An Efficient Multi-agent System Combining POS-Taggers for Arabic Texts

Chiraz Ben Othmane Zribi, Aroua Torjmen, and Mohamed Ben Ahmed

RIADI laboratory, National School of Computer Sciences, 2010,
University of La Manouba, Tunisia
{Chiraz.benothmane, Aroua.torjmen,
Mohamed.benahmed}@riadi.rnu.tn

Abstract. In this paper, we address the problem of Part-Of-Speech tagging of Arabic texts with vowel marks. After the description of the specificities of Arabic language and the induced difficulties on the task of POS-tagging, we propose an approach combining several methods. One of these methods, based on sentences patterns, is original and very attractive. We present, afterward, the multi-agent architecture that we adopted for the conception and the realization of our POS-tagging system. The multi-agent architecture is justified by the need for collaboration, parallelism and competition between the different agents. Finally, we expose the implementation and the evaluation of the system implemented.

1 Introduction

The process of Part-Of-Speech tagging was widely automated for English and French and for many others European languages giving a rate of accuracy ranging from 95 % to 98 %. We find on the Web, many tagged corpora as well as programs of POS-tagging for these languages. The methods used by these POS-taggers are various, namely stochastic approaches such as the Hidden Markov Model [1], the decision trees [2], the maximum entropy model [3], rules-based approaches inspired in their majority of the transformation rules-based POS-tagging [4], hybrid approaches [5] (statistics and rules-based), or combined ones [6] and [7].

Unfortunately, the situation is different for Arabic as there are neither POS-taggers nor tagged corpora available. Nevertheless, some Arabic POS-taggers [8], [9] and [10] started to appear with an accuracy going from 85% to 90% on average for texts with vowel marks and by about 65% for texts without vowel marks.

This gap noted for Arabic language is especially due to, its particular characteristics, which, involve firstly, a rate of grammatical ambiguity relatively more significant than for other languages, and secondly, make impossible the application of existing POS-taggers without any change. Thus, obtaining improving accuracy remains a challenge to reach for Arabic language.

Accordingly, we propose a POS-tagging system for Arabic texts. Due to the complexity of the problem, and in order to decrease grammatical ambiguity, we have restricted the scope of our investigation: we only treat texts with vowels marks.

The remainder of this paper is organized as follows: First, we present the Arabic language characteristics making the task of POS-tagging more difficult. We then present the general principle of our combined approach. Next, we show the general architecture of our multi-agent system and present a detailed description of the work of each agent. Finally, we present the method we used to evaluate the efficiency of our system and the results obtained.

2 Difficulties of Arabic Languages

In Arabic, the problem of POS-tagging is much more complicated than in other languages. Indeed, Arabic has numerous writing constraints such as vowels, agglutination and grammatical ambiguity, which can lead to ambiguities.

2.1 Vowel Marks

The vowel marks in words are vocalic signs that facilitate the reading and the comprehension of texts written in Arabic. Without vowels, the reader has to see the context to find the good vowels of the textual form, because Arabic words are vocalically ambiguous. This vocalic confusion involves naturally much more grammatical ambiguity.

Table 1. Example of vocalic ambiguity

	كتب	
كَتِّبْ	*Kattib*	Make write
كُتِّبَ	*Kuttiba*	Has been made write
كُتِبَ	*Kutiba*	Has been written
كَتَبَ	*Kataba*	Wrote
…	…	…

2.2 Agglutination

Arabic is an agglutinative language. Textual forms are made of the agglutination of prefixes (articles, prepositions, conjunctions) and suffixes (linked pronouns) to the stems (inflected forms). In general, to obtain the different decompositions of a textual form, a morphological analyzer is needed. The ambiguities of decomposing textual forms induce a significant ambiguity of tagging. When the text is without vowel marks, the decomposing ambiguity increases.

Table 2. Example of decomposing ambiguity

	أكبر	
+ أكبر +	*Akabara*	Did it grow?
أ + كبر +	*Akbar*	Higher
أك + بر +	*AkaBir*	Like benevolence

2.3 Grammatical Ambiguity

Arabic words are grammatically ambiguous. The statistics carried out in definition by [11] confirm this ambiguity. The author noted the importance of the rate of grammatical ambiguity for the lexical forms with vowel marks, which is equal to 2.8 on average. This rate increases by the absence of the vowels to reach 5.6 possible tags per lexical form. Because of the agglutination of affixes to lexical forms, the rate of grammatical ambiguity is more significant for textual forms. According to the counting carried out by [8] on texts with vowel marks this rate is equal to 5.6 on average, and could reach an average of 8.7 for texts without vowel marks.

3 Suggested Approach

To achieve our POS-tagging system, we opted for combining methods (probabilistic and rules–based) in a multi-agent architecture.

3.1 Combined Method

We combine different methods trying to benefit from advantages for each method used and to improve our system's accuracy. This implies the construction of a number of POS-taggers where each operates according to the principle of the method that it represents. Each POS-tagger proposes one tag for the treated word and by voting the best one is assigned as the final tag to the target word.

3.2 A Multi-agent Architecture

The following arguments can justify the choice of this architecture, in addition to its originality:

- Combination of several methods: we combine several methods to realize our POS-tagging system.
- Competition and parallel work of agents: the POS-taggers agents treat the same sentence, which is extracted from the text to be tagged concurrently.
- Communication and cooperation between agents: The agents' system can communicate and cooperate for example to solve unknown words.

4 Part-of-Speech Tagger

We considered the following hypotheses to accomplish our POS-tagging system:

- We chose a supervised training mode to construct linguistic and probabilistic training data, from a pre-treated corpus (morphologically analyzed and manually tagged).
- We considered a sentence as a sequence of words limited by punctuations.
- The input of our system is the set of part of speech tag proposed by the morphological analyzer for each textual.

4.1 Tag Sets

In this work, we manipulate two main tag sets. The first one involves simple tags, also called micro tags. These tags are assigned to lexical units. We count 223 tags for the inflected forms and 65 for the affixes. The second tag set is devoted to textual forms and is constructed by the licit combination of the simple tags {*prefix's tag* + *simple form's tag* + *suffix's tag*}.

We consider two other tag sets as well. Firstly, we use 22 macro tags, which are less detailed than micro tags. Secondly, we use a tag set representing the three principle part-of-speech tags: **S**ubstantive, **V**erb and **P**article (SVP). We use these tag sets to a simple matching between their tags and the micro tags. This is, in order, to make a comparison between the results given by taggers and to adapt the results of our system to various applications requirements.

4.2 System Agents

Some agents participate to accomplish the global objective of our POS-system that consists in assigning appropriate tags to each textual form of a given text. We cite:

- Sentences' extracting agent;
- Tagger agents,
- Unknown words solver agent;
- Voting agent.

The following figure illustrates the general architecture of this system.

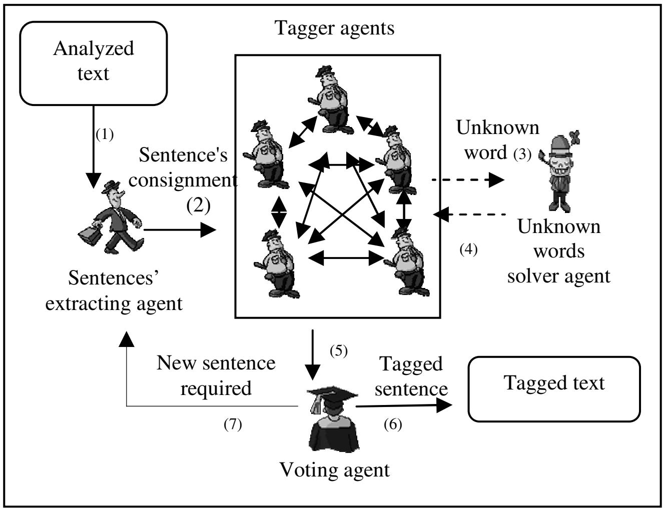

Fig. 1. General architecture of the POS-tagging system

Sentences' Extracting Agent. This agent is responsible of the extraction of the sentences from the text to tag. Each word in a sentence has a set of tags proposed by the morphological analyzer[1] developed by [11]. When it loads a sentence, the

[1] The morphological analyzer gives for each word all possible partitions in prefix, stem and suffix and for each partition, a set of all potential tags.

sentences' extracting agent activates all tagger agents to start the tagging of this sentence.

Tagger Agents. Given a sentence, five POS-taggers agents will work in parallel, each applying its own method, aiming to find for each word of the sentence the suitable tag among the tags proposed by the morphological analyzer**.**

Unigram Tagger Agent. For each word of a sentence received, the unigram tagger:

1. recuperates tags proposed by the morphological analyzer;
2. accedes to a lexicon which is containing the different words of the training corpus and their tags with their occurrence's frequencies;
3. seeks the target word in this lexicon;
4. chooses the most frequent tag for this word.

Bigram Tagger Agent. This tagger uses the binary succession probabilities recovered from the training corpus and saved in a binary succession matrix. We calculate the binary succession probability as follows:

$$p(t_i \backslash t_{i-1}) = \frac{\text{number of occurences of the succession } (t_{i-1}, t_i)}{\text{number of occurrence s of } t_{i-1}} \,. \tag{1}$$

The bigram tagger follows these steps to tag a word, it:

1. recovers the tags proposed by the morphological analyzer;
2. recovers the tag of the word preceding the target word;
3. accedes to the matrix of binary succession probabilities;
4. chooses the tag belonging to the set of tags proposed by the morphological analyzer having the higher binary transition probability considering the tag of the previous word;
5. assigns the tag that it found to the word to tag.

Trigram Tagger Agent. This trigram tagger agent works similarly to the precedent one, but it takes into account ternary succession probabilities recovered from the training corpus and saved in a ternary succession matrix. We determine the ternary succession probability as follows:

$$p(t_i \backslash t_{i-2}, t_{i-1}) = \frac{\text{number of occurences of the succession } (t_{i-2}, t_{i-1}, t_i)}{\text{number of occurrence s of the succession } (t_{i-2}, t_{i-1})} \,. \tag{2}$$

Here, the principle of tagging each word in a given sentence consists in:

1. recovering the tags proposed by the morphological analyzer;
2. recovering the two previous tags in relation with the target word;
3. acceding to the matrix of ternary transition probabilities;
4. choosing the grammatical tag, which belongs to the tags proposed by the morphological analyzer and has the higher ternary transition probability considering the two tags of the two previous words;
5. assigning the tag found to this word.

Hidden Markov Model Tagger Agent. This tagger agent operates according to Hidden Markov Model's principle.

Given a sequence of n words $W = w_1 \ldots w_{n,}$ this tagger tries to find the tag sequence $T = t_1 \ldots t_{n,}$ that maximizes the conditional probability $p(T \backslash W)$.

We note:

Max $T=$ arg $\text{Max}_T\ p(T\backslash W)$

By some assumptions[2] :

$$\text{Max } T = \arg \text{Max}_T \prod_{i=1}^{n} p(w_i \backslash t_i) \times p(t_i \backslash t_{i-1}) \ . \tag{3}$$

Where:

$p(w_i\backslash t_i)$ is the emission probability that is calculated with the following formula :

$$p(w_i \backslash t_i) = \frac{\text{number of occurences of } w_i \text{ tagged with } t_i}{\text{number of occurrences of } t_i} \ . \tag{4}$$

and $p(t_i\backslash t_{i-1})$ is the transition probability that is determined as follows:

$$p(t_i \backslash t_{i-1}) = \frac{\text{number of occurences of succession } (t_{i-1}, t_i)}{\text{number of occurrences of } t_{i-1}} \ . \tag{5}$$

Where:

$p(t_1\backslash t_0) = p(t_1)$ called initial probability.

When it receives a sentence, including for each word all the tags proposed by the morphological analyzer, this tagger agent applies the VITERBI algorithm [12]. The latter takes all the needed frequencies from the training corpus and tries to find the tag sequence that has the maximum likelihood.

Agent based on Sentences Patterns. The sentences patterns–based method presented here is new and has not been approached before. We define a sentence pattern as a model of sentence made of a succession of tags.

Example:
The sentence: “The child eats a cake” can matches with the following pattern: “Definite-Article + Noun + Verb + Indefinite-Article + Noun”.

In the practice, the possession of all sentences patterns for a language is difficult. That is why this tagger manipulates the longest successions of tags of adjustable size. The sentence pattern considers the positional character of tags in the sentence (1st tag, 2nd tag…).

[2] Independency assumption and Markov assumption k=1 (using binary successions).

The principle of this tagger consists in:

1. considering the first word of the sentence and extracting the tags that have been assigned by the morphological analyzer;
2. acceding to the set of sentences patterns and seeking the patterns that start with one of the tags proposed by the morphological analyzer;
3. treating the second word. Among models found in patterns, the second tag correlates to one of tags proposed by the morphological analyzer for this word.
4. this process is repeated until the words of the sentence are tagged completely considering the position of words while the matching between the tags proposed by the morphological analyzer for the treated word and the tags of patterns are proposed. Thus, the number of the candidates patterns decreases when the tagger goes forward in the treatment of the sentence.

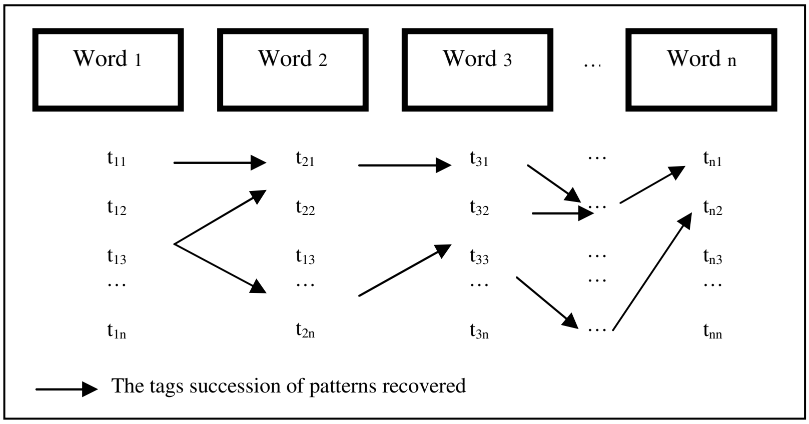

Fig. 2. Example of the progress of sentences patterns exploration

If for a given word no pattern is founded, this tagger agent examines all the training patterns to extract the longest succession tags matching to the tags proposed by the morphological analyzer. When the extraction is made, the tagger joins the segments patterns to the segments previously retained, to form new patterns that are going to serve to the research of the tags of the following words. When all words of the sentence have been treated, and if several candidate patterns were kept as result of the tagging, the tagger chooses the pattern having the highest weight that is calculated from the sum of the initial probabilities of its words' tags. If several patterns have the same weight then it keeps the one that is most frequent in the training corpus.

Voting Agent. After achieving their works, the tagger agents activate the voting agent to decide which tag to assign for a word. We have three cases:

1. If all taggers elect the same tag then this tag is affected to the target word;
2. If the majority and not the totality of taggers agree about a given tag, this tag is assigned to the treated word;

3. If all taggers are in a total disagreement, the voting agent uses heuristics to decide and to choose one and only one tag to assign to this word.

   These heuristics are:

   - The reliance degree in progress: Voting agent considers the tag of the tagger having the higher reliance degree. For each tagger an indicator is provided and is incremented each time the voting process considers its tag in the vote.
   - The reliance degree in historic: In case two or several taggers have the same highest reliance degree, the voting agent sees the historic of every tagger in competition and chooses the one which previously achieved the best tagging accuracy.

Unknown Words Resolution Agent. We have two cases of unknown word:

- If the morphological parser does not propose tags for the treated word:
  - The tagger agent asks the assistance help of the other tagger agents. If one of them solves this problem, it sends to him the found tag.
  - If no tagger could help it, the tagger agent calls the unknown word resolution agent.
- If the morphological parser proposes a set of tags for the target word and the tagger does not find the suitable tag, because of lack of training data :
  - If one of the tagger agents solves this problem before the tagger agent asking help, it sends to him the found tag.
  - Otherwise, the unknown word resolution agent is required:
    - ▪ If the unknown word resolution agent proposes a tag that exists among the set of tags proposed by the morphological analyzer, this tag will be considered as the final tag to assign to the word to tag;
    - ▪ If the unknown word resolution agent proposes a tag which is not among the set of tags proposed by the morphological analyzer, then the tagger accedes to the training data and recovers the most frequent tag among the set of tags proposed by the morphological parser to assign it to this word.

To guess a tag, the unknown words resolution agent works according to this principle:

1. uses the schemes of verbs and personal names as well as the lexical rules to determine the nature of the treated word (noun, personal name, verb…);
2. takes the corresponding tag from a training list containing schemes and their relative tags, if the word to tag has a scheme of a verb or personal name;
   Examples:
   Verb: If a word has the scheme إِفْعَلُوا it can be tagged فعل أمر.
   Personal name: If a word has the scheme فَعْلاَنُ it can be tagged اسم علم مرفوع.
3. applies lexical rules if the word is assumed to be a noun.
   Example: noun starting with الْ and ending with ضمّة ـُ is likelier to have the tag اسم معرّف مرفوع.

5 Experimentations and Results

Our experiment was carried out in two stages: one stage of training during which, a textual corpus containing about **6000** textual forms was manually annotated and probabilities were collected. The second stage is the testing, which consists in using these probabilities to tag a testing text. We tested two different environments. In the first one, we toke the testing text from training corpus. In the second, we chose the testing text out of the training corpus.

For the system evaluation, we used the accuracy rate that is calculated as follows:

$$\text{Tagging accuracy} = \frac{\text{number of correctly tagged words}}{\text{total number of tagged words}} . \quad (6)$$

5.1 Ad Hoc Environment

As shown in the table 3, the probabilistic taggers are not very efficient (maximum accuracy of **93.73%** for simple tags and **95.78%** for composed tags) since they require a big training data. In general, the accuracy for all taggers increases (except for the Trigram tagger), when we manipulate the composed tags. The accuracy of the global system increases as well, and it is due to the diversity of mistakes that taggers provoke. We observe also that the use of macro tags increases significantly the accuracy of the taggers. This is due to the nature of mistakes caused by taggers that confuses tags belonging to the same class of tags.

Table 3. Tagging accuracy in the ad hoc environment

Taggers	**Simple tags (%)**	***Macro* tags (%)**	**Substantives (%)**	**Verbs (%)**	**Particles (%)**	**Composed tags (%)**
Unigram	92.17	94.68	97.33	90.43	99.45	94.42
Bigram	90.47	95.88	93.44	76.33	98.26	92.35
Trigram	93.73	96.84	94.37	87.5	98.85	90.99
HMM	91.52	94.23	96.01	89.88	**100**	95.78
Sentences Patterns	**95.33**	**97.24**	**97.76**	**90.53**	98.88	**95.91**
Global system	**97.54**	**98.64**	**97.77**	**94.79**	**100**	**98.35**

We can also notice that the sentences patterns tagger achieved best results: **95.33%** for simple tags and **95.91%** for composed tags. This reflects the efficiency of this new method.

5.2 Out of the Training Data Environment

In the table below, we can observe that the tagging accuracy of the sentences patterns tagger becomes the weakest (except for composed tags), whereas in the ad hoc environment it was the best. This is because we are in an environment out of training data, and our training data are insufficient for such a method. Therefore, the accuracy rate achieved by this tagger is comprehensible.

However, the accomplished tagging accuracy of the global system, using the simple and composed tags is satisfactory compared to results achieved by the other Arabic tagging systems in a similar environment of experimentation.

Table 4. Tagging accuracy in the out of training data environment

Taggers	**Simple tags (%)**	***Macro* tags (%)**	**Substantives (%)**	**Verbs (%)**	**Particles (%)**	**Composed tags (%)**
Unigram	90.92	**96.14**	97.14	80.87	**96.77**	90.11
Bigram	90.29	95.02	95.79	78.74	96.61	92.05
Trigram	**92.26**	95.41	93.10	**87.61**	96.15	89.62
HMM	91.42	95.4	93.85	86.43	96.66	90.59
Sentences patterns	90.09	94.86	**97.36**	76.11	92.06	**92.38**
Global system	**92.35**	**96.68**	**98.26**	**85.0**	**96.77**	**94.81**

6 Conclusion

Our POS-tagging system is based on a combined approach. The efficiency of this approach was proved by the accuracy generated by the global system. In fact, this accuracy is generally higher than the tagging accuracy of each tagger. The new method of tagging based on sentences patterns gives also satisfactory results and proves to be promising. In spite of the lack of our training data and the ambiguous specificities of the Arabic language, the choices that we adopted enabled us to reach our initially drawn up goals. However, the results can be still ameliorated by improving largely the training data. Considering that the majority of the texts available are without vowel marks, we plan to treat this type of texts in a further work.

References

1. Cutting D., Kupiec J., Pedersen J. And Sibun P.: A practical Part-Of-Speech Tagger. In: proceedings of the Third Conference on Applied Natural Language Processing, (1992) 133–140
2. Schmid H. et Stein A. : Etiquetage morphologique de textes français avec un arbre de décision. Le traitement automatique des langues: Traitements probabilistes et corpus, Vol.36, No. 1-2, (1995) 23–35
3. Ratnaparkhi Adwait: A maximum Entropy Model for part of speech tagger. In: proceedings of the first empirical methods in natural language processing conference, Philadelphia, USA, (1996) 133–142
4. Brill E.: Some Advances in Transformation-based part of speech Tagging. In: proceedings of the 12th national conference on artificial intelligence (1992), 722-727
5. Marshall I.: Choice of Grammatical Word-class without Global Syntactic Analysis: Tagging Words in the LOB Corpus. Computers and the Humanities, No.17, (1983) 139–50

6. Brill E. and Wu J.: Classifier combination for improved lexical disambiguation. In: proceedings of the thirty-sixth ACL and seventeenth COLING, Montréal, Canada, (1998) 191–195,
7. Sjöbergh Jonas: Combining POS-Taggers for improved accuracy on Swedish text. NoDaLiDa, Reykjavik, (2003)
8. Debili F., Achour H., Souissi E. : La langue arabe et l'ordinateur : de l'étiquetage grammatical à la voyellation automatique. Correspondances, No. 71, Institut de recherche sur le Maghreb contemporain, CNRS, Tunis, (2002) 10–28
9. Khoja S.: APT: Arabic Part-of-speech Tagger. In: proceedings of the student workshop at the second meeting of the north American chapter of the Association for computational linguistics (NAACL'01), Carnegie Mellon University, Pennsylvania, (2001) 20–26
10. Zemirli Z. et Khabet S. : TAGGAR : un analyseur morphosyntaxique destiné à la synthèse vocale des textes arabes voyellés. JEP-TALN 2004, Traitement Automatique de l'Arabe, Fès, (2004)
11. Ben Othman C. : De la synthèse lexicographique à la détection et la correction des graphie fautives arabes. Thèse de doctorat, Université de Paris XI, Orsay, (1998)
12. Rajman M. et Chappelier J.C. : Chaînes de Markov cachées. Cours TIDT, Département informatique, Ecole Polytechnique de la Lausanne, (2003)