A Hybrid Morphology-Based POS Tagger for Persian

Mehrnoush Shamsfard[1], Hakimeh Fadaee[1]

[1]NLP Research Laboratory , Faculty of Electrical & Computer Engineering,
Shahid Beheshti University,
Tehran, Iran

E-mail: m-shams@sbu.ac.ir, Ha.Fadaee@mail.sbu.ac.ir

Abstract

In many applications of natural language processing (NLP) grammatically tagged corpora are needed. Thus Part of Speech (POS) Tagging is of high importance in the domain of NLP. Many taggers are designed with different approaches to reach high performance and accuracy. These taggers usually deal with inter-word relations and they make use of lexicons.
In this paper we present a new tagging algorithm with a hybrid approach. This algorithm combines the features of probabilistic and rule-based taggers to tag Persian unknown words. In contrast with many other tagging algorithms this algorithm deals with the internal structure of the words and it does not need any built in knowledge. The introduced tagging algorithm is domain independent because it uses morphological rules. In this algorithm POS tags are assigned to unknown word with a probability which shows the accuracy of the assigned POS tag. Although this tagger is proposed for Persian, it can be adapted to other languages by applying their morphological rules.

1. Introduction

Part of speech tagging is one of the principal issues in natural language processing. Tagged corpora play an important role, in many NLP systems, to simplify the more sophisticated applications. Different approaches are followed to achieve a tagger with higher accuracy and performance. Although the general approach of each tagger could be used in many languages, but there are some specifications in any language that should be considered in designing a tagger. This paper describes the design and implementation of a word learning system which extracts the new words from an input corpus, selects the most probable POS tag for them and adds them to a lexicon.
The tagging algorithm used in this system is base on morphological rules of Persian language and in contrast with many other tagging systems, this system makes no use of any built in knowledge. The introduced algorithm is a combination of statistical and rule-based algorithms. The paper is organized as follows: in second section the general approaches in POS tagging are introduced. Third section describes the pre-processing phase of our tagger. The tagging algorithm and the process of calculating the probabilities are explained in section 4 and finally in section 5, some examples of the functionality of the tagger are presented.

2. Related Works

There exist different tagging algorithms each of which tries to remove the defects of its previous algorithms. Taggers are categorized in some groups with regard to their tagging algorithm.
In this section we note two categorizations proposed by Jurafsky and Megerdoomian. As you can see the general concepts of these two categorizations are the same and even they have categories with the same names.
Jurafsky (Jurafsky & Martin, 2000) divides taggers into 3 categories.

- **Rule-based taggers:** In these taggers the tagging process has two phases. In first phase words are looked up in a dictionary and the found tag or tags are assigned to them. In second phase tagger tries to disambiguate the words with more than one tags. To do so, tagger makes use of some limiting rules which concern syntax.
- **Stochastic taggers:** These taggers use probabilistic methods. They try to assign a tag to the word, which has the most probability by considering the sequence of words it is in.
- **Transformation based taggers:** These taggers have also two phases. In the first, tagger assigns the most probable tag to the word. In second

phase, which is the correction phase, tagger tries to change the wrong tags by following some rules. These rules concern the words adjacent to the considered word.

Megerdoomian (Megerdoomian, 2004, a) categorize tagging algorithms into two groups.

- **Statistical taggers:** These tagging algorithms apply probabilistic methods. They are usually trained by using a tagged corpus. They learn the POS tags of the words and their probabilities from the corpus and also they learn the distributional probability of the word. When these taggers encounter an unknown word they use distributional information of the word to suggest a tag for it. Statistical taggers are of high accuracy but their performance is difficult to improve. Also they need a tagged training corpus which would be unavailable in some languages.
- **Rule-based taggers:** These taggers select the proper tag by using the grammatical and morphological rules. These rules are defined for the tagger and there in no need for tagger to learn them. AS these taggers use the rules defined for them, they are incapable of tagging unknown structures but for the known structures hey usually work accurately.

We follow Megerdoomian's categorization (Megerdoomian, 2004, a). To take the advantages of these two algorithms hybrid taggers are suggested. They combine these two methods to reach higher performance and accuracy. Usually at first rule-base taggers are used to tag the considered corpus, and then statistical taggers are employed to disambiguate the words with more than one tag or unknown words.

The algorithms described above need tagged corpora or a large lexicon. When these taggers encounter an unknown word which is not in the lexicon or the tagged corpora they are incapable of tagging. These algorithms usually concern the interword relations and they have nothing to do with internal word structure.

In many languages like Persian tagged corpora or large lexicons may be unavailable, especially for specific domains, so the algorithms mentioned above may have some shortages. Also Persian is a productive language and it's powerful in word generating especially in new domains. So it's necessary to have an algorithm which has less dependency to tagged corpora or lexicons.

In this paper we present an algorithm which tags the words regarding their internal structure. This algorithm doesn't need a tagged corpus in the process of tagging. In the suggested system a tagged corpus is needed for calculating the probabilities of morphological rules which are used in the process of tagging. After learning these probabilities, the tagger would be capable of tagging unknown words and as the morphological rules are the same in all domains, this tagger is not domain dependant. In the following sections the structure of the proposed word learning system is described.

3. Pre-processing

The first step in forming the lexicon is to extract the words from the corpus. To perform this task, it is necessary to detect the boundaries of words. The word boundaries in Persian are not clear because there exist many words that have more than one parts and in which the parts are separated by white spaces. So some ambiguities arise in detecting the tokens in Persian.

To simplify the process of tokenizing we have considered the white spaces and the punctuation marks as separators. By this assumption, in pre-processing phase, our system extracts the tokens from the corpus and refers them to tagging module for further processes.

There are some rules, concerning writing, that force us to write some parts of the words separately(Megerdoomian, 2004, b). To extract the words correctly from the corpora we should consider these rules. Bellow we mention some of these rules that are applied in our system.

- "می" *(mi)* which appears in the beginning of the verbs and is the sign of present continuous verbs should be written separately: like "می" *(mi)* in "می خوانم" *(mi-xän-am)* (I'm reading). In some texts this rule is not followed and these verbs are written in a single string.
- "**ها**" (hä) which is one of the plural signs in Persian is sometimes written separately: like "**ها**" *(hä)* in "**درخت ها**" *(deraxt-hä)* (The trees).
- "**بی**" *(bi)* which is a prefix in Persian and makes the words semantically negative is usually written separately: like "**بی صدا**" *(bi-sedä)* (voiceless).

4. Tagging

As it was mentioned in section 2 there are different principal algorithms for tagging unknown words. The algorithm suggested in this section is a combination of rule-based and statistical algorithms. There are 4 steps in tagging an unknown word.

1. Detecting the probable affixes in the word
2. Constructing word's parse tree
3. Pruning the parse tree
4. Calculating the truth probability of the remaining derivations

5. Assigning the most probable tag to the word and adding it to the lexicon

In the following subsections each step will be described.

4.1. Detecting Probable Affixes

Persian includes a set of affixes which are used in the structure of the words. Every affix has its own role in the word. These affixes are separated into two categories: Inflectional and derivational affixes.
Inflectional affixes have syntactic roles; they can be usually used with all of the words in a certain syntactic category.
Derivational affixes are used to form new words like "**ناک**" *(näk)* in "**دردناک**" *(dard-näk)* (painful) and "**انه**" *(äne)* in "**آگاهانه**" *(ägahaäne)* (consciously). Derivational affixes usually appear before inflectional ones in word structure.
Affixes can help us to find the syntactic category of the words. So in first step we determine which of these affixes are used in the structure of our word. For example in the word "**کارمندهایش**" *(kärmand-hä-yash)* (his employees), 3 suffixes are used: "**ش**" *(sh)*, "**ها**" *(hä)* and "**مند**" (mand). It is probable that some or all of the affixes found in the word do not have an affix role but are simply a substring of the stem e.g. the word "**آبان**" *(äban)* has "**ان**" *(än)* as a substring which is a known affix in Persian, but it is obvious that it doesn't have affix role here.
In this step system detects all the affixes appeared in the considered word and uses them to parse the word and finds its proper tag. In our system a set of 60 Inflectional and derivational affixes are used.

4.2. Forming Parse tree

There exist two types of morphological rules: derivational and inflectional rules. Derivational rules describe the structure of the words. These rules consist of 3 parts: prefix, stem and suffix and their form is as follows: P1 + R1 + S1→ POS Tag1 in which one of P1 or S1 is usually empty string. It means that if prefix p1 and suffix S2 are adjoined to a stem with POS of R1 the result would be a word of POS Tag1. It should be mentioned that in our system there is no difference between these two types of rules and system uses these rules without knowing their real category.
In our system about 140 morphological rules are defined and used. A number of these derivational and inflectional rules are presented in Table 1 and Table 2 (Kalbasi, 2001; Anvari & Ahmadi givi, 1997; Mogharabi, 1994).

Prefix	Stem	Suffix	Tag	Example
-	Noun	مند	Adj.	قدرتمند *(ghodratmand)* (powerful)
نا *(nä)*	Noun	-	Adj.	ناامید *(näomid)* (hopeless)
بر*(bar)*	Verb	-	Verb	برانگیخت *(barangikht)* (provoked)

Table 1. Derivational rules in Persian

Prefix	Stem	Suffix	Tag	Example
-	Noun	ها *(hä)*	Noun	کتابها *(ketäbhä)* (books)
ب *(b)*	Present stem	ید *(id)*	Imperative verb	بروید *(beravid)* (go)
-	Past stem	م *(m)*	Past Simple Verb	رفتم *(raftam)* (I went)

Table 2. Inflectional rules in Persian

To help the foreigners not familiar with Persian we add similar tables for English derivational and inflectional morphology rules (Table 3 and 4) too.

Prefix	Stem	Suffix	Tag	Example
-	Noun	al	Adj.	hopeless
un	Adj.	-	Adj.	uncomfortable

Table 3. Derivational rules in English

Prefix	Stem	Suffix	Tag	Example
-	Noun	s	Plural Noun	cats
-	verb	ing	Present participle	working

Table 4. Inflectional rules in English

To form parse tree we start our work with outer affixes. For each <prefix, suffix> pair for which, there exists a rule in system, we add a derivation to the parse tree. When related prefix and suffix are separated from the word the remaining part is called "stem". We continue parsing by considering the stem as our new word. The process finishes when no more derivation is possible. When the parse tree is generated, each path from its stem to any leaf shows a derivation for the considered word. The parse tree generated for the word "**انسانیت**" *(ensäniat)*(humanity) is shown in Fig.1.

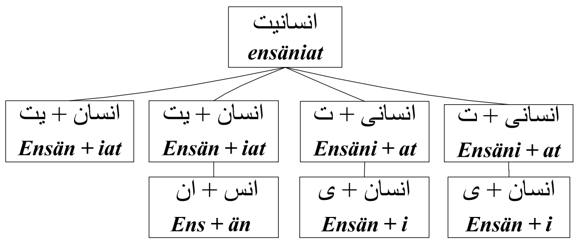

Fig.1. The parse tree of the word "انسانیت" *(ensäniat)* (humanity)

Pruning

By following the process proposed for generating parse tree in section 3.4, we add a number of incorrect derivations to tree. We try to eliminate incorrect derivations by following these steps.

- **Removing useless derivations:** In each node of the parse tree there are some derivations for the stem of the node. We call a derivation "useless" if the tag it suggests is not equal to the tag of its parent's stem. We remove all the nodes caused by useless derivations from the parse tree. It should be mentioned that when a node is removed from the tree all his children will be removed too.
- **Removing repeated affixes:** An affix should not appear in a word more than once. So we remove the nodes containing an affix that exists in one of its parents.
- **Removing affix order violating derivations:** To apply affix order rules we have divided affixes into 9 different categories. This categorization helps us to generalize the rules and simplify system's function. These categories are shown in Table 5.

	Definition	**Examples**
1	Possessive pronouns	م، ت، ش، مان، تان، شان
2	Personal endings	م، ی، د، یم، ید، ند
3	Plural signs	ان، ها، ات، ین، ون
4	Adj. making suffixes	انه، سیر، مند، ان، ناک
5	Adv. making suffixes	انه، اً
6	N. making suffixes	دان، ستان، کده، انه، نا
7	Verbal prefixes	ب، ن، می
8	Clitic	ی نکره
9	Others	

Table 5. Affix categories

A number of rules which concern affix order are as follows.

- The affixes in the first group always appear after the affixes of groups 4, 5 and 6.
- The affixes in the first group always appear after the affixes of the third group.
- The affixes of the second group never appear with the affixes of 3, 4, 5 and 6 groups in a single word.
- The affixes of group 8 always appear after the affixes of groups 3, 4, 5 and 6.

4.3. Calculating Probability

The input of probability calculating module is the pruned parse tree. It's obvious that at last one of the derivations proposed by parse tree would be convenient for the considered word. To find this derivation the system calculates the probability of each derivation and chooses the most probable derivation as result. The tag suggested by this derivation would be assigned to the selected word. To perform this task 3 new parameters are defined and calculated.

P_{Rule}: In section 3.4 we talked about derivation rules of Persian but it should be mentioned that these rules are not certain. Consider the rule R_1 as below:

$$p_i + r_i + s_i \rightarrow t_i \; (1)$$

L indicates, if a prefix P_i and a suffix S_i are added to a word tagged as R_i, the resulted word **would be tagged** as T_i. There are some uncertainties in these rules, i.e. there are some words with the same conditions but they are not tagged as T_i. As we want to use derivation rules for tagging unknown words we should now the truth probability of these rules. This truth probability of rule i is shown by $P_{Rule}(i)$.To calculate $P_{Rule}(i)$ we used a statistical method. For each rule like R_i the process of calculating $P_{Rule}(i)$ is as follows: The corpus is searched for the words with a prefix P_i, a suffix S_i and a stem tagged as R_i. The number of these words is shown as $\boldsymbol{n_{total}}$. In this set of $\boldsymbol{n_{total}}$ words we count the number of words tagged as T_i and show it as $\boldsymbol{n_{target}}$. The truth probability of R_i is equal to: $\boldsymbol{n_{target}/n_{total}}$.

In the definition of $P_{Rule}(i)$ we considered that the stem of the word is tagged as T_i and we are sure about its tag. To find the stem's tag we look it up in the lexicon, but as the lexicon is empty at the beginning of the process, it's possible that we don't find the word's stem in the lexicon. Also the truth probability of the tag assigned to the word could be less than 1. What dose system do when encountering such a word? To solve this problem we need a second parameter.

P_{RulePS}: When the stem of the word is not found in the lexicon the truth probability calculated for the rule is invalid. In this case we define another truth probability which does not depend on the stem of the word but it only concerns the prefix and the suffix of the word and it is shown as $P_{RulePS}(i)$. $P_{RulePS}(i)$ indicates the truth probability of the tag suggested by the rule R_i when our word has a prefix P_i and a suffix S_i and an unknown stem. The value of $P_{RulePS}(i)$ is calculated like $P_{Rule}(i)$ but in this case n_{total} is the number of words with a prefix P_i and a suffix S_i.

P_{branch}: In the parse tree each path from tree's stem to any of the leaves shows a derivation for the considered word. Do all these derivations have the same occurrence probability? The answer is no. There exist differences between these derivations and this difference is caused by affixes separated from the word in each derivation.

In Fig.2 the parse tree of the word "می بینند" *(mi-binand)* (they see) is shown. In this tree, the stem has 3 children, but as it was told these 3 children have different truth probabilities i.e. <prefix, suffix> pair may lead us to the correct derivation but with different probabilities.

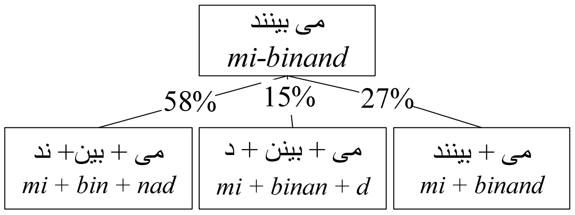

Fig.2. The parse tree of the word "می بینند" *(mi-binand)* (they see)

To apply this difference in our decision making a third parameter is defined, $P_{branch}(i)$, which is assigned to each edge i of the parse tree. The value of $P_{branch}(i)$ depends on the prefix and the suffix of the child node adjacent to edge i. For an edge i in parse tree which leads to a node containing the word W_i with a prefix P_i and a suffix S_i, $P_{branch}(i)$ shows the probability of P_i and S_i to be prefix and suffix for W. So this parameter is calculated for each <prefix, suffix> pair in Persian that can appear in a single word while one of the element of this pair could be an empty string. To calculate the value of this parameter a statistical method is again used. For a pair $<P_i, S_i>$ we search the corpus for all the words starting with P_i and ending with S_i. The number of these words is shown as n_{total}. Within this set of n_{total} words we count the number of words in which P_i is the prefix and S_i is the suffix and show it as n_{target}. $P_{branch}(i)$ for such an edge would be n_{target}/n_{total}.

By using these 3 parameters, now we can calculate the truth probability of the derivations in parse tree. We divide the nodes of parse tree into 4 categories and present a calculating method for each category.

- **Word's stem exists in the lexicon and the node is a leaf:** For these nodes the stem exists in the dictionary. If the truth probability of the tag assigned to word's stem is shown as p_{rDic} then

$$prob(i) = p_{rDic}(i) \times p_{rule}(i) \quad (2)$$

- **Word's stem exists in the lexicon and the node is an intermediate node:** in Fig.3 an intermediate node with its 3 children is shown. For this node $prob(i)$ is equal to the maximum of $p_i \times pb_i$ for all children and $p_{rDic}(i) \times p_{rule}(i)$.

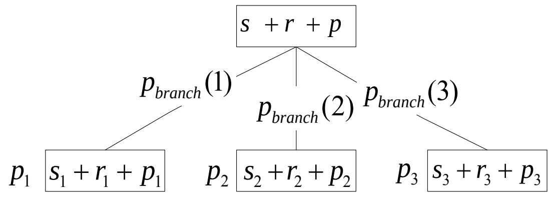

Fig.3. The sub tree for an intermediate node

- **Word's stem is not in the lexicon and the node is a leaf:** For these rules as the stem is not in the lexicon and is unknown for the system $prob(i) = p_{RulePS}(i)$.
- **Word's stem is not in the lexicon and the node is an intermediate node:** For this kind of nodes $prob(i)$ depends only on the children of the node i.e. node $prob(i)$ equals the maximum of $p_i \times pb_i$ for all children.

Now by following a recursive process we can find all the truth probabilities of all derivations in the tree. We start from the stem and try to calculate its truth probability by using the truth probabilities of its children. We continue this recursive process until we reach the leaves.

4.4. Adding the Word to Lexicon

In the previous step we calculated the truth probabilities of all the derivations suggested by parse tree. As we mentioned before we choose the most probable tag for the word. Starting by stem we should select its most probable child and follow this process until we reach a leaf. The traversed path would show the most proper derivation. We enter the word with the probability proposed by this derivation into lexicon. If the word exists in the lexicon with same tag, we replace its probability with the newly calculated if the calculated probability is more that previous otherwise we keep the data in the lexicon, unchanged.

5. Experimental Results

In this section we try to show the abilities of system with some examples. To calculate the morphological rules' probabilities a corpus of 300,000 words was used. This corpus contains some parts of the articles of "Hamshahri" newspaper. The taggset of this system contains 25 tags.

- **Example 1**

Consider the word "**بروند**" *(be-ravand)* (SBJN-To go-3rd pl.**1**) . The final generated parse tree is shown in Fig.4. Pbranch is indicated on any edge. The rules followed in each derivation and their probabilities are specified in Table 6.

This tree has just one level and this fact simplifies the calculations. For each node, the rule probability of the node should be multiplied by its branch probability. As in is shown in the parse tree there are 4 possible derivations for the given word. We calculate the probabilities of these derivations and we choose the most probable one as the proper derivation for the word. For the examples in this section we assume that the lexicon is empty.

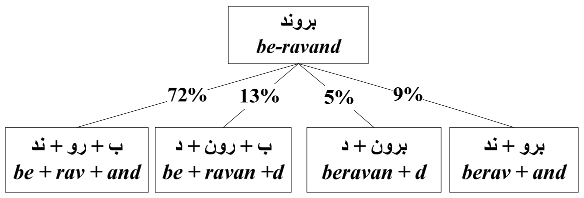

Fig.4. The parse tree of the word "بروند" (be-ravand) (SBJN-To go-3rd pl.)

[1] - "To go" subjunctive, 3rd person plural

no.	Morphological rule	P_{RulePS}	P_{Rule}
1	**ب** + present stem + **د** → Subjunctive verb	65%	100%
2	**ب** + present stem + **ند** → Subjunctive verb	64%	100%
3	present stem + **ند** → Subjunctive verb	42%	100%
4	present stem + **د** → Subjunctive verb	57%	100%

Table 6. Rules used in the parse tree of "بروند" *(be-ravand)* **(**SBJN-To go-3rd pl**.)**

- as the stem "رو" is not in the lexicon we should use P_{RulePS} so the probabilities of this derivation is equal to 64% × 72% = 46%
- as the stem "رون" is not in the lexicon we should use P_{RulePS} so the probabilities of this derivation is equal to 65% × 13% = 8%
- as the stem "برون" is not in the lexicon we should use P_{RulePS} so the probabilities of this derivation is equal to 57% × 5% = 3%
- as the stem "برو" is not in the lexicon we should use P_{RulePS} so the probabilities of this derivation is equal to 42% × 9% = 4%

We can see that the first derivation is the most probable one, so we select this derivation as the proper one. So the word "**بروند**" *(be-ravand)* (SBJN-To go-3rd pl.) is tagged as "Subjunctive verb-3rd person".

It should be considered that all nodes are not decomposable. There are some words that have neither of the affixes as a substring, the nodes containing such words are not decomposable. There exist some other words that have the same affixes as substrings but these affixes are a substring of their stem, so in each node there exists a probability with regards to which the word is indecomposable. For the nodes with no affix as substring this probability equals 1, but for the other words this probability equals $\mathbf{1} - \sum_{i=1}^{n} p_{branch}\,(i)$ in which n is the number of children of the node. This probability should be considered in selecting the proper derivation. If this probability is greater than the target derivation's probability, system prefers to keep the word not decomposed.

- **Example 2**

Consider the word "**کتابهایشان**" *(ketäb-hä-yeshän)* (their books). The final generated parse tree is shown in Fig.5.

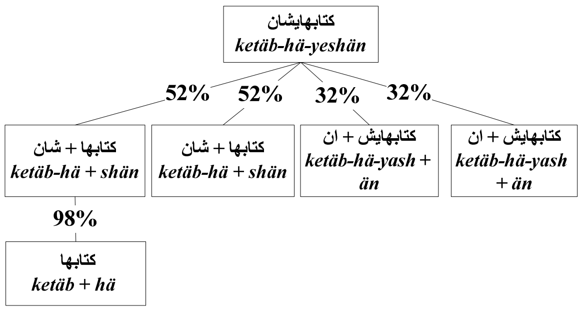

Fig.5. The finale parse tree generated for the word (کتابهایشان) *(ketäb-hä-yeshän)* (their books)

As you can see the branch probabilities are written on the branches. In Table7 the rules used for parsing are indicated.

no.	Morphological rule	P_{RulePS}	P_{Rule}
1	Noun + **شان** → Noun	69%	100%
2	Adjective + **شان** → Adjective	15%	100%
3	Noun + **ان** → Noun	55%	100%
4	Present stem + **ان** → Adjective	48%	72%
5	Noun + **ها** → Noun	99%	100%

Table 7 :Rules used in the parse tree of (کتابهایشان) *(ketäb-hä-yeshän)* (their books)

Now we calculate the probability of each derivation, from left to right, by considering their category regarding their stem.

- This derivation has 2 levels, We start from the lower one. As the stem "**کتاب**" *(ketäb)* (book) is not in the lexicon we should use P_{RulePS}. So the probability of lower derivation is equal to $\mathbf{99}\% \times \mathbf{98}\% = \mathbf{97}\%$. It means that with the probability of 97% the word "**کتابها**" (*ketäb-hä*) (books) has the Noun tag. Now in the parent of this node the probability of the proposed derivation is equal to $\mathbf{97}\% \times \mathbf{69}\% = \mathbf{67}\%$. In which 97% is the probability of "**کتابها**" (*ketäb-hä*) (books) to be tagged as Noun and 69% is the probability of the first rule. Now we should consider the edge connecting stem whit this node, as the probability of this edge is 52%, the probability of the first derivation is equal to $\mathbf{67}\% \times \mathbf{52}\% = \mathbf{35}\%$.
- The second derivation has one level, so calculating the probability is like the previous example. We again use P_{RulePS} and the probability is equal to $\mathbf{15}\% \times \mathbf{52}\% = \mathbf{18}\%$.
- The probability of the third derivation is equal to $\mathbf{55}\% \times \mathbf{32}\% = \mathbf{18}\%$
- The probability of the forth derivation is equal to $\mathbf{48}\% \times \mathbf{32}\% = \mathbf{15}\%$

So we can see that the first derivation is the most probable one and is our target derivation. So we tag the word "**کتابهایشان**" *(ketäb-hä-yeshän)* (their books) as "Noun-Plural-Possessive suffix 6" which means that the word is a plural noun with a possessive suffix adjoined to it. This suffix is the 6th possessive suffix that is related to 3rd person plural.

The tests conducted on the sample inputs show that in about 65% cases, this tagger tags the words correctly. The mistakes made by this tagger are usually in encountering the indecomposable words which have one or some affixes as their substring. Since at the beginning our lexicon is empty, tagger can't understand whether its proposed derivation is right or not. If we enter Persian stems into the lexicon tagger won't make these mistakes. So if this tagger works in combination with stochastic or rule-based taggers and uses their lexicons, it can solve their problems on unknown words and it will have much higher precision.

6. Conclusion

In this paper we have suggested a new tagging algorithm that uses morphological rules to tag the words. As it was described, this algorithm doesn't need any built in knowledge and can tag unknown words. This tagger is domain independent because morphological rules are the same in all the domains. To use morphological rules we extracted them from related references (Kalbasi, 2001; Anvari & Ahmadi givi, 1997; Mogharabi, 1994). and then we have calculated their probabilities by statistical methods.

The proposed algorithm only deals with the internal structure of the words and we don't pay attention to the situation of the word among the other words in the same sentence. It is suggested to apply syntactic rules to reduce the ambiguities of this tagger.

As it was mentioned, at the beginning of the tagger's process our lexicon is empty. Although this eliminates the bottleneck of lexicon acquisition and the need to a preconstructed lexicon, it makes the tagger's work, more difficult. If we enter some entries in the lexicon there would be fewer ambiguities in the tagging. Also this tagging algorithm is proposed to tag unknown words and if it is used in combination with the other algorithms described in section 2 the results would be more accurate.

7. References

Jurafsky,D, H.Martin, J. (2000). Speech and Language Processing. Prentice Hall, New Jersey.

Megerdoomian,K. (2004). Developing a Persian Part of Speech Tagger, First Workshop on Persian Language and Computers, Iran.

Kalbasi,I.(2001), The derivational Structure of Word in Modern Persian, Institute for Humanities and Cultural studies, Tehran, Iran.

Anvari, H., Ahmadi givi, H. (1997). Persian Grammar, 2nd ed., vol. 2, Fatemi, Tehran, Iran.

Megerdoomian, K. (2004). Finite-State Morphological Analysis of Persian, Coling 2004, University of Geneva. August.

Mogharabi, M. (1994). Synthesis in Persian, Tous, Tehran, Iran.