retrieval. In this paper, we present the preliminary achievement of Bigram Hidden Markov Model (HMM) to tackle the POS tagging problem of Arabic language. In addition, we have used different smoothing algorithms with HMM model to overcome the data sparseness problem. The Viterbi algorithm is used to assign the most probable tag to each word in the text. Furthermore, several lexical models have been defined and implemented to handle unknown word POS guessing based on word substring i.e. prefix probability, suffix probability or the linear interpolation of both of them. The average overall accuracy for this tagger is 95.8

Keywords: Arabic languages, Hidden Markov model, Smoothing methods, Part of speech tagging.

1 Introduction

Part of speech disambiguation is the ability to computationally determine which POS of a word is activated by its use in a particular context. POS tagging is an important pre-processing activity in many NLP applications like speech processing and machine translation. Being one of the first processing steps in any such application, the performance of the POS tagger directly impacts the performance of any subsequent text processing steps. The main challenges in POS tagging involve resolving the MorphoSyntactic ambiguity where a word can occur with different POS tags in different contexts, and handling unknown words i.e. words that appear in the test data but they do not appear in the training data.

Arabic language is a Semitic language spoken by over 250 million people and the language is used by 1.2 billion Muslims. Arabic language has many varieties that diverge widely from one another. Arabic is a strongly structured, highly

J. Yu et al. (Eds.): RSKT 2010, LNAI 6401, pp. 361–370, 2010.

derivational and grammatically ambiguous language. In spite of the widely use of the Arabic language in the world and the huge amount of electronic available Arabic data online, Arabic is still a language, for which very few computational linguistic resources have been developed. Up to date, there are a few annotated corpora for Arabic language, which are not freely available for research activity. Moreover, the domain of the text in these corpora is limited to newswire data. They are only suitable for some domains. As a matter of fact, taggers trained with tagged corpora from a domain perform quite poorly in others. So our motivation for the development of this work is to provide a tool that can help to accelerate the development of such resources, which in their turn can be used in further research to train more accurate NLP tools.

In this paper, we have developed a bigram HMM POS tagger for Arabic text. The HMM parameters are estimated from our small training data. We have studied the behavior of the HMM applied to Arabic POS tagging using small amount of data. For dealing with sparseness problem, we use several smoothing algorithms. Furthermore, we use word string information (suffix and prefix) features, beside the context information (previous tag) to guess the POS tag of an unknown word. The tagger has an accuracy of about 95% on the test data provided.

The remainder of this paper is organized as follows: First, some previous work in the field is described in Section 2. A brief overview of the Arabic language is introduced in Section 3, and then the used tagset and data is described in Section 4. Section 5 introduces the used smoothing methods. Unknown word handling is described in Section 6. The results of experiments are introduced in Section 7 together with a short discussion. Finally, Section 7 concludes the paper and points to ways in which we believe that the results can be improved in the future.

2 Related Work

Different techniques have been used for POS tagging in the literature. Some of these techniques focused on rule based using linguistic rules. On the other side, POS tagging is viewed as a classification problem: the tagset is the classes and a machine learning algorithm is used to assign each occurrence of a word to one class based on the evidence from the context. This machine learning methods range from methods with full supervision, to fully unsupervised methods. The supervised POS disambiguation approaches use machine learning techniques to learn a classifier from labeled training sets such as Maximum Entropy Model [1], Hidden Markov Model [2] and Conditional Random field [3].

Unsupervised POS disambiguation methods are based on unlabeled corpora, and do not exploit any manually POS tagged corpus such as Contrastive Estimation [4], Fully Bayesian approach [5] and Prototype Driven Learning [6].

Different Arabic taggers have been recently proposed. Some of them use supervised machine learning approaches such as [7] and [8], while others are using hybrid approaches, combining rule based approach and statistical approach such as [9] (for more details about works on Arabic taggers, see [10])

Almost all of these taggers are generally developed for Modern Standard Arabic (MSA) and few works are interested in Classical Arabic [7]. Moreover, most of these taggers are assumed closed vocabulary assumption in which the test data only contain words that appear in the training data.

For these reasons, we developed our tagger which is trained and tested on a hybrid annotated data, which come from both MSA and CA. Moreover, in order to overcome the limitation of the small training data, we propose several lexical models based on word substring information to handle unknown word POS problem.

3 Overview of Arabic Language

Arabic language is a Semitic language spoken by over 250 million people and the language is used by 1.2 billion Muslims. It is one of the seven official languages of the United Nations. Arabic is a strongly structured and highly derivational language. The Arabic language has many varieties that diverge widely from one another. Arabic language consists of Modern standard Arabic and set of spoken dialects.

Arabic words are quite different from English words, and the word formation process for Arabic words is quite complex. The main formation of English word is concatenative i.e. simply attach prefixes and suffixes to derive other words . In contrast, the main word formation process in Arabic language is inherently non-concatenative. Words are derived by interdigitating roots into patterns. Arabic words is derived by inserting root's radical characters into pattern's generic characters. The first radical is inserted into the first generic character, the second radical fills the second generic and the third fills the last generic as shown in Table 1.

Arabic language is grammatically ambiguous. There are many sources of ambiguity in Arabic language such as:

- Arabic is highly inflective language: Arabic words are often formed by concatenating smaller parts. A word may be segmented into different candidate sequences of segments each of which may be ambiguous with respect to the choice of a POS tag. Moreover, Arabic words are segmentation ambiguous;

Table 1. Arabic Word deviation process

Pattern	Pattern in Arabic	Root	The resulting word
XوXX	فعول	علم	علوم
Xا|Xت|X	افتعَال	قصد	اقتصَاد
يستXXX	يستفعل	عمل	يستعمل
تXXX	تفعل	طور	تطور
يXXXون	يفعلون	عمل	يعملون

there is different alternative segmentation of the words. The result is a relatively high degree of ambiguity.
- The lack of the short vowels in the modern Arabic text. Short vowels are special symbols that smooth the processes of reading and understanding the Arabic text especially for foreign readers. An Arabic word without short vowels can have different meanings and also different POS .However, in almost all modern Arabic text, short vowels are no longer used.

4 The Used Data and Tagset

Arabic linguists and grammarians classify Arabic words into nominals (أسمَاء), verbs (أفعَال) and particles (حروف). The nominals include nouns, pronouns, adjectives and adverbs. The verbs include perfect verbs, imperfect verbs and imperative verbs. The particles include prepositions, conjunctions and interrogative particles, as well as many others. Our tagset have been inspired by Arabic TreeBank (ATB) POS Guidelines [11].The used tagset consists of 23 tags (see Table 2). Our training corpus consists of 26631 manually annotated word forms from two types of Arabic texts: the Classical Arabic text and from Modern Standard Arabic. We use this corpus to train and test our tagger. We split the corpus into training set with size 23146 words and test set with size 3485 words.

Table 2. Arabic Part-of-speech tagset

POS Tag	Label	POS Tag	Label
Conjunction	**CC**	Possessive Pronoun	**POSS_PRON**
Number	**CD**	Imperfective Verb	**VBP**
Adverb	**ADV**	Non Inflected Verb	**NIV**
Particle	**PART**	Relative Pronoun	**REL_PRON**
Imperative Verb	**IV**	Interjection	**INTERJ**
Foreign Word	**FOREIGN**	Interrogative Particle	**INTER_PART**
Perfect Verb	**PV**	Interrogative Adverb	**INTER_ADV**
Passive Verb	**PSSV**	Demonstrative Pronoun	**DEM_ PROP**
Preposition	**PREP**	Punctuation	**PUNC**
Adjective	**ADJ**	Proper Noun	**NOUN_PROP**
Singular Noun	**SN**	Personal Pronoun	**PRON**
Plural Noun	**SPN**		

5 Our Approach: The Bigram HMM Tagger

Hidden Markov Model (HMM) is a well-known probabilistic model, which can predict the tag of the current word given the tags of one previous word (bi-gram) or two previous words (trigram). The HMM tagger assign a probability value to each pair$\langle w_1^n, t_1^n \rangle$, where $w_1^n = w_1 ..... w_n$ is the input sentence and $t_1^n = t_1 ... t_n$ is

the POS tag sequence. In HMM, the POS problem can be defined as the finding the best tag sequence t_1^n given the word sequence w_1^n. This is can be formally expressed as:

$$t_1^n = \arg\max_{t_1^n} \prod_{i=1}^{n} p\left(t_i|t_{i-1}\ldots.t_1\right).p\left(w_i|t_i\ldots.t_1\right) \ . \tag{1}$$

Since it is actually difficult or even impossible to calculate the probability of the formp $\left(t_i|t_{i-1}\ldots.t_1\right)$ for large i because of the sparse problem, bi-gram or tri-gram model is often used to alleviate data sparseness problem. Here we assumed that the word probability is constrained only by its tag, and that the tag probability is constrained only by its preceding tag, the bigram model:

$$t_1^n = \arg\max_{t_1^n} \prod_{i=1}^{n} p\left(t_i|t_{i-1}\right).p\left(w_i|t_i\right) \ . \tag{2}$$

The probabilities are estimated with relative frequencies from the training data. The tag-word pair probabilities and tag transition probabilities are commonly called as lexical probabilities, and transition probabilities, respectively. Having computed the lexical and transition probabilities and assigning all possible tag sequences to all words in a sentence, now we need an algorithm that can search the tagging sequences and find the most likely sequence of tags. For this purpose, we use Viterbi algorithm to find the most probable sequence of POS tags (t_1^n) for a given sentence. The Viterbi algorithm is the most common decoding algorithm for HMMs that gives the most likely tag sequence given a set of tags, transition probabilities and lexical probabilities.

6 Smoothing

The simplest way to compute the parameters for our HMM is to use relative frequency estimation, which is to count the frequencies of word/tag and tag/tag. This way is called Maximum Likelihood Estimation (MLE). MLE is a bad estimator for statistical inference, because data tends to be sparse. This is true even for corpus with large number of words. Sparseness means that various events, here bigrams transition, are either infrequent or unseen. This leads to zero probabilities being assigned to unseen events, causing the probability of the whole sequence to be set to zero when multiplying probabilities. There are many different smoothing algorithms in the literature to handle sparseness problem [12], all of them consisting of decreasing the probability assigned to the known event and distributing the remaining mass among the unknown events. Smoothing algorithms are primarily used to better estimate probabilities when there is insufficient data to estimate probabilities accurately. In the following subsections, we give a brief description of the used smoothing techniques.

6.1 Laplace Estimation

Laplace estimation is the simple smoothing method for data sparseness [13]. In this method, we add one for each bigram frequency count either seen or unseen in the training data. Since there is T tags in the tagset, and each one got incremented, we also need to adjust the denominator. We add the number of tags in the tagset to the unigram count of the tag t_{i-1}. The main problem of Laplace method is that it assigns overvalued probabilities to unseen bigram, which is principally not true.

$$P_{Laplace}(t_i|t_{i-1}) = \frac{C(t_{i-1}t_i)+1}{C(t_{i-1})+T} \,. \tag{3}$$

6.2 The Kneser Ney Smoothing

Chen et al. [12] showed that Kneser-Ney smoothing and its variations outperform all other smoothing techniques. In Kneser-Ney smoothing, the transition probabilities is formulated as:

$$P_{\mathrm{KN}}(t_i|t_{i-1}) = \frac{\max\{f(t_{i-1}t_i)-D,0\}}{\sum_{t_i} f(t_{i-1}t_i)} + \frac{\mathrm{D}\cdot \mathrm{N}_{1+}(t_{i-1},\bullet)}{\sum_{t_i} f(t_{i-1}t_i)} \cdot \frac{\mathrm{N}_{1+}(\bullet,t_i)}{\mathrm{N}_{1+}(\bullet\bullet)} \,. \tag{4}$$

where $\mathrm{N}_{1+}(\bullet,\bullet) = |\{(\mathrm{t_{i-1}t_i}) : \mathrm{f(t_{i-1}t_i)1}\}|, \mathrm{D} = \frac{\mathrm{n_1}}{\mathrm{n_1+2n_2}}$ and n1 and n2 are the total number of bi-grams with exactly one or two counts, respectively, in the training corpus.

6.3 The Modified Kneser Ney Smoothing

Modified Kneser-Ney smoothing is an interpolated variation of Kneser Ney smoothing with an augmented version of absolute discounting. Modified Kneser-Ney smoothing is proposed by [12].They found that it outperforms all other methods. However, in this smoothing algorithm,the transition propability $\mathrm{p}\left(\mathrm{t_i}|\mathrm{t_{i-1}}\right)$ is calculated as in the following equation:

$$P_{MKN}(t_i|t_{i-1}) = \frac{f(t_{i-1}t_i)-D(f(t_{i-1}t_i))}{\sum_{t_i} f(t_{i-1}t_i)} + \gamma(t_{i-1}) \cdot \frac{\mathrm{N}_{1+}(\bullet,t_i)}{\mathrm{N}_{1+}(\bullet\bullet)} \,. \tag{5}$$

where $\mathrm{D(f)} = \begin{cases} 0 \text{iff} = 0 \\ \mathrm{D_1 iff} = 1 \\ \mathrm{D_2 iff} = 2 \\ \mathrm{D_{3+}} \text{iff} \geq 3 \end{cases}$ and $\gamma(\mathrm{t_{i-1}}) = \frac{\mathrm{D_1N_1(t_{i-1},\bullet)+D_2N_2(t_{i-1},\bullet)+D_{3+}N_{3+}(t_{i-1},\bullet)}}{\sum_{\mathrm{t_i}} \mathrm{f(t_{i-1}t_i)}}$.

The D values for the model are calculated using the following equations: $\mathrm{Y} = \frac{\mathrm{n_1}}{\mathrm{n_1+2n_2}}$,
$\mathrm{D_1} = 1 - 2\mathrm{Y}\frac{\mathrm{n_2}}{\mathrm{n_1}}, \mathrm{D_2} = 2 - 3\mathrm{Y}\frac{\mathrm{n_3}}{\mathrm{n_2}}$ and $\mathrm{D_3} = 3 - 4\mathrm{Y}\frac{\mathrm{n_4}}{\mathrm{n_3}}$.

7 Unknown Words Handling

Applying a POS tagger to new data can result in two types of errors: misclassification of the ambiguous words and failure to find the right class label for an unknown word that appears in the test data but not in the training data. The number of words in the corpus is always finite. It is not possible to cover all words of the language. In addition, new words are being created everyday, so it is very difficult to keep the dictionary up-to-date. Handling of unknown words is an important issue in POS tagging. For words which have not been seen in the training set, is estimated based on features of the unknown words, such as whether the word contains a particular suffix. We use a successive abstraction scheme. The probability distribution for an unknown word suffix is generated from all words in the training set that have the same suffix up to some predefined maximum length. This method was proposed by Samuelsson [14] and implemented for English and German [2]. The method is expressed in the following equation:

$$P\left(t|c_{n-i+1},...,c_n\right) = \frac{P(t,c_{n-i+1},...,c_n) + \theta P(t,c_{n-i+2},...,c_n)}{1+\theta} \ . \tag{6}$$

$$\theta = \frac{1}{S-1}\sum_{J=1}^{S}\left(P\left(t_j\right)-\overline{P}\right)^2, \overline{P} = \frac{1}{s}\sum_{J=0}^{S}P\left(t_j\right)$$

where $c_{n-i+1},...,c_n$ represent the last n characters of the word. In addition to word suffix, the experiments utilize the following features: the presence of non-alphabetic characters and the existence of foreign characters. In addition to the suffix guessing model, we define another lexical model to handle unknown words POS guessing based on word prefix information. The new lexical model is expressed as in the following equation:

$$P(t|w) = P\left(t|c_1,...,c_m\right) \ . \tag{7}$$

where $c_1,...,c_m$ represent the first m characters of the word w . The probability distribution in Equation 7 is estimated as in Equation 6. Furthermore, we have defined another lexical model based on combining both word suffix information and word prefix information. The main linguistic motivation behind combining affixes information is that in Arabic word sometimes an affix requires or forbids the existence of another affix [15]. Prefix and suffix indicate substrings that come at the beginning and end of a word respectively, and are not necessarily morphologically meaningful. In this model, the lexical probabilities are estimated as follows:

1. Given an unknown word w, the lexical probabilities P(suffix(w)|t) are estimated using the suffix tries as in Equation 6.
2. Then, the lexical probabilities P(prefix(w)|t) are estimated using the prefix tries as in Equation 6. Here, the probability distribution for a unknown word prefix is generated from all words in the training set that have the same prefix up to some predefined maximum length.

3. Finally, we use the linear interpolation of both the lexical probabilities obtained from both word suffix and prefix to calculate the lexical probability of the word w as in the following equation :

$$\mathrm{P}\,(\mathrm{w}|\mathrm{t}) = \lambda \mathrm{P}\,(\mathrm{suffix(w)}|\mathrm{t}) + (1-\lambda)\mathrm{P}(\mathrm{prefix(w)}|\mathrm{t})\ . \tag{8}$$

where λ is an interpolation factor, experimentally set to 0.7. prefix(w) and suffix(w) are the first m and the last n characters, respectively.

8 Experiments and Results

For our experiments, we evaluated the performance of the bigram HMM with different smoothing techniques. We divide our corpus into training set with size 23146 words and test set with 3485 words. The test set contains 263 unknown words. We define the tagging accuracy as the ratio of the correctly tagged words to the total number of words. Table 3 summarizes the results of Bigram Hidden Markov Model with smoothing methods.Among smoothing algorithms, the modified Kneser-Ney approach yields the highest results, with the Kneser-Ney algorithm close behind, at 95.4. Finally, The Laplace algorithm achieved an accuracy of 95.1. From Table 3, we notice that the different smoothing techniques exhibit the same behaviors due to the reasons that Arabic is free ordering which reduced the number of the missing bigrams transition probabilities and also due to the small size of the used tagset. However, the results of unknown words POS guessing in Table 3 is achieved using suffix guessing technique with successive abstraction. These results also show that this techniques which proved to be effective for some languages does not work well for Arabic language. The suffix guessing algorithem proved to be a good indicator for unknown word POS guessing in English and German[2].

Furthermore, in Table 4, we summarize and compare the results of the three unknown word POS guessing algorithms. These algorithms are the suffix guessing algorithm, the prefix guessing algorithm and the linear interpolation of both prefix guessing algorithm and suffix guessing algorithm for unknown words POS guessing. As we can see, the third algorithm which combine information from both suffix and prefix, gives a considerable rise in accuracy compared to the suffix guessing method. This result of the combined algorithm is achieved when the value of in Equation 8 experimentally set to 0.7. We can see from these expriments that Arabic word suffix is a better indicator than its prefix in guessing

Table 3. Tagging accuracy obtained with bigram HMM coupled with smoothing techniques

Smoothing	Overall acc.	Unknown words acc
Laplace	95.1	65.8
Kneser-Ney	95.4	70.3
modified Kneser-Ney	95.5	70.7

its POS. But, the combination of both word suffix and prefix does better than using one of them alone.In general,unknown word accuracy for Arabic is not so high due to many reasons such as Arabic has complex morphology than English i.e. the non concatenative nature of Arabic words and the Arabic words affixes are ambiguous, short and sparse. However, we have obtained an overall accuracy of 95.8% which is quite encouraging especially with the small size of our training corpus. The main remark which can be extracted from these results is that a competitive Arabic HMM taggers may be built using relatively small train sets, which is interesting especially, with the lack of language resources.

Table 4. Unknown word POS tagging tagger accuracies with the three lexical models

Model	% of unknown word	Unknown acc.	Overall acc.
Suffix guessing algorithm	7.5	70.7	95.5
Prefix guessing algorithm	7.5	60.8	94.7
The combined guessing algorithm	7.5	73.5	95.8

9 Conclusion

Part of speech tagging is an important tool in many NLP applications. In addition, it accelerates the development of large manually annotated corpora. In this paper, we described preliminary experiments in designing HMM-based part-of speech tagger for Arabic language. We have investigated the best configuration of HMM when small amount of data is available. We implemented bigram HMM-based disambiguation procedure. HMM parameters were automatically estimated from our training corpus of size 23146 words. We conducted a series of experiments using different smoothing techniques, such as Laplace and Kneser-Ney and the modified Kneser-Ney. Furthermore, three lexical models have been used to handle unknown words POS tagging. These algorithms are suffix tries and successive abstraction and prefix guessing algorithm and the linear interpolation of both suffix guessing algorithm and prefix guessing algorithm. Our Arabic part-of speech tagger achieves near 95.8 % tagging accuracy. This result is very encouraging especially with the small size of our training corpus. In the future, we plan to improve the tagging accuracy of unknown words by increasing the size of our training corpus as well as by using specific features of Arabic words, which can be better predictor for Arabic words POS than words suffixes and prefixes.

References

1. Ratnaparkhi, A.: A maximum entropy part of speech tagger. In: Brill, E., Church, K. (eds.) Conference on Empirical Methods in Natural Language Processing, University of Pennsylvania (1996)
2. Brants, T.: TnT: A statistical part-of-speech tagger. In: Proceedings of the 6th Conference on Applied Natural Language Processing, Seattle, WA, USA (2000)

3. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the International Conference on Machine Learning, MA, USA (2001)
4. Smith, N., Eisner, J.: Contrastive estimation: Training log-linear models on unlabeled data. In: Proceedings of ACL 2005, pp. 354–362 (2005)
5. Goldwater, S., Griffiths, T.: A fully Bayesian approach to unsupervised part-of-speech tagging. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (2007)
6. Haghighi, A., Klein, D.: Prototype-driven learning for sequence models. In: Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the ACL, pp. 320–327 (2006)
7. El Hadj, Y., Al-Sughayeir, I., Al-Ansari, A.: Arabic Part-Of-Speech Tagging using the Sentence Structure. In: Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt (2009)
8. Al Shamsi, F., Guessoum, A.: A hidden Markov model-based POS tagger for Arabic. In: Proceeding of the 8th International Conference on the Statistical Analysis of Textual Data, France, pp. 31–42 (2006)
9. Zribi, C., Torjmen, A.: An Efficient Multi-agent System Combining POS-Taggers for Arabic Texts. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 121–131. Springer, Heidelberg (2006)
10. Albared, M., Omar, N., Ab Aziz, M.J.: Arabic Part of Speech Disambiguation: A Survey. International Review on Computers and Software, 517–532 (2009)
11. Maamouri, M., Bies, A., Krouna, S., Gaddeche, F., Bouziri, B.: Arabic Tree Banking Morphological Analysis And POS Annotation, Ver. 3.8, Linguistic Data Consortium, Univ. Pennsylvania (2009), http://projects.ldc.upenn.edu/ArabicTreebank/ATB-POSGuidelines-v3.7.pdf
12. Chen, S.F., Goodman, J.: An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Computer Science Group, Harvard University (1998)
13. Jurafsky, D., Martin, J.H.: SPEECH and LANGUAGE PROCESSING: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Prentice-Hall, Englewood Cliffs (2000)
14. Samuelsson, C.: Handling sparse data by successive abstraction. In: COLING 1996, Copenhagen, Denmark (1996)
15. Attia, M.: Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation. PhD thesis, School of Languages, Linguistics and Cultures, Univ. of Manchester, UK (2008)