Tokenization as Preprocessing for Arabic Tagging System

Ahmed H. Aliwy

***Abstract*—Tokenization is very important in natural language processing. It can be seen as a preparation stage for all other natural language processing tasks. In this paper we propose a hybrid unsupervised method for Arabic tokenization system, considered as a stand-alone problem. After getting words from sentences by segmentation, we used the author's analyzer to produce all possible tokenizations for each word. Then, written rules and statistical methods are applied to solve the ambiguities. The output is one tokenization for each word. The statistical method was trained using 29k words, manually tokenized (data available from http://www.mimuw.edu.pl\~aliwy) from Al-Watan 2004 corpus (available from http://sites.google.com/site/mouradabbas9/corpora). The final accuracy was 98.83%.**

***Index Terms*—Arabic Tokenization, Arabic segmentation, Arabic tagging.**

I. Introduction

Tokenization is the task of separating out words (morphemes) from running text [1]. It (also sometimes called segmentation) refers to the division of a word into clusters of consecutive morphemes, one of which typically corresponds to the word stem, usually including inflectional morphemes [2]. We can use blanks (white space) to help in this task, but there are hard cases. This definition is for English language but for Arabic the situation is deferent. In discussing tokenization, it is important to remember that there is no single optimal tokenization. What is optimal for IR may not be true for SMT. Also, what is optimal for a specific SMT implementation may not be the same for another [2].

Habash [2] Shows number of different levels of tokenization schemas. It starts from Simple Tokenization which is limited to splitting off punctuation and numbers from words. Then Orthographic Normalization which unified various forms of one letter. Then Decliticization schemes that split off clitics. The last can be done according to stem & affixial morphemes or lemmas & clitics and so on.

In my work, there is clear distinction between segmentation and tokenization. Segmentation is related to splitting running text into sentences (sentence segmentation), into words (word segmentation) and the word to its segments without regards to how this word was constructed. On other hand, tokenization is related to getting token from running text. But in most cases there is overlapping between them. In other words, segmentation is related to splitting all affixes and clitics[1] but tokenization is splitting clitics only with extra retriving the changed or the delted letters results from the inflections. I take the segmentation process as splitting running text into sentences (sentence segmentation), into words (word segmentation) [1] but the tokenization as splitting the words into morphemes.

II. The Whole Pre-Processing System

The whole pre-processing for Arabic tagging system can be consist of Tokenization and Analysing. Figure 1 shows the whole pre-processing for tagging system. After completing all these stages, the final results are Lemma and Clitics with their Features. We must see that the Lemma has ambiguous meaning in Arabic language. For solving this ambiguity we depend on the definition written in [2]. In this paper, I will focus on tokenization only.

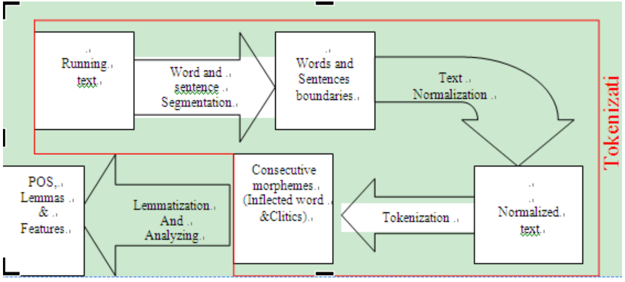

Running text
Word and sentence Segmentation
Words and Sentences boundaries
Text Normalization
Tokenizati
POS, Lemmas & Features
Lemmatization And Analyzing
Consecutive morphemes (Inflected word &Clitics)
Tokenization
Normalized text

Fig. 1. The whole pre-processing task for tagging process. The output is Lemma +Clitics+ Features for each word.

III. Related Work

In some works (MADA+TOKEN Habash 2009 [3], BAMA Buckwalter 2002 [4], AMIRA Mona Diab 2009 [5], Beesley's Xerox Arabic Morphological Analyzer and generator 1996&2001 [6,7], Sakhr's Arabic Morphological Analyzer [8], Khoja's stemmer 1999 [9] and almost morphological Analyzers) this step of natural language processing must be solved inclusively (partially or completely).

Y.Benajiba(2010)[10] presents two segmentation schemes that are morphological segmentation and Arabic TreeBank segmentation and he shows their impact on an important natural language processing task that is mention detection. Experiments on Arabic TreeBank corpus show 98.1% accuracy on morphological segmentation but not Tokenization.

Lee 2003[11] he depends on the form of the word as prefix*-stem-suffix*. The algorithm uses a trigram language model to determine the most probable morpheme sequence for a given input. The language model is initially estimated from a small manually segmented corpus of about 110,000 words. The resulting Arabic word segmentation system achieves around 97% exact match accuracy on a test corpus containing 28,449 word tokens.

Manuscript received May 7, 2012; revised June 10, 2012.

H. Aliwy is with Institute of Informatics, University of Warsaw, Warsaw, Poland (ahmed_7425@yahoo.com; aliwy@mimuw.edu.pl).

[1] See section 7-1 for clitics definition.

The systems of Benajiba [10] and Lee [11] deal with stem rather than lemma. According to Habash [2] stem is not a legal Arabic word form, unlike lemma.

The most related works to our work, in case of tokenization, are AMIRA [5] and MADA+TOKEN [3] but they are packages and the tokenization is not separated task. They used Support Vector Machine (SVM) but Habash [3] used morphological analyzer with SVM. They have accuracy on tokenization 99.12 and 99.21 respectively.

IV. Word and Sentence Segmentation

A. Sentence Segmentation

It is a crucial first step in text processing. Segmentation a text into sentences is generally based on punctuation [1]. In Arabic language, estimating boundary of sentence is relatively simple task approximately same as in English language. The average number of words per sentence is larger than the average in English word which will not affect on segmentation process but on the parsing process. The sentences boundaries and phrase boundaries can be estimated according to Arabic punctuation marks which are {، ,؛ ,. ,: ,... ,؟ ,"" ,- ,[ ] ,=}.

B. Word Segmentation

It is getting words from text. The space is a good separator for this task but it will not work with special cases as compound words. Some compound words are written with a space in the middle even though they are single words. Such cases must be solved at this stage. As example the word “إسلام آباد” “IslAm |bAd” is a name of a city in Pakistan. It means that we must have knowledge base with similar words. With solving this problem, this stage is relatively easy. There is another difficulty, when a few words are attaching together without spaces (i.e. there are not spaces between two words when the first one ends with one of the letters “و” “w”, “د” “d”, “ر” “r”, “ز” “z”, “ذ” “*”). It is formally a mistake, but may happen when dealing with non formal text. I assume this mistake does not occur in the texts.

V. Normalization

Orthographic normalization is a basic task that researchers working on Arabic NLP always apply with a common goal in mind: reducing noise in the data [2]. This is true regardless of the task: preparing parallel text for machine translation, documents for information retrieval or text for language modeling. Normalization can be Tatweel removal (removing Tatweel symbol), Diacritic removal and Letter normalization (variant forms to one form conversion). Figure 2 show letter normalization example.

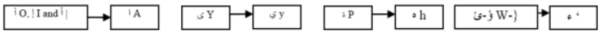


Fig. 2. An example of Arabic letter normalization 2

This normalization will help us in searching or matching process but after this stage, the normalization process will increase the ambiguity in tokenization process. As example if we normalized Ta-Marbuta (P) to Ha (h), the last will be tokenized as pronoun. For this reason, in my work I take normalization as temporary stage for matching, searching and so on.

VI. Arabic Tokenization

Tokenization is a necessary and non-trivial step in natural language processing [12]. It is closely related to the morphological analysis but usually it has been seen as an independent process [13].

Arabic words are often ambiguous in their morphological analysis. This is due to Arabic’s rich system of affixation and clitics and the omission of disambiguating short vowels and other orthographic diacritics in standard orthography (“undiacritized orthography”). On average, a word form in the ATB has about 2 morphological analyses [14].

Arabic word can be in the form [Procltics] + [inflected word] +[Enclitics]. Then, tokenization here is equivalent to word segmentation in Chinese language where Arabic word is as a sentence in Chinese language.

VII. Arabic Word Form

The written Arabic word has special case where the letters are attached together with high possibility of including two categories of Part Of Speech (POS) or more. It leads to problems in stemming and segmentation stages in NLP application as in Tagger. Let’s take the word “وبسيارتهم” “wbsyArth” “and by their car”. Is it a word? How is it constructed? In classical[3] definition of a word, it is a word but, as we can see, it has four POSs.

In this paper I will distinguish constructing of a word from a number of POSs and the inflected word (construction Perfect, imperfect, imperative, mood, person and so on). i.e. we will distinguish Clitics and affixes.

Arabic clitics attach to the inflected base word (see the next section A) in a strict order that can be represented as follows using general class names [2]: [QST+ [CNJ+ [PRT+ [DET+ BASE +PRO]]]] [4]

But in more general way, we can represent the word as:

BASE + Affixes + Clitics ≡ lemma+ morphological features+Clitics

≡ Stem + affixes + Clitics ≡ Inflected word +Clitics

Some researchers didn’t differentiate between affixes and Clitics who are taking the Arabic word generally as (prefixes + stem + suffixes). In my work, I will focus on the form (inflected word + Clitics) where Inflected word is consisting of lemma and morphological features. This will help us encoding word feature and POS without doing an unwanted segmentation. The boundary of inflected word is POS and word feature according to my Tagset in previous work.

[2] I used xml Buckwalter transliteration for transliterate Arabic script.

[3] The word is the letters enclosed by two spaces.

[4] Any sequence written in English is from left to right and the compatible Arabic sequence is from right to left (the first in the left must be first in right and so on) and vice versa.

VIII. Word Clitics

Clitic is a unit whose status lies in between that of an affix and a word. The phonological behaviour of clitics is like affixes; they tend to be short and unaccented but their syntactic behaviour is more like words, often acting as pronouns, articles, conjunction, or verbs [1]. A clitic is a morpheme that has the syntactic characteristics of a word but shows evidence of being phonologically bound to another word [2]. **Clitics** can be **Proclitics** which are precede the word (like a prefix) or **Enclitics** which are follow the word (like a suffix). **Proclitics** can be prefixes of verb, noun, pronoun and particles**.** We can see figure 3 which list approximately all known combination of verbs and Nouns proclitics. Each level can be one or zero occurs except the last level must be existed (the noun or the verb).

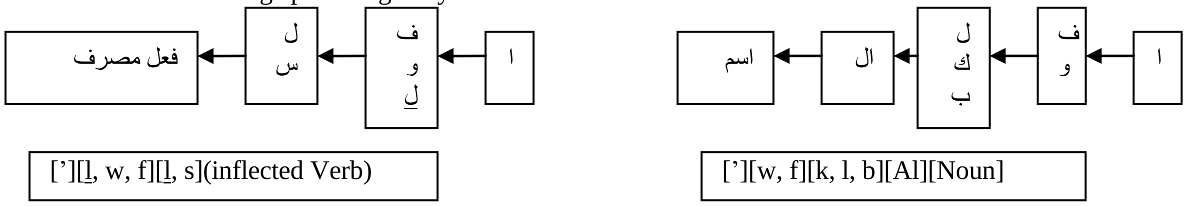

فعل مصرف
ل
س
ف
و
لِ
ا
['][l, w, f][l, s](inflected Verb)
اسم
ال
ل
ك
ب
ف
و
ا
['][w, f][k, l, b][Al][Noun]

Fig. 3. Verb and noun proclitics.

Figure 4 shows cliticization of attached pronouns [5]with particles. Selecting which is the base depends on the priority shown in the figure by number. As example "أفإنهم" "AfInhm" "then, are that they" cliticized as "أ" "A" "are" and "ف" "f" "then" are proclitcs, "إن" "In" "that" is base and "هم" "hm" "they" is Enclitics. The book [2][5] is a good reference for other special cases in cliticization.

The particles can be combined for constructing word but the easy way for dealing with them is by taking these combinations as stop words.

Enclitics can be after verb or noun. The Enclitic "نا" "nA" "we-our" is ambiguous and has two possible roles (either a clitic or an inflection suffix). As example the word "قتلنا" "qtlnA" can be "we killed" or "he killed us" which is affix in the first context and enclitic in the second context.

All enclitics are pronouns and therefore pronouns themselves don't have enclitics. Figure 5 shows all common enclitics for nouns and verbs with their order.

This set of clitics and their order of precedence (summarized here and described also in other papers and books) are the base of our algorithm. Adding a few rules for deleting unwanted combinations of clitics we can get a good segmentation program, as we will see in the implementation section later in this paper.

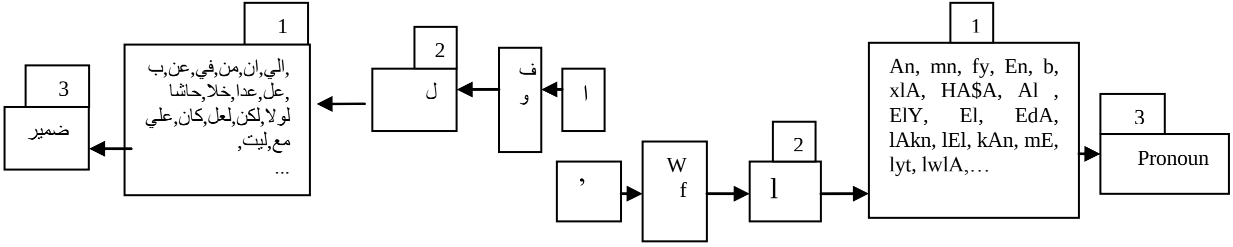


Fig. 4. Proclitics for pronoun and pronoun as enclitics according to the priority number5 of taking the base

(فعل) [ني][نا,كن,كم,كما,ك,هن,ها,هم,هما,ه]
(verb)[ny][nA,kn,kmA,k,hn,hA,hm,hmA,h]

(اسم) [ي,نا,كن,كم,كما,ك,هن,ها,هم,هما,ه]
(Noun) `[nA,kn,kmA,k,hn,hA,hm,hmA,h]`

Fig. 5. Enclitics for noun and verb.

IX. Tokenization Techniques

Habash [16] showed that Tokenization techniques can be simple as regular expression and/or complex as Morphological analyses (Form-based and Functional). But from definition of Morphological analyses [2] we can see that regular expression is part from it. The main classification of Tokenization is Supervised and unsupervised. Unsupervised as (Manual analysis of text and writing custom software [18], unsupervised Language Model Based [11],). Supervised (Annotate the sample corpus with boundary information and use Machine Learning). The other classification is Language Dependent (methods used for one language or class of group of languages, there are many methods in this type) and Language Independent (methods used for any languages).

Arabic language has middle level in segmentation complexity; it is between English (and similar languages) and Chinese (and similar languages), because Arabic language has mixing features. In Arabic words are typically separated by spaces (as in English), but it is possible that an Arabic word is a whole sentence, like in Chinese. Therefore we should use a hybrid method for dealing with segmentation or split the segmentations task into two steps. The interesting thing is that the forms of Arabic word are known which simplify the segmentation of word when compared to Chinese language.

X. Challenges to Arabic Tokenization

There are many challenges to Arabic Tokenization. The Complexity of the morphology together with the under specification of the orthography creates a high degree of

5In Arabic language there are two types of pronouns: attached to a word(us, me..) and separated (I,we…).

6 pages, 48-50

7 the numbers(1,2 and 3) which used in figiure 7 are the priority of taking the base of the word. If one word from 1 exist then it is the base ,if not, then from 2 if not then from 3. note that one of priority at least must be exist.

ambiguity [80]. Some of these ambiguities can be summarized by:

1) Orthography problems result from writing the letter in ambiguous case as in “ى” “Y” and “ي” “y” or unification of some forms of a letter as in “ا” “A”, “أ” “O”, “إ” “I” and so on.

2) Encliticization of a word ending with “ة” “P”:
جمعتهم “jmEthm” “collect them” → هم + جمعت
جمعتهم ““jmEthm” “their Friday” → هم + جمعة

3) Encliticization of word ending with “ى” “Y”:
“مستوى” “mstwY” “level”+ “ك” “k” “your” → “ مستواك” “mstwY” “your level”

4) “نا” “nA” and “ي” “y” are ambiguous and can be either Enclitics or suffixes. See section 6.1.

5) Normalization will add another ambiguity as example normalizing “ة” “P” to “ه” “h” will create wrong enclitics. As example the word “امة” “Amp” “Nation” after normalization will “امه” “Amh” then if we doing the tokenization to the last, it will be “ه + ام” “her mother” but the right tokenization is “امة” “Nation”.

6) Ambiguity results from decliticization of “ل” “l” “ا” “A” and “ال” “Al” “the”.

All these and other ambiguities were solved during tokenization stage by our system. As example the word “حملونا” “HmlwnA” “they rise us” after tokenization will be “حملوا+نا” “HmlwA+nA” where the tokenizer adds the removed letter result from morphological rules. The tokenizer will do the same at the same situation. Another example “زملائي” “zmlA}y” “my colleagues” after tokenization will be “زملاء+ي” “zmlA□ +y”and so on.

Some of ambiguities in POS tagging was solved in tokenization. As example the words “بكتبنا” “bktbnA” “by our books” that “كتبنا” “ktbnA” after tokenization will be “كتب+نا” because of existing the preposition “ب” “b” “by”. The other tokenization is “كتبنا” “ktbnA” “we write” which was neglected by the tokenizer because of the inflected verb can not be appearing after preposition.

7) My approach

We use a hybrid method for tokenization which is a combination of unsupervised method which depends on rules for getting segments, and statistical method for solving ambiguity. My algorithm works as follows:

Task1: As a preparation to the segmentation process, we first compute all Verb, Noun and Pronoun Proclitics and Enclitics storing these combinations in lists. Then, the text is segmented into sentences and the sentences into words according to space and Arabic punctuations as in section 4-1. Segmenting the words into clitics & bases is done by analyzer which produces all possible segments for each word. After this stage every word may have many segmentations.

Task2: Now we remove noise introduced in the first task. We do so by deleting segmentations which produced one letter words with proclitics and enclitics (which is impossible in Arabic) and duplicate segmentations (which may result from segmenting the same word treated once as a Verb and once as a Noun). We also remove segmentations whose inflected word is not in the dictionary (constructed separately from many resources). However, if all produced segmentations of a word should be removed, they are all passed to Task3 for special treatment. Words whose segmentations are not all removed are passed to Task4.

Task3: Because the used dictionary does not cover all words in the language, there are many unknown words whose segmentations are passed from Task2 and must be processed here as out of vocabulary (OOV). These words are manipulated by simple method which is selecting the longest possible combinations of Proclitcs (enclitics), and among them the minimal Proclitics (enclitics) number.

Task4: Because the system produces many segmentations for one word, in order to get one segmentation for each word, we select the segmentation with the least number of segments. If this still does not produce a unique segmentation, we use the same method as in Task3.

Task5: Using statistical estimation to improve resolving ambiguity resulting from Task1. This Task is done in parallel with Tasks 2, 3 and 4. This task is described below in Section XI.

Task6: Filtering by rules to reduce error results from the previous tasks.

For example we add the following rule for differentiating between the word ending with normal Taa (“ت” “t”) or Taa Marbuta (“ة” “p”):

IF *((the base word has Taa AND has enclitics) AND (has proclitic of type preposition OR the previous base is preposition))* ***THEN*** *Change Taa to Taa Marbuta.*

There are many other similar rules used in this task.

XI. Applying Statistical Improvement

Our philosophy of using statistical support is same as using it in POS Tagging system. If we have a sentence: w_1 w_2 … w_n with n words. Let the set of tokenizations of word w_i in this sentence be $\{s_1 \ldots s_j\}$, where j is the number of segmentation[8] of this word. Now we can apply any statistical method, used for tagging, for tokenization. For example if we want to apply HMM for tokenization according to this approach, we will apply the following formula:

$$\hat{s}_{1,n} = \underset{s_{1,n}}{\arg\max} \prod_{i=1}^{n} p(w_i \mid s_i)\, p(s_i \mid s_{i-1}) \qquad (1)$$

So $S_{1,n}$ is the best (maximum probability) Tokenization sequence for sentence of n words. $P(w_i \mid s_i)$ is probability of the ith word given the segmentation s. The segmentation transition probabilities, $P(w_i \mid s_{i-1})$, represent the probability of a segmentation given the previous segmentation. We must see that the number of segmentations change from word to word, and results from an unlimited number of segmentations, while in tagging the set of possible tags is of bounded size.

We have two facts: in our approach, first we used dictionary and rules for tokenization and solving ambiguities. The second is that in a small training corpus, one seldom finds a sequence of more than two words from the sentence under consideration. Therefore bigrams are used, and we do not consider n-grams for n>2. We did not use HMM in our implementation. The Bigrams equation which we used practically is:

$$\hat{s}_i = \arg\max_{s_j} \quad p(w_i \mid s_j)\, p(s_j \mid s_{i-1}) \qquad (2)$$

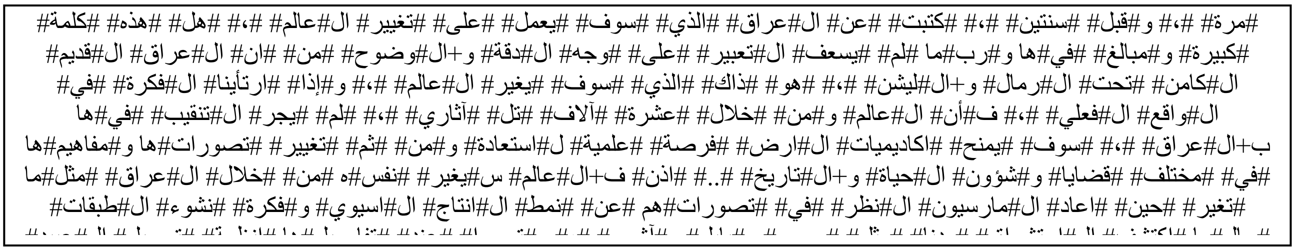
P(w_i | s_j) is probability of ith word given jth segmentation. P(s_j | s_{i-1})is probability of jth segmentation given previous segmentation.

XII. Results

After applying all the previously described simple methods, we got on the following results, in which we used Bigrams on 45 files with size of 29092 tokens: Without statistical support the recall is 0.9877462, Precision is 0. 8617793 and F-measure is 0.920473. Without statistical support (one choice for each word) the accuracy is 0. 9802977. With statistical support (one choice for each word) ten-fold Cross-validate accuracy is 0.9883473. In our tests, tokenizations “#اسرت#ها”[9] “#Asrt#hA” and “#اسرة#ها” “#Asrp#hA” were taken as not match (error). Also the tokenizations “#نرا#ها” “#nrA#hA” and “#نرى#ها” “#nrY#hA” are taken as error. In general, any change to the ending letter of the word resulting from morphology, if it is not compatible with the original letter, is assumed to be an error. Practical tokenized Arabic text and its transliteration are shown in figures 6 and 7 respectively[10]. Comparing with other works, the best known tokenization results have accuracy 99.12% - 99.2 (Diab and Habash respectively) on data set of ATB. They did not solve following problems: in some times they take “AL” as part from word not as clitics leads to decreasing ambiguity between A+L and AL clitics (i.e. increasing accuracy). In most of cases, they did not manipulate changing the letter results from morphology problems. i.e. the last two example in this section is not matter in these works. Their work are data dependent because they used statistical method only but our work is data independent because of using written rules.

XIII. Discussion

We can see that we collect more than one method for solving ambiguity in Tokenization. We introduced very simple and effective methods for making decisions in tokenization. Using dictionary, written rules, selecting longest combination of Proclitcs (enclitics) with minimum Proclitics (enclitics) number with minimum segments number and finally adding statistical decision making. All these methods collectively are applied for getting high accuracy Arabic tokenization system. My approach inclusively solved most of ambiguities in tokenization. The Tokenization were taken as separate task which can be efficient tool for annotation large corpus by correcting the wrong cases manually which leads to improving the next stages in tagging system.



#مرة# #،# و#قبل# #سنتين# #،# #كتبت# #عن# ال#عراق# #الذي# #سوف# #يعمل# #على# #تغيير# ال#عالم# #،# #هل# #هذه# #كلمة#
#كبيرة# و#مبالغ# #في#ها و#رب#ما #لم# #يسعف# ال#تعبير# #على# #وجه# ال#دقة# و+ال#وضوح# #من# #ان# ال#عراق# ال#قديم#
ال#كامن# #تحت# ال#رمال# و+ال#ليشن# #،# #هو# #ذاك# #الذي# #سوف# #يغير# ال#عالم# #،# و#إذا# #ارتأينا# ال#فكرة# #في#
ال#واقع# ال#فعلي# #،# ف#أن# ال#عالم# و#من# #خلال# #عشرة# #آلاف# #تل# #آثاري# #،# #لم# #يجر# ال#تنقيب# #في#ها
ب+ال#عراق# #،# #سوف# #يمنح# #اكاديميات# ال#ارض# #فرصة# #علمية# ل#استعادة# و#من# #ثم# #تغيير# #تصورات#ها و#مفاهيم#ها
#في# #مختلف# #قضايا# و#شؤون# ال#حياة# و+ال#تاريخ# #..# #اذن# ف+ال#عالم# س#يغير# #نفس#ه #من# #خلال# ال#عراق# #مثل#ما
#تغير# #حين# #اعاد# ال#مارسيون# ال#نظر# #في# #تصورات#هم #عن# #نمط# ال#انتاج# ال#اسيوي# و#فكرة# #نشوء# ال#طبقات#

Fig.6. Sample of Arabic tokenized text

#mrp# #,# w#qbl# #sntyn# #,# #ktbt# #En# Al#ErAq# #Al*y# #swf# #yEml# #ElY# #tgyyr# Al#EAlm# #,# #hl# #h*h# #klmp# #kbyrp# w#mbAlg# #fy#hA w#rb#mA #lm# #ysEf# Al#tEbyr# #ElY# #wjh# Al#dqp# w+Al#wDwH# #mn# #An# Al#ErAq# Al#qdym# Al#kAmn# #tHt# Al#rmAl# w+Al#ly$n# #,# #hw# #*Ak# #Al*y# #swf# #ygyr# Al#EAlm# #,# w#I*A# #ArtOynA# Al#fkrp# #fy# Al#wAqE# Al#fEly# #,# f#On# Al#EAlm# w#mn# #xlAl# #E$rp# #|lAf# #tl# #|vAry# #,# #lm# #yjr# Al#tnqyb# #fy#hA b+Al#ErAq# #,# #swf# #ymnH# #AkAdymyAt# Al#ArD# #frSp# #Elmyp# l#AstEAdp# w#mn# #vm# #tgyyr# #tSwrAt#hA w#mfAhym#hA #fy# #mxtlf# #qDAyA# w#$Wwn# Al#HyAp# w+Al#tAryx# #..# #A*n# f+Al#EAlm# s#ygyr# #nfs#h #mn# #xlAl# Al#ErAq# #mvl#mA #tgyr# #Hyn# #AEAd# Al#mArksywn# Al#nZr# #fy# #tSwrAt#hm #En# #nmT# Al#AntAj# Al#Asywy# w#fkrp# #n$w□ # Al#TbqAt# #HAl#mA #Akt$f# Al#Ast$rAq# #mdnA# #mvl# #swmr# w#bAbl# w#|$wr# #,# w#tHrwA# #End# #tfASyl#hA #AnZmp# #tsjyl# Al#Ebyd# w+Al#AjrA□ # w+Al#mwZfyn# w#A$kAl# #tnZym# Al#Eml# w#AdArp# Al#dwlp# #,# w#lw# #kAn# Al#Ast$rAq# #fy# #zmn# #mArks# w#Anjls# #qd# #twSl# #AlY# #Akt$Af# #tlk# Al#mdn# w#dqA}q#hA Al#ywmyp# l#mA# #ktbA# #$y}A# #En# Al#ArD# Al#m$AEp# w#m$klp# Al#bzl# #All*yn# #HAlA# #dwn# #ArtqA□ # Al#mlkyp# Al#frdyp# w#mnEA# #mn# #qyAm# Al#SrAE# Al#Tbqy# #,# w#rb#mA #kAnt# Al#mArksyp# #gyr#hA #fy# Al#nZr# #AlY# Al#$rq# w+Al#grb# #lw# #kAn# Al#Ast$rAq# #fy# Al#mstwY# Al#tfSyly# k#mA# #jA□ ##bEd# #mArks# #.#

Fig. 7. Transliteration of Arabic tokenized text

Acknowledgments

This paper is a part of my PhD dissertation in computer science, concerning Arabic tagging system with a special Tagset. I would like to thank my advisor Prof. Jerzy Tyszkiewicz.

Reference

[1] Daniel Jurafsky and H. James Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, 2000.

[2] Nizar Y. Habash, *Introduction to Arabic Natural Language Processing Synthesis Lectures on Human Language Technologies*. Morgan and Claypool Publishers 2010.

[3] Nizar Habash, “Owen rambow and Ryan Roth: MADA+TOKAN: A toolkit for Arabic tokenization, diacritization, morphological, disambiguation, pos tagging, stemming and lemmatization,” *Proceedings of the 2nd International Conference on Arabic Language Resources and Tools (MEDAR)*, Cairo, Egypt, 2009.

[4] T. Buckwalter, “Buckwalter Arabic morphological analyzer version 1.0.” *Linguistic Data Consortium*, University of Pennsylvania, 2002.

[8] We must see that s1, …, sj are segmentations but not segments. i.e. each one of these segmentation has one or more segments.

[9] Practicaly the tokenized text has format: proclitics#inflectedWord#enclitics. If there are more than one proclitics\enclitics then they are separted by + symbol.

[10] There are 45 tokenized files freely available on my website: http://www.mimuw.edu.pl/~aliwy

[5] M. Diab, “Second generation tools (AMIRA 2.0): Fast and robust tokenization, pos tagging, and base phrase chunking,” *In Proceedings of 2nd International Conference on Arabic Language Resources and Tools (MEDAR),* Cairo, Egypt, April 2009.

[6] K. Beesley, “Arabic finite-state morphological analysis and generation,” *In Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*. Copenhagen, Denmark ,1996, volume 1: 89–94.

[7] K. Beesley, “Finite-state morphological analysis and generation of Arabic at Xerox research: status and plans in,” *Proceedings of the Arabic Language Processing: Status and Prospect, 39th Annual Meeting of the Association for Computational Linguistics*. Toulouse, France 2001.

[8] Sakhr. Software, Arabic Morphological Analyzer. [Online]. Available: http://www.sakhr.com.

[9] S. Khoja and R. Garside, Stemming Arabic Text, Lancaster, UK, computing department, Lancaster university, 1999. [Online]. Available:http://www.comp.lancs.ac.uk/computing/users/khoja/stemmer.ps.

[10] Y. Benajiba and I. Zitouni, “Arabic word segmentation for better unit of analysis,” *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), 2010.

[11] Y-S Lee, K. Papineni, S. Roukos, O. Emam, and H. Hassan, “Language model based Arabic word segmentation,” *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*. Sapporo, Japan 2003, 399-406.

[12] S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python*. O’Reilly Media, 2009.

[13] J-P Chanod and P. Tapanainen, “A non-deterministic tokeniser for finite-state parsing,” *ECAI 96. 12th European Conference on Artificial Intelligence*, 1996.

[14] N. Habash and O. Rambow, “Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop,” *Proceedings of the 43rd Annual Meeting of the ACL*. 2005: 573–580.

[15] J. Olive, C. Christianson, and J. McCary (Editors). *Handbook of Natural Language Processing and Machine Translation.* DARPA Global Autonomous Language Exploitation, 2011. Springer Book.

[16] N. Habash and F. Sadat, “Arabic preprocessing schemes for statistical machine translation,” *In the Proceedings of Human Language Technology Conference*. North American Chapter of the Association for Computational Linguistics (HLT/NAACL), 2006.