**Arabic Language Text Classification**

**Using Dependency Syntax-Based Feature Selection**

Yannis Haralambous\(^1\), Yassir Elidrissi\(^1\) and Philippe Lenca\(^1\)

---

**Abstract**—We study the performance of Arabic text classification combining various techniques: (a) tfidf vs. dependency syntax, for feature selection and weighting; (b) class association rules vs. support vector machines, for classification. The Arabic text is used in two forms: rootified and lightly stemmed. The results we obtain show that lightly stemmed text leads to better performance than rootified text; that class association rules are better suited for small feature sets obtained by dependency syntax constraints; and, finally, that support vector machines are better suited for large feature sets based on morphological feature selection criteria.

---

**I. INTRODUCTION**

Text classification is an important Natural Language Processing task in the current era of big data, and even more so for Arabic, the fifth language of the world, in terms of numbers of native speakers (according to Wikipedia). It is a supervised classification task, where documents are classified into predefined classes. It can be subdivided in four tasks: pre-processing (where the corpus documents are prepared, text is lemmatized or stemmed, stop-words are removed, morphosyntactic or semantic information is added, etc. so that at the end we have not words but “features”), feature selection (where we select the most relevant features, in order to increase pertinence and decrease memory and CPU cost), feature ranking (where we establish a weight between the features which will be used by the classifier, this step is also called “feature ranking” when the weight is simply the rank of the feature) and finally classification per se.

Feature selection and feature weighting are sometimes merged into a single operation, for example one may use tfidf, which is, strictly speaking, a feature weighting method, for feature selection, by sorting features in decreasing order and taking the most highly ranked. Nevertheless, feature selection and weighting are clearly independent of the classification operation and one can combine feature selection/weighting and classification methods freely, in search of optimal solutions.

**A. Arabic Text Classification**

Although quite young, the literature on Arabic text classification is very prolific, starting with the influential works by Sawaf et al. [50] in 2001, El-Kouri et al. [23] in 2004, Khreisat [37] and Syiam et al. [51] in 2006, El-Halees [21] and Mesleh [42] in 2007, Al-Harbi et al. [2] and Thabet et al. [54] in 2008. In the last decade, more than seventy papers have been written on the subject. After a peek of publications in 2009 and 2011, the situation is now stable and one observes 4–6 publications per year on the topic.

Let us see, task by task, how the issue of Arabic text classification has evolved over time.

1) **Pre-processing Task**: The task of pre-processing is obviously more difficult for Arabic than for English, because of the complexity of Arabic morphology and the lack of complete vowelization of standard Arabic texts. In the early days, some authors ([23], [19]) have considered using roots, while others ([50], [37], [6], [38]) have evacuated the problem by using n-grams of letters as features, instead of words. In the meantime there has been progress on the issue of lemmatization/root extraction/stemming of Arabic ([3], [36], [10], [16], [39], [18], [17], [47], [7], [55], [29], [45]) so that researchers working on Arabic text classification were able to use increasingly efficient stemmers (the best known being the ones by Khoja [36] and by Darwish [16]).

Papers [51], [48], [49], [46] and [12] compare and evaluate various stemming methods. Also [27] attempts an hybrid approach (combining rule-based and statistical stemming), and [35] goes one step farther by replacing words not by roots or stems, but by concepts taken from Arabic WordNet [24].

2) **Feature Selection and Weighting Task**: As for these tasks, most authors have used tfidf, sometimes compared to other methods like boolean ranking (“a word belongs to a document or it does not”), tf, log(1 + tf) ([54]) and LTC (a variant of tfidf, cf. [8]). A limited number of authors ([34], [51], [42], [49], [5], [43], [8], [33]) adopt the methodology of distinguishing between feature selection and feature weighting. They use and combine IG (information gain), \(\chi^2\), NGL (Ng, Goh & Low), GSS (Galavotti, Sebastiani & Simi), OR (odds ratio) and/or MI (mutual information) for the former, and mostly tfidf for the latter. [56] uses LDA (Latent Dirichlet Allocation) for feature selection (and tfidf for feature weighting). The most synthetic work is [43], which compares 17 different feature selection methods.

Finally, the authors of [8] consider feature selection as a decision problem and attempt an aggregation of feature selection methods (nevertheless, their conclusion is that “combining two or three feature selection methods showed [only] insignificant improvement in classification accuracy”).

3) **Classification**: Concerning the last task, namely classification per se, we encounter a great variety of approaches. The four most frequent classifiers used are SVM, Naive Bayes (as well as its variants, cf. [5] and [33]), kNN and neural networks.
Often several classifiers are evaluated on the same corpus and compared.

Finally, and this brings us closer to our work, a small number of authors ([9], [11], [4], [53], [25], [26]) use rule-based systems, be it decision trees (C4.5, ID3) or class association rules (Apriori, CHARM). The most synthetic papers are [22], [27], [48], [31], [56], which compare various algorithms of quite different nature, on the same corpora.

B. Our Approach

As far as we can tell, no author uses morphosyntactic information for feature selection, and this is the main contribution of our paper. We show that selecting and weighting features by the syntactic role of words, combined with their POS tags is a more efficient feature selection and weighting method than tfidf.

The syntax theory we use is dependency syntax [52], [41], [30], and it has the advantage of representing a sentence by a tree entirely built of words of the sentence (i.e., there are no intermediate nodes as in constituency syntax, where we have nodes such as NP “nominal phrase,” VP “verbal phrase,” etc.). In dependency syntax theory, every sentence contains a “most important word” called head, and this is an obvious first candidate when we wish to represent a sentence by a single word/feature.

It turns out, as we will see, that selecting as features the head combined with nouns at distance 1, 2 or 3 of the head is almost as discriminative as selecting the complete set of nouns of the sentence. To establish this claim, we evaluate several feature selection strategies, starting by selecting only the heads, then nouns at given distances from the head, etc. We show that for class association rules based on small feature sets, dependency syntax feature selection is significantly more efficient than tfidf.

It should be noted that since dependency syntax operates on the sentence level, we will—as in [30]—restrict ourselves to the sentence level for all calculations. Once classes for all sentences have been predicted, we use a weighted (over the total confidence) majority vote to predict the class of a document. Therefore, tfidf will also be calculated on the sentence level and classifiers (CARs or SVMs) will predict classes for sentences; the document class will be predicted afterwords, as described in [30, Alg. 2]. This explains that our results are slightly inferior to those of [25, Tab. III], after using \( \rho \) to rank documents, we kept the 1,000 first ones for each class.

Finally, the corpus segmentation into sentences has been manually verified, and documents inappropriate for further analysis (poetry, dialectal texts, corrupted documents) have been removed (and replaced by other documents properly ranked).

B. Word Stemming and Rootification

Having no access to a real Arabic lemmatizer (in the sense of an algorithm replacing verb forms by the 3rd masculine person of the singular present indicative, nouns by the 1st person of the singular nominative and adjectives by the 1st person of the singular nominative masculine), we have chosen to use and compare two extreme approaches: light stemming and rootification.

1) Light Stemming: We have separated clitics from stems using the Stanford Word Segmenter [44]. Besides being, in essence, a “very light” stemming method, this step was necessary to be able to use the Stanford Parser [28] to obtain dependencies, since it operates solely on segmented text.

Features we obtained by adding to each stem a simplified version of the POS tag given by the Stanford Parser: ‘n’ for nouns, ‘np’ for proper nouns, ‘v’ for verbs, and ‘x’ for all other cases.

2) Rootification: Besides raw text, the Kalimat corpus includes also morphological information for each word, produced by the Alkhali morphological analyzer [13]. We have kept the root associated to each word (except for proper nouns, that is those having the Stanford tags NNP

| Culture       | words/line | words/doc | lines/doc |
|---------------|------------|-----------|-----------|
| Kalimat       | 26.46      | 541.54    | 20.47     |
| Local news    | 46.91      | 361.73    | 7.71      |
| International | 27.13      | 499.38    | 18.40     |
| Sports        | 35.42      | 325.15    | 9.18      |
| Religion      | 39.04      | 482.99    | 12.37     |
| Economy       | 33.83      | 457.12    | 13.51     |

The selection of these documents among the 20,291 documents provided by Kalimat, has been based on the following criteria:

1) to have short sentences: let \( #p \) be the number of groups of periods, exclamation marks and question marks, divided by the number of words;
2) to have simple sentences, that is sentences with a minimum of connectives tagged WP, WRB WDT (mostly words)
3) to have large texts: let \( #w \) be the number of words.

We aggregated these three parameters into a single one:

\[
\rho = \frac{#p \cdot \log(#w)}{#c}
\]

where we use a logarithm for the number of words criterion in order to limit its strength relatively to the other two criteria. After using \( \rho \) to rank documents, we kept the 1,000 first ones for each class.

Finally, the corpus segmentation into sentences has been manually verified, and documents inappropriate for further analysis (poetry, dialectal texts, corrupted documents) have been removed (and replaced by other documents properly ranked).

II. CORPUS

A. Selection of Documents

We extracted, 6,000 documents from the Kalimat corpus [20]. These documents belong to the following 6 classes (1,000 documents each):
or DTNNP) and then, to obtain features, combined each root (preceded by the √ symbol, to distinguish a root from a word with exactly the same letters) with a simplified POS tag, as in the case of stemming described above. We call this process rootification.

Our corpus Kalimat corpus contains 3,689 distinct roots. Among them, 1,441 appear in all classes, 407 in exactly five classes, 325 in exactly four classes, 375 in exactly three classes, 483 in exactly two classes, and 658 in a single class.

In the following table we display the number of roots unique to each class (on the diagonal) and the number of roots appearing in exactly two classes (and appearing more frequently in the class of the line than on the class of the column). The root under the number is the most frequent one for each case.

| CUL | LOC | INT | SPO | REL | ECO |
|-----|-----|-----|-----|-----|-----|
| 294 | 132 | 21  | 8   | 7   | 29  |
| 35  | 21  | 60  | 29  | 16  | 20  |
| 30  | 3   | 16  | 16  | 16  | 16  |
| 132 | 3   | 16  | 16  | 16  | 16  |
| 25  | 13  | 13  | 13  | 13  | 13  |
| 29  | 13  | 13  | 13  | 13  | 13  |
| 20  | 13  | 13  | 13  | 13  | 13  |

For example, there are 161 roots appearing only in class Religion and the most frequent one is نجس√، وجهم√؛ similarly there are 235 (= 132 + 103) roots appearing exactly in classes Religion and Culture, 132 out of which appear more frequently in class Religion and 103 more frequently in class Culture. The more frequent ones in these two cases are resp. نجس√ و وجهم√.

### III. FEATURE SELECTION AND WEIGHTING METHODS

Two methods are used for feature selection and weighting: tfidf and dependency grammar properties.

#### A. tfidf

The legacy feature weighting method tfidf is defined as the ratio of the frequency of a word in a given document, divided by the number of documents in which it occurs. To first weight and then select the features by tfidf, we take for each sentence $S$ of corpus $C$ the (stemmed or rootified) words $w \in S$, and apply the following formula:

$$\text{tfidf}_S(w) = \frac{\#(\tau(w) \in \tau(S))}{\#S} \log \frac{\#(\{S \in C \mid \tau(w) \in \tau(S)\})}{\#(\{S \in C \mid \tau(w) \in \tau(S)\})}$$

where $\tau$ is either stemming or rootification and $\#$ denotes the cardinal of a set (the precise algorithm for calculating tfidf is given in [30, Alg. 4]). Then we rank words according to tfidf value and use the $N$ highest ranked as features for a given sentence.

Here is an example: for the sentence

Here are the features that the system has calculated:

| n:العربية | np:العربية | x:العربية |
|------------|------------|------------|
| 0.259      | 0.249      | 0.226      |
| 0.156      | 0.151      | 0.150      |
| 0.218      | 0.140      | 0.137      |
| 0.198      | 0.134      | 0.157      |

The reader may observe that some words (like النابض) indeed characterize the sentence, but others (like ست (masculine number six), or ا retain without the ا) are rather misleading.

We present in § V-A the results of our text classification algorithm using tfidf as feature selection method.

#### B. Dependency grammar properties

Dependency grammar [52], [41] is a syntactic theory, alternative to the very popular phrase-structure analysis [14]. In phrase-structure syntax, trees are built by grouping words into “phrases” (with the use of intermediate nodes NP, VP, etc.), so that the root of the tree represents the entire sentence and its leaves are the actual words. In dependency grammar, trees are built using solely words as nodes (without introducing any additional “abstract” nodes). A single word in every sentence becomes the head of the tree. An oriented edge between two words is a dependency and is tagged by a representation of some (syntactic, morphological, semantic, prosodic, etc.) relation between the words. For example in the sentence “محمد يعني مفتاحًا للفانة.”, the verb “يعني” is the head of the sentence and we have the following five dependencies:

[Diagram of dependencies]

where tags nsubj, dobj, prep, dep, etc. denote “noun subject,” “direct object,” “prepositional object” and “preposition,” and

$^1$Kalimat gives an erroneous root here: بطريرك, being a foreign word, there should be either no root or, if the word is considered as arabified, the pseudo-root بطريرك.
NLP, IN, NN, VBP are Stanford POS tags for “proper noun,” “preposition,” “noun” and “imperfect verb.”

To calculate dependencies in our corpus, we used first the Stanford Word Segmenter [44] and then the Stanford Parser [28] to parse the corpus, and then merged the results with word roots provided by Kalimat.

We claim that dependencies provide better features for classification than tfidf, since the latter is based purely on frequencies of words while the former takes advantage of morphosyntactic properties.

To compare with the tfidf method, in Fig. 3, the reader can see the complete dependency tree of the sentence of previous section. Indeed,

- the head of the sentence is َنَشِرُ (word نشرت (word نشرت which has four dependencies:
  - the proper noun َنَمْتَ (word نامته
  - the word َنَمْمَدُ (word نمتد which initiates the following chain of dependencies: َنَمْمَدُ (word نمتد
  - the word َنَمْمَدُ (word نمتد
  - the word َنَمْمَدُ (word نمتد

In these dependency chains, one finds َنَمْمَدُ (word نمتد and َنَمْمَدُ (word نمتد and َنَمْمَدُ (word نمتد and َنَمْمَدُ (word نمتد which seem interesting. It should be noticed that these words have been put forward without any frequency-based calculation and solely based on syntactic criteria.

We present in § V-B the results of our text classification algorithm using dependency grammar properties in the feature selection process.

IV. CLASSIFICATION METHODS

A. Support Vector Machines

Support vector machines (SVMs, cf. [15]) are a nowadays very widespread classification method. They are based on separation of predicted classes by hyperplanes in high dimensional data spaces and provide very good results in general. We use them (in fact, the SVMmulticlass implementation by Thorsten Joachims [32]) on our corpus (rootified and/or the values of some global parameters—one cannot directly modify their behavior.

The disadvantage of SVMs is that they are “black boxes,” i.e., it is not possible for the final user to find out why a given result has been obtained, and—besides changing the training corpus and/or the values of some global parameters—one cannot directly modify their behavior.

B. Class Association Rules

There is another classification method, namely association rules [1], which can be used in our context. This method has the advantage that the user can see the rules used for classification and edit them in order to adapt them to a given situation. Here is a formal definition of association rules:

\[
\text{Let } I \text{ be a set of objects called items and } C \text{ a set of classes. A transaction } T \text{ is a pair } \{(i_1, \ldots, i_n), c\}, \text{ where } \{i_1, \ldots, i_n\} \subseteq I \text{ and } c \in C. \text{ We denote by } T \text{ the set of transactions, by } \text{items}(T) \text{ the set of items (or “itemset”) of } T \text{ and by class}(T) \text{ the class of } T.
\]

\[
\text{Let } I \text{ be an itemset. The support of } I \text{ is defined by }\]

\[
\text{supp}(I) := \frac{\#\{T \in \text{items}(T)\}}{\#I}.
\]

\[
\text{Let } \sigma \in [0, 1] \text{ be a value called minimum support. An itemset } I \text{ is called frequent if its support exceeds } \sigma.
\]

\[
\text{The confidence of a transaction } t \text{ is defined as } \text{conf}(t) := \frac{\#\{T \in \text{items}(T) \setminus \text{class}(T) \mid \text{class}(T) = c\}}{\#\{T \in \text{items}(T) \mid \text{class}(T) = c\}}.
\]

\[
\text{Let } \kappa \in [0, 1] \text{ be a value called minimum confidence. A class association rule (or “CAR”) } r = \{(i_1, \ldots, i_n), c\} \text{ [40] is a transaction with frequent itemset and confidence exceeding } \kappa.
\]

To classify text with CARs, we consider words as being items, sentences as being itemsets and pairs (sentence, class) as being transactions. The advantage of this technique is that CARs can be easily understood—and hence potentially improved—by the user. Once the classifier is trained, to classify a new sentence we first find all CARs whose items are contained in the sentence, and then use an aggregation technique to choose a predominant class among those of the CARs we found. Finally, from classes predicted for sentences we obtain a class prediction for an entire document, by applying a weighted majority vote technique (cf. [30, Alg. 2]).

We evaluate the classifier by using 10-fold cross validation to obtain average values of recall, precision and F-measure. Comparing rule-based classification methods is tricky because one can always increase F-measure performance by increasing the number of rules, which results in overfitting. To avoid this phenomenon and compare methods in a fair
way, we have chosen to fix the number \( N_R \) of rules and find values of minimal support and confidence so that F-measure is maximal under this constraint. (See [30, §2] for more details on the algorithms used.) In Fig. 1 the reader can see F-measure as a function of the number of CARs, obtained by varying support while keeping confidence fixed. At \( N_R = 10,000 \), the CPU charge is not yet very important and F-measure remains relatively stable (in this example, increasing \( N \) from 10,000 to 200,000, increases F-measure by only 0.55%, while processing time is multiplied by a factor of 17). This is why we have chosen to fix the number of rules to \( N_R = 10,000 \) for all subsequent calculations.

Here is an example of the thirty first (by order of decreasing support) CARs we get by using dependency grammar properties:

1. religion \( \leftarrow \) n: \( \text{i} \), (fatwa), 0.173%
2. culture \( \leftarrow \) n: \( \text{المصلح} \), 0.145%
3. economy \( \leftarrow \) n: \( \text{أولك} \), 0.142%
4. economy \( \leftarrow \) n: \( \text{بوروندو} \), \( \text{سيم} \), 0.141%
5. sports \( \leftarrow \) n: \( \text{منتخب} \), 0.141%
6. sports \( \leftarrow \) n: \( \text{البارة} \), 0.135%
7. economy \( \leftarrow \) n: \( \text{بوروندو} \), \( \text{عمان} \), \( \text{سيم} \), 0.094%
8. sports \( \leftarrow \) n: \( \text{سيم} \), \( \text{رجل} \), 0.081%
9. economy \( \leftarrow \) n: \( \text{بوروندو} \), \( \text{عمان} \), \( \text{سيم} \), 0.076%
10. economy \( \leftarrow \) n: \( \text{عمان} \), \( \text{بيضة} \), 0.076%
11. economy \( \leftarrow \) n: \( \text{الإلفاتات} \), 0.076%
12. economy \( \leftarrow \) n: \( \text{الإلفاتات} \), 0.076%
13. economy \( \leftarrow \) n: \( \text{الإلفاتات} \), 0.062%
14. economy \( \leftarrow \) n: \( \text{سيم} \), \( \text{بيضة} \), \( \text{الإلفاتات} \), 0.060%
15. economy \( \leftarrow \) n: \( \text{بيضة} \), \( \text{الإلفاتات} \), 0.060%
16. economy \( \leftarrow \) v: \( \text{فصل} \), \( \text{سيم} \), \( \text{رجل} \), 0.058%
17. religion \( \leftarrow \) v: \( \text{يجب} \), \( \text{رجل} \), 0.058%
18. sports \( \leftarrow \) n: \( \text{مانشستر} \), 0.058%
19. economy \( \leftarrow \) n: \( \text{بوروندو} \), \( \text{قل} \), \( \text{سيم} \), \( \text{رجل} \), 0.054%
20. economy \( \leftarrow \) n: \( \text{سيم} \), \( \text{رجل} \), \( \text{رجل} \), 0.051%
21. economy \( \leftarrow \) n: \( \text{سيم} \), \( \text{رجل} \), \( \text{رجل} \), 0.050%
22. economy \( \leftarrow \) n: \( \text{سيم} \), \( \text{فقر} \), \( \text{بيضة} \), 0.048%
23. local \( \leftarrow \) v: \( \text{قل} \), \( \text{سيم} \), \( \text{رجل} \), 0.048%
24. economy \( \leftarrow \) n: \( \text{فقر} \), \( \text{بيضة} \), 0.048%
25. sports \( \leftarrow \) n: \( \text{رژال} \), 0.047%
26. sports \( \leftarrow \) n: \( \text{برانش} \), 0.047%
27. sports \( \leftarrow \) n: \( \text{برانش} \), 0.044%
28. economy \( \leftarrow \) n: \( \text{سيم} \), \( \text{سهم} \), \( \text{بيضة} \), 0.043%
29. economy \( \leftarrow \) n: \( \text{سيم} \), \( \text{رجل} \), \( \text{رجل} \), 0.043%
30. intern \( \leftarrow \) n: \( \text{قطر} \), \( \text{قطر} \), 0.043%

The first column displays the rule consequent (the predicted class) while the third column displays the items of the rule, and the fourth column the rule’s support (confidence is 100% for these rules).

The reader can observe the pertinence of these rules:

1. rules 1, 2, 5, 10, 11, 18, 19, 25, 26, 27, 30 contain proper nouns: OPEC, Purnomo Yusgiantoro (former president-secretary general of OPEC), Oman (the data of the Kalimat corpus originate from an Omani newspaper) for the Economy category, Manchester United, Arsenal and Real Madrid for the Sports category, Bush for the International news category;
2. many Economy rules contain the words ريال (currency of Oman: \( 1,000 \) ريال = ١٠٠٠ ريال) (currency), \( \text{بيضة} \) (stock market share) and \( \text{سهم} \) (rate);
3. Religion rules contain important words such as احكام (fatwa), (provisions for reading the Koran, etc.), and roots such as 
   
   & أحكام 
   & ج educación 

   & (guardian), 
   & رجل (man).

V. RESULTS

A. Results Based on tfidf Feature Selection and Weighting

On Fig. 2 the reader can see the results (in terms of F-measure) of using CAR and SVM classifiers on features selected by tfidf. More precisely, we ranked (rootified or stemmed) words in each sentence by tfidf value and kept \( N \) of them as features for the given sentence. The figure shows F-measure as a function of \( N \), for values between 1 and 20. In the case of rootified data, the reader can observe that for small values of \( N \) (\( N \leq 5 \)) CARs give better results than SVM, that CARs stabilize around a F-measure of 67% and that SVMs rise until 72%. In the case of stemmed data, both CARs and SVMs give similar results and attain F-measures of 65–67%. At around \( N = 20 \) we have a small decrease of performance, probably an effect of semantic noise due to less relevant words.

B. Results Based on Morphosyntactic Properties

In this section we investigate feature selection strategies based on morphosyntactic features. We will compare them with tfidf methods of similar transaction size.

Our approach will be the following: we will select features (either rootified or stemmed) by taking:

1. only the head of the dependency tree (Strategy I);
2. only nouns neighboring the head (Strategy II);
3. the head plus nouns at distance 1 from the head (Strategy III);
4. the head plus nouns at distance \( \leq 2 \) from the head (Strategy III).
5) the head plus nouns at distance $\leq 3$ from the head (Strategy III$_3$);
6) the head and all nouns of the sentence (Strategy IV);
7) the head, all nouns and all verbs of the sentence (Strategy IV$'$).

The reader can see an illustration of the application of these strategies to the example sentence of the previous section, in Fig. 3: boxes are nouns, blue boxes are those taken by Strategies II–III$_3$, dashed boxes show the limits of strategies. Strategy IV will consider all nouns, IV$'$ will consider all nouns and verbs of the sentence.

1) Strategy I: Our first strategy will be to keep only the head (in the sense of dependency grammar) of each sentence, provided it is a noun (which happens in 5.12% of cases), proper noun (in 2.15% of cases), or verb (in 81.1% of cases).

On Fig. 4 the reader can see the results obtained by applying this strategy to rootified (above) and stemmed (below) data.

Globally these results are rather bad and show that the head alone is not a good feature to select. The results obtained from stemmed data are slightly better than those from rootified data. There are some interesting outliers: for example, the recall of International news is quite high (in the case of the CAR classifier), they are probably easy to detect because of the many foreign proper nouns. Also very high is the precision of Sports (again in the case of the CAR classifier only), and since heads are mostly verbs, we can deduce from this result that sport verbs allow identification of sport text with high precision. The same phenomenon (in a slightly lesser degree) can be observed for Religion.

These three cases there is an enormous difference between the performance of CARs and SVMs.

The rules generated in this strategy have a transaction size of one (!), in other words there are of the form $i \rightarrow c$ where $i$ is a single item and $c$ is the consequent (the predicted class). This is an important issue because CPU charge increases exponentially with respect to transaction size.

Before we move to the next strategy, let us compare performances of Strategy I and of tfidf feature selection. To
be fair, we must compare performance while keeping the same transaction size. For tfidf, a transaction size of 1 is obtained for \( N = 1 \), and the F-measure in this case is of 57.86% for rootified data and of 48.81% for stemmed data. It follows that Strategy I is slightly better than tfidf feature selection for stemmed data, but significantly less effective (almost 10% less F-measure) in the case of rootified data.

2) Strategy II: Since heads are not sufficiently discriminative features, we will now turn to the neighboring nodes in the dependency grammar tree: let us take nodes located at distance 1 from the head. In order to increase pertinence and reduce size of our itemsets, we will restrict ourselves to nouns and proper nouns.

On Fig. 5 the reader can see the results obtained by applying this strategy to rootified (above) and stemmed (below) data.

|          | CAR Recall | CAR Precision | CAR F-measure | SVM Recall | SVM Precision | SVM F-measure |
|----------|------------|---------------|---------------|------------|---------------|---------------|
| MinSupp=0.004, MinConf=55, AvgTransSize=1.63 | 66.39 | 52.99 | 87.81 | 75.26 | 70.53 | 48.49 | 57.35 |
| MinSupp=0.004, MinConf=46.2, AvgTransSize=1.64 | 52.73 | 40.03 | 55.32 | 55.15 | 43.20 | 34.45 | 46.81 |
| MinSupp=0.004, MinConf=47.4, AvgTransSize=2.25 | 73.22 | 64.20 | 88.91 | 77.83 | 74.72 | 58.92 | 62.54 |

3) Strategy III: Our next attempt will be to combine the head node with neighboring noun (and proper noun) nodes. This approach will be incrementally generalized in Strategies III\(_2\) and III\(_3\): in fact we take, in Strategy III\(_1\), all nouns and proper nouns at distance \( \leq n \). On Fig. 6 the reader can see the results obtained by applying strategy III\(_1\) to rootified (above) and stemmed (below) data (\( n = 1 \)).

|          | CAR Recall | CAR Precision | CAR F-measure | SVM Recall | SVM Precision | SVM F-measure |
|----------|------------|---------------|---------------|------------|---------------|---------------|
| MinSupp=0.004, MinConf=55, AvgTransSize=2.25 | 70.54 | 68.67 | 66.29 | 88.75 | 83.91 | 77.55 | 65.10 |

Surprisingly, the results are much better than those of Strategy I: we observe a performance increase of 29.6% for CARs and of 51.3% for SVMs (rootified data). This is due to two factors: the semantic pertinence of nouns dependent on the head, and the increase of the average transaction size from 1 (Strategy I) to 2.25 (Strategy II).

Otherwise we still observe the small difference between rootified and stemmed data as well as the outliers we encountered in Strategy I.

Let us now compare this strategy with tfidf feature selection: in the case of rootified data, we get a comparable average transaction size for \( N = 2 \); in that case F-measure is 62.10, which is almost exactly the result we get in Strategy II. In the case of stemmed data, we get a comparable average transaction size for \( N = 3 \), that corresponds to a F-measure of 55.89, which is, once again, significantly less efficient than Strategy II.

As we can see, the results are slightly better (by about 3%) than Strategy II, while the average transaction size is still low: 2.25. The high recall of International News and the high precision of Sports and Religion have increased as well. The results of the SVM classifier are still around 15% behind those of the CAR classifier. Finally, to compare with tfidf feature selection we need to consider tfidf results based on the same transaction size: for rootified text this is the case for \( N = 2 \), the F-measure of which is 62.10% which is equal to the III\(_2\) result; for stemmed text, \( N \) will rather be 3, the F-measure of which is 55.89%, which is way beneath the corresponding III\(_1\) result.

To conclude: III\(_2\) gives better results than II, without increasing average transaction size beyond measure, CAR classifier is better than SVM, and dependency grammar equal in performance to tfidf for rootified text and better than tfidf for stemmed text.

4) Strategy III\(_3\): In this case we take all nouns and proper nouns at distance 1 or 2 from the head. On Fig. 7 (p. 8) the reader can see the results obtained by applying this strategy to rootified (above) and stemmed (below) data.

We observe a significant improvement compared to Strategy III\(_2\) but also that the average transaction size has almost doubled: it is now around 4.28. Recall of International News and precision of Religion have increased, although not as much as the global F-measure, while precision of Sports has not changed. SVMs still perform less than CARs, but the difference between the two has decreased.
To compare with tfidf feature selection, we need to take \( N = 4 \) for rootified data and \( N = 7 \) for stemmed data. In the first case, the F-measure is 65.20\%, which is a bit less than \( \text{III}_2 \), and in the second case, F-measure is 62.36\% which is significantly worse than \( \text{III}_2 \).

5) Strategy \( \text{III}_3 \): In this case we take all nouns and proper nouns at distance 1, 2 or 3 from the head. On Fig. 8 the reader can see the results obtained by applying this strategy to rootified (above) and stemmed (below) data.

As expected, the results are better than those of \( \text{III}_3 \) but only by 1.87\% for rootified data and by 1.28\% for stemmed data (!). On the other hand, average transaction sizes are more than the double than the corresponding \( \text{III}_3 \): 13.76 for rootified and 13.90 for stemmed text. This shows clearly that it not necessary to go beyond \( \text{III}_3 \), if CARs are to be used. Nevertheless, the spectacular result of this strategy is the performance of SVMs: not only has it increased, but it now exceeds significantly the one of CARs. This shows that SVMs become interesting classifiers when the average transaction size gets high.

| CUL | LOC | INT | SPO | REL | ECO | AVG |
|-----|-----|-----|-----|-----|-----|-----|
| SVM Recall | 72.02 | 86.08 | 94.58 | 88.30 | 89.04 | 63.48 | 70.49 |
| SVM Precision | 87.82 | 65.31 | 81.52 | 94.55 | 89.71 | 83.65 | 71.79 |
| SVM F-measure | 79.14 | 74.27 | 87.57 | 91.32 | 89.38 | 72.14 | 70.54 |
| MinSupp=0.022, MinConf=46.6, AvgTransSize=6.44 |

Fig. 8. Results of Strategy III applied to rootified (above) and stemmed (below) data.

Performance has still increased in quite homogeneous way, by around 3\%. The average transaction size (6.44) is higher than for \( \text{III}_2 \), but remains in reasonable bounds. Recall of International News and precision of Sports seem to have reached their highest possible values which are around 95–96\%.

To compare with tfidf feature selection, we need to take \( N = 7 \) for rootified data and \( N = 13 \) for stemmed data. In the first case, the F-measure is 65.64\%, and in the second case, F-measure is 66.33\%. Both of them are significantly worse than \( \text{III}_3 \).

6) Strategies IV and IV′: Instead of continuing in the same realm and defining \( \text{III}_n \) for \( n \geq 4 \), we will adopt a new strategy and use all nouns of each sentence. The goal is to show that this new strategy (called Strategy IV) has only slightly better performance than \( \text{III}_3 \), while the average transaction size grows significantly.

On Fig. 9 the reader can see the results obtained by applying this strategy to rootified (above) and stemmed (below) data.

As expected, the results are better than those of III but only by 1.87\% for rootified data and by 1.28\% for stemmed data (!). On the other hand, average transaction sizes are more than the double than the corresponding III: 13.76 for rootified and 13.90 for stemmed text. This shows clearly that it not necessary to go beyond III, if CARs are to be used. Nevertheless, the spectacular result of this strategy is the performance of SVMs: not only has it increased, but it now exceeds significantly the one of CARs. This shows that SVMs become interesting classifiers when the average transaction size gets high.

How about going even farther, and taking also all verbs of the sentence? We have tried this strategy (which we call IV′), and the results are disappointing. On Fig. 10 (p. 9) the reader can see the results obtained by applying Strategy IV′ to rootified (above) and stemmed (below) data.

As the reader can see, while average transaction size has increased as expected, performance has decreased, and this goes both for CARs and for SVMs.
In this paper we have studied various text classification techniques applied to Arabic text. We have prepared two versions of the corpus: one with rootified words and one with stemmed words. We have considered sentences as transactions of features which we have selected in two ways: by tf-idf techniques and by dependency grammar techniques. Finally we have applied two kinds of classifiers to our data: class association rules and support vector machines. The results can be synthesized as follows:

1) classification of stemmed text is slightly better than classification of rootified text;
2) dependency grammar feature selection gives better results when compared over transactions of the same average size;
3) class association rules are better than support vector machines in small transaction sizes but get better when transaction size gets high (in which case dependency grammar is not used anymore and the feature selection criteria are purely morphological).

REFERENCES

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In Proceedings of the ACM SIGMOD International Conference on Management of Data 1993, Washington DC, USA, pages 207–216, 1993.

[2] Ahmed H. Al-Harbi, Abdurahman Alnuhureb, Abdulmohsen Al-Thubaity, Mohammad S. Khorsheed, and Abdullah Al-Rajeh. Automatic Arabic text classification. In Proceedings of Journées internationales d’analyse statistique des données textuelles 2008, Lyon, France, pages 77–83, 2008.

[3] Ibrahim A. Al-Kharashi and Martha W. Evens. Comparing words, stems, and roots as index terms in an Arabic information retrieval system. Journal of the American Society for Information Science, 45(8):548–560, 1994.

[4] Qasem Al-Raidaieh, Emad Al-Shawafka, Abdullah S. Ghareb, and Hani Abu-Salem. An approach for Arabic text categorization using association rule mining. International Journal of Computer Processing of Languages, 23(1):81–106, 2011.

[5] Bassam Al-Salemi and Mohammad Juzaddin Ab Aziz. Statistical Bayesian learning for automatic Arabic text categorization. Journal of Computer Science, 7(1):39–45, 2011.

[6] Riyad Al-Shalabi and Rasha Obeidat. Improving kNN Arabic text classification using tfidf techniques and by dependency grammar techniques. In Proceedings of INFOS 2008, Cairo, Egypt, pages NLP–108–112, 2008.

[7] Eiman Tamah Al-Shammari. Improving Arabic document categorization: Introducing local stem. In Proceedings of ISDA 2010, Cairo, Egypt, pages 385–390, 2010.

[8] Abdulmohsen Al-Thubaity, Norah Abanumay, Sara Al-Jerayyed, Aljoharah Alrulkban, and Zarah Manaa. The effect of combining different feature selection methods on Arabic text classification. In Proceedings of ACIS/SPND 2013, Honolulu, Hawaii, USA, pages 211–216, 2013.

[9] Aya Al-Zoghby, Ahmed Sharaf Eldin, Nabil A. Ismail, and Taher Mansour. Categorize Arabic data sets using multi-class classification of CIKM 2002, McLean, VA, USA, pages 340–347, 2002.

[10] Mohammad Aljilaiy and Ophir Frieder. On Arabic search: Improving the retrieval effectiveness via a light stemming approach. In Proceedings of Journées internationales d’analyse statistique des données textuelles 2008, Lyon, France, pages 77–83, 2008.

[11] Jaber Alwedyan, Wa’el Musa Hadi, Ma’an Salam, and Hussein Y. Mansour. Categorize Arabic data sets using multi-class classification based on association rule approach. In Proceedings of ISWA 2011, Amman, Jordan, pages 181–188, 2011.

[12] Mohamed Alzara, Tamer Fatayer, and Aalaa El-Halees. Arabic text classification using Learning Vector Quantization. In Proceedings of INFOS 2012, Cairo, Egypt, pages NLP–39–43, 2012.

7) Synthesis: We are synthesizing the results of this section in Fig. 11, where we represent F-measure as a function of the different strategies we used. One can see clearly that stemmed text gives always slightly better results than rootifted text, and also that SVMs are significantly less powerful in small transaction size strategies, but have a higher slope and hence exceed CAR strategies in Strategy IV.

VI. CONCLUSION

In this paper we have studied various text classification techniques applied to Arabic text. We have prepared two versions of the corpus: one with rootified words and one with stemmed words. We have considered sentences as transactions of features which we have selected in two ways:
Abderrahim Boudlal, Abdelhak Lkahouaja, Azzeddine Mazroui, Abde-louafi Meziane, Mohamed Ould Abdallah Ould Bubah, and M. Shoul. Alkalhi Morpho Sys: A morphosyntactic analysis system for Arabic texts. In Proceedings of ACLIT 2010, Benghazi, Libya, 2010. http://sourceforge.net/projects/alkalhi/.

Noam Chomsky. Syntactic structures. Mouton, 1957.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine Learning, 20(3):273–297, 1995.

Kareem Darwish. Building a shallow Arabic morphological analyzer in one day. In Proceedings of the ACL 2002 Workshop on Computational Approaches to Semitic Languages, Stroudsburg, PA, USA, pages 1–12, 2002.

Ezra Doya, Dan Roth, and Shuly Wintner. Identifying semantic roots: Machine learning with linguistic constraints. Computational Linguistics, 34(3):429–448, 2008.

Joseph Dicky and Ali Farghaly. Roots & patterns vs. stems plus grammar-lexis specifications: On what basis should a multilingual lexical database centred on Arabic be built? In Proceedings of MT Amicitia 2003, New Orleans, USA, 2003.

Rehab Duwairi. Arabic text categorization. International Arab Journal of Information Technology, 4(2):125–131, 2007.

Mahmoud El-Haj and Rim Koulali. Kalimat, a multipurpose Arabic corpus. In Second Workshop on Arabic Corpus Linguistics, Lancaster University, UK, 2013. http://sourceforge.net/projects/kalimat/.

Alaa El-Halees. Arabic text classification using maximum entropy. The Islamic Journal of University, 15(1):157–167, 2007.

Alaa El-Halees. A comparative study on Arabic text classification. Egyptian Computer Science Journal, 30(2), 2008.

Mohamed El Kourdi, Amine Bensaid, and Tajee-eddine Rachidi. Automatic Arabic document categorization based on the naïve Bayes algorithm. In Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages 2004, Geneva, Switzerland, pages 51–58, 2004.

Christiane Fellbaum, Masa Alkhalifa, William J. Black, Sabri Elkabt, Horacio Rodriguez, and Piek Vossen. Introducing the Arabic WordNet project. In Proceedings of the 3rd Global Wordnet Conference, Jeju Island, Korea, 2003.

Abdullah S. Ghareb, Abdul Razak Hamdan, and Azaraliza Abu Bakar. Text associative classification approach for mining Arabic data set. In Proceedings of DMO 2012, Langkawi, Malaysia, pages 791–796, 2012.

Abdullah S. Ghareb, Abdul Razak Hamdan, and Azaraliza Abu Bakar. Integrating noun-based feature ranking and selection methods with Arabic text associative classification approach. Arabian Journal for Science and Engineering, pages 1–16, 2014.

Tarek F. Ghurib, Mena B. Habib, and Zakki T. Fayed. Arabic text classification using Support Vector Machines. I. J. Comput. Appl., 16(4):192–199, 2009.

Spence Green and Christopher D. Manning. Better Arabic parsing: Baselines, evaluations, and analysis. In Proceedings of COLING 2010, Bejing, China, pages 394–402, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Meryem Hadni, Abdelmonaime Lachkar, and Said alaaou Ouatik. A new and efficient stemming technique for Arabic text categorization. In Proceedings of ICMC 2012, Tangier, Morocco, pages 791–796, 2012.

Yannis Haralambous and Philippe Lena. Text classification using association rules, dependency pruning and hyperonymization. In Proceedings of DMNL 2014, Nancy, France, volume 1202 of CEUR Workshop Proceedings, pages 65–80, 2014.

Majed Ismail Hussein, Fekry Olayah, Minwer AI-Dwan, and Ahlami Shamsan. Arabic text classification using SMO, Naïve Bayesian, I48 algorithms. IJRRAS, 9(2):306–316, 2011.

Thorens Joachims. Text categorization with Support Vector Machines: Learning with many relevant features. In Proceedings of ECML 1998, Chemnit, Germany, pages 137–142, Berlin, 1998. Springer.

Mahmood H. Kadhim and Nazlia Omar. Bayesian learning for automatic Arabic text categorization. Journal of Next Generation Information Technology, 4(3):1–8, 2013.

Ghassan Kanaan, Riyadh Al-Shalabi, and Abdallah Al-Akhras. kNN Arabic text categorization using IG feature selection. In Proceedings of CSIT 2006, Amman, Jordan, 2006.

Abdi Karima, Eiberehrr Zakaria, and Titi Guissa Yamina. Arabic text categorization: A comparative study of different representation modes. Journal of Theoretical and Applied Information Technology, 38(1):1–5, 2012.

Shereen Khoja, Roger Garside, and Gerry Knowles. An arabic tagset for the morphosyntactic tagging of arabic. In Proceedings of Corpus Linguistics 2001, Lancaster, UK, 2001.

Laila Khiarais. Arabic text classification using n-gram frequency statistics. A comparative study. In Proceedings of Data Mining 2006, Las Vegas, USA, pages 78–82, 2006.

Laila Khiarais. A machine learning approach for Arabic text classification using n-gram frequency statistics. J. Informetrics, 3:72–77, 2009.

Leah S. Larkey, Lisa Ballesteros, and Margaret E. Connell. Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis. In Proceedings of ACM SIGIR 2002, Tampere, Finland, pages 275–282, 2002.

Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In Proc. of the Int. Conf. on Knowledge Discovery and Data Mining (New York), pages 80–86, 1998.

Igor A. Mel’tu. Dependency syntax: theory and practice. Albany: State University Press of New York, 1987.

Abdelwadood Mesleh. Chi square feature extraction based SVMs Arabic language text categorization system. Journal of Computer Science, 3(6):430–435, 2007.

Abdelwadood Moh’d Mesleh. Feature sub-set selection metrics for Arabic text classification. Pattern Recognition Letters, 32(14):1922–1929, 2011.

Will Monroe, Spence Green, and D. Christopher Manning. Word segmentation of informal Arabic with domain adaptation. In Proceedings ACL 2014, Baltimore, Maryland, pages 206–211, 2014.

Atiaa Nehar, Djelloul Ziadi, Hadda Cherroun, and Younes Guellouma. An efficient stemming for Arabic text classification. In Proceedings of IIT 2012, Abu Dhabi, UAE, pages 328–332, 2012.

Hatem M. Noaman, Sameh El-Nassef, Ahmed Ghoneim, and Taher Hamza. Naïve Bayes classifier based Arabic document categorization. In Proceedings of INFOS 2010, Cairo, Egypt, pages 1–5, 2010.

Marwan Ali H. Omer and Ma Shi Long. Stemming algorithm to classify Arabic documents. In Proceedings of the Symposium on Progress in Information & Communication Technology 2009, Kuala Lumpur, Malaysia, pages 111–115, 2009.

Saeed Raheel, Joseph Dicky, and Mohamed Hassoun. The automatic categorization of Arabic documents by boosting decision trees. In Proceedings of SITIS 2009, Marrakesh, Morocco, pages 294–301, 2009.

Dina A. Said, Nayer M. Wanas, Nevin M. Darwish, and Nadia H. Hegazy. A study of text preprocessing tools for Arabic text categorization. In Proceedings of 2nd International Conference on Arabic Language Resources & Tools, 2009, Cairo, Egypt, pages 230–236, 2009.

Hassan Sawaf, Jörn Zaplot, and Hermann Ney. Statistical classification methods for Arabic news articles. In Proceedings of Arabic Natural Language Processing in ACL 2001, Toulouse, France, 2001.

Mostafa M. Syiam, Zakki T. Fayed, and Mena B. Habib. An intelligent system for Arabic text categorization. LUCIS, 6(1):1–19, 2006.

Lucien Tesnière. Éléments de syntaxe structurale. Klinckseck, 1959.

Fadi Thabthah, Omar Gharbath, and Hussein Abdellajber. Comparison of rule based classification techniques for the Arabic textual data. In Proceedings of ISICT 2011, Amman, Jordan, pages 105–111, 2011.

Fadi Thabthah, Wa’el Masa Hadi, and Gaith Al-Shammare. VSMs with k-nearest neighbour to categorise Arabic text data. In Proceedings of WCECS 2008, San Francisco, USA, 2008.

Abdullah Wahbeh, Mohammed Al-Kabi, Qasem Al-Radaideh, Emad Al-Shawkawa, and Izzat Alsmad. The effect of stemming on Arabic text classification: An empirical study. International Journal of Information Retrieval Research, 1(3):54–70, 2011.

Mounir Zrigui, Rami Ayadi, Murad Mars, and Mohsen Maraoui. Arabic text classification framework based on Latent Dirichlet Allocation. Journal of Computing and Information Technology, 20(2):125–140, 2012.