Supervised classification of languages used by Moroccans in social networks

Moussaoui Otman[1], El Younoussi Yacine[1], Azroumahli Chaimae[1]

[1] Information System and Software Engineering, National School of Applied Sciences, Abdelmalek Essaadi University, Tetouan, Morocco

E-mail: otman.moussaoui@etu.uae.ac.ma, yacine.elyounoussi@uae.ac.ma, c.azroumahli@uae.ac.ma

Article history

Received July 25, 2022
Revised Oct 10, 2022
Accepted Nov 05, 2022
Published Dec 31, 2022

ABSTRACT

On social networks, such as Facebook, users's comments cover several languages, thus, knowing the language of a comment could be very valuable for any further processing. Across this paper, we compare the performance of some typical classification approaches applied on our manually annotated dataset. This dataset is composed of Facebook comments of Moroccan users. The classification approaches we have considered in this work are Naive Bayes, Support Vector Machines, K-Nearest Neighbors, Logistic Regression, Gradient Boosting, Random Forest, Decision Trees as well as Multi-layer Perceptron. The results obtained show that the Multi-layer Perceptron algorithm scored the highest success rate (86.79%), followed by the logistic regression (86.71%) and the Naive Bayes (85.64%).

Keywords: ***NLP, Morocco Dialect, automatic classification, social media, Comment***

I. Introduction

The Web is an ever-growing source of textual data, especially data generated in social networks. In fact, many people use these media on a daily basis, not only as a means of communication, but also for exchanging opinions, comments and experiences. Hence, social networks, such as Facebook, has become essential resources for collecting textual data for research in Natural Language Processing (NLP). In order to be able to use this data for other NLP tasks, we need to know in which languages it was written. Automatic language identification is the first step for most NLP applications, such as social media analysis and machine translation [1]. The process of hiring human participants to identify languages and dialects is very expensive and time-consuming. Therefore, machine automation has proven useful in reducing the time needed for information extraction and data analysis provided that we can create an efficient mix of new methods with the appropriate dataset [2]. For the Moroccan dialect, this identification is made a difficult task due to several considerations, including the use of Latin letters and numbers to write the Arabic language, as well as the nature of the Moroccan dialect, which contains several words from Arabic standard. The use of informal language on social media platforms has also become common when users post their comments or share their opinions with other users, most of whom use common abbreviations. The informal nature of social media languages has become an additional challenge to typical NLP issues [3]. Moroccan people may express their social networks comments using several languages such as modern standard Arabic (MSA), dialectal Arabic, French, Spanish or even English. Therefore, language classification of user comments becomes fundamental to build a social media analytics system.

This work focuses on the classification of language used in Moroccan social media users' comments as it is carried out on the basis of a dataset from Facebook comments[4]. The dataset was manually annotated using seven different language labels, namely, Dialect in Latin Alphabet (DAL), Dialect in Arabic Alphabet (DAA), Classical Arabic (ARC), French (FRN), English (ANG), Spanish (ESP) and Others (AUT). To achieve our goal, we applied and compared several classification approaches on this dataset. We used multi-class classifiers, to label a comment under an appropriate class among seven classes. We used a supervised approach for comment classification. We implemented the Term Frequency-Inverse Document Frequency (TF/IDF) method for word representation. Naive Bayes, Support Vector Machines, K-Nearest Neighbors, Logistic Regression, Gradient Boosting, Random Forest, Decision Trees as well as Multi-layer Perceptron classifiers are used in the classification step. The importance of this work lies in the nature of the classification problem which is the identification of the language used by Moroccans in social media comments, the use of hyperparameter tuning algorithms and the number of machine learning models. The contributions of this work are:

• Classification of the language used by Moroccans to write comments on Facebook using a manually annotated dataset of seven different language classes.

• Multiple machine learning classification models were built to find out the most accurate one for our dataset.

• Hyperparameter tuning is to get the best classification accuracy of the eight machine learning algorithms used.

The document is structured as follows: the literature review is presented in section 2. Section 3 presents the classification approaches. The results and discussion are presented in section 4. Finally, we conclude with general findings and future directions in section five.

II. Literature review

Recently, many works have focused on dialect identification for Arabic. However, to our knowledge, there has not been any work implemented for Moroccan dialectal variations classification.

Authors of [5] presented an approach for Arabic dialect identification at the sentence level based on supervised learning. They distinguished between Modern Standard Arabic and Egyptian Dialectal Arabic using the Naive Bayes classifier on an annotated dataset.

In [6], authors used Naïve Bayes classifiers and the n-gram model to distinguish 18 different Arabic dialects in social media datasets. The results showed that the Naïve Bayes classifier performs better if it's based on the bigram character and with great accuracy.

In [7], the authors detected Arabizi words in a text blended with English using Conditional Random Fields (CRF) to identify Arabizi words. They built a set of tweets containing Arabizi, English or a mixture of Arabizi and English, the latter being used to categorize the language at the word level.

Authors of [1] used automatic classifiers and n-gram models to distinguish between four Arabic varieties in an annotated dataset. The system has achieved near-human classification accuracy.

Authors of [8] used a linear Support Vector Machine to perform Arabic dialects identification using a Multidialectal Parallel Corpus covering six classes: MSA, Egyptian (EG), Tunisian(TN), Syrian (SY), Jordanian (JO) and Palestinian (PA).

In [9], authors classified Arabic dialects by building a system that uses the Sequential Minimal Optimization (SMO) algorithm and WEKA data analytic tool. They achieved an average accuracy score of 42.85% on the test data.

In [2], the authors used Naïve Bayes, Support Vector Machine, k–Nearest Neighbor and Decision Trees to automatically detect dialects in a dataset comprising Standard Arabic, Egyptian dialect, Gulf dialect, Levant dialect and North African dialect by applying the method Subtractive Bivalency Profiling (SBP).

Authors of [10] used several classifiers with different features for Arabic dialects identification. They showed that at low resource condition a traditional machine learning classifier tends to perform better when compared to neural network models and that the features help improve the accuracy of dialect classification.

Authors of [11] are combined the classifiers Decision Tree, Naïve Bayes and Logistic Regression using voting for Arabic dialects identification applied on shared dataset of very close 21 classes.

In [12], authors proposed five models: Complement Naïve Bayes, Support Vector Machine, Logistic Regression, Random Forest and Decision Trees to perform dialect identification for Arabic tweets. The Term Frequency Inverse Document Frequency (TF/IDF) technique is implemented for feature extraction. The development results show that the Complement Naïve Bayes classifier outperforms all other classifiers.

Most of the references mentioned above concern Arabic dialects identification whereas the main goal of our work is to recognize the Moroccan dialect varieties used by Moroccans in Facebook. Furthermore, the models we created attempt also to distinguish between Moroccan dialects and other languages used usually by Moroccans such as French, English or even Spanish.

III. Approaches

A. Classification approaches

To be able to choose the most suitable classification approach to solve our problem, we have built and tested several algorithms. Afterwards, based on the test results, we ranked all the algorithms and then we chose the one with the highest score.

We describe below the eight classification methods as well as their hyperparameters that must be tuned to optimally solve our problem.

1) Naïve Bayes (NB)

Naïve Bayes is a model inspired by the effective Bayes theorem in machine learning based on probabilistic classification [13]. Bayes' theorem formula calculates the probability of an event A conditional on another event B that has already occurred.

Hyperparameter tuning for NB [14]

- alpha: It accepts the floating value representing the parameter of additive smoothing
- var_smoothing: Specifies the part of the greatest variance of all the characteristics to be added to the variances for the stability of the calculation.

2) Support Vector Machine (SVM)

The purpose of SVMs [15] is to use a simple boundary to separate data into categories, where the distance between this boundary and different datasets is at the max. SVMs are often based on the employ of kernels. The latter are mathematical functions that allow data to be separated by projecting it into a higher-dimensional vector space.

Hyperparameter tuning for SVM [14]

- kernel: Specifies the type of kernel to use in the algorithm. The latter whose main function is to convert the input data from the data set to the desired form. Among the types of functions, we find the Radial Basis Function (RBF) function, polynomial and linear.
- C: Is the penalty parameter used to indicate to SVM the degree of bearable error or the degree of bearable misclassification.
- gamma: Coefficient of the kernel 'poly', 'sigmoid' and 'rbf'. If we want to count only the nearby points in the calculation of the separation line, we only take lower values of gamma, while we take the upper value of gamma if we want to count all the data points in the calculation of the separation line.

3) K-nearest neighbors (kNN)

The K-Nearest Neighbor algorithm [16] is a nonparametric method in which the model memorizes the observations of the training dataset for the classification of the test dataset. To predict the category of the new data point, it will look for its K nearest neighbors and will choose the class of the majority neighbors.

Hyperparameter tuning for kNN [14]

- n_neighbors: Represents the number of neighbors that will enter the voting process.
- weights: Weight parameters used for prediction
- algorithm: Used to choose the most appropriate algorithm for calculating nearest neighbors.
- p: The hyperparameters used in the Minkowski scale.
- metric: The distance matrix used for the tree.

4) Logistic regression (LR)

Logistic regression [17] is one of the classification algorithms used to classify data into separate classes such as 0/1. This algorithm converts the output using the sigmoid function into probability values which will be categorized into one of the categories within this algorithm.

Hyperparameter tuning for LR [14]

- penalty: Determine the type of penalty settlement.
- C: The C parameter controls the strength modification of regularization; must be a positive float.
- solver: Algorithm used to solve optimization problems.
- max_iter: Determines the maximum number of iterations that can be performed until the solvers converge.
- l1_ratio: The Elastic-Net mixing parameter, with 0<= l1_ratio <=1.

5) Decision Tree (DT)

Decision tree algorithms produce a set of decision rules through which the class is found. On the principle of partitioning into nodes from top to bottom, the dataset is divided into smaller and smaller subsets until reaching the target class and then divides according to the column of the class label that depends on it in classification [18]

Hyperparameter tuning for DT [14]

- criterion: Segmentation quality scale used for decision tree training.
- splitter: Used to identify split features.
- max_depth: Specifies the threshold on the depth maximum to build the tree.
- min_samples_split: The minimum number of data points in demand to split an inner node.
- min_samples_leaf: The minimum number of data points that must be by a sheet node.

6) Random Forest (RF)

This method is based on the notion of multiplying the trees. In order to slightly modify the data, the bootstrap method is used. This is a method that allows, from a sample, to build new samples by random drawing with replacement. Once the samples have been generated, the trees can be constructed. [19]

Hyperparameter tuning for RF [14]

- n_estimators: Determining the number of estimators in trees.
- criterion: Segmentation quality scale used for random forest training.
- max_depth: The maximum depth of the tree. The deeper the tree, the more divisions it has and it captures more information about the data.
- max_features: The size of the random subsets of features that are considered when splitting a node. When constructing each decision tree, only the number of "max_features" features are considered for node splitting.

7) Gradient Boosting (GB)

Gradient boosting is a machine learning algorithm used in classification and regression tasks. Boosting is a method based on the principle of the whole sequentially. It brings together a group of weak learners and provides improved prediction accuracy so that each model learns from errors before it and so on with each iteration.[20]

Hyperparameter tuning for GB [14]

- learning_rate: We can reduce the present each tree. To enable the model to generalize well, it is preferable to use lower values and thus make the model robust to the specific properties of the tree.
- n_estimators: Determining the number of estimators in the boosting stages.
- Subsample: The ratio of the training instance set that can be used in any boosting round. Where doing sampling with replacement at every step of the boosting process.

- max_depth: Specifies the threshold on the depth maximum to build the tree. It depends on the interaction of the input variables to determine the best value.
- max_features: The size of the random subsets of features that are considered when splitting a node.

8) Multi-layer Perceptron (MLP)

MLP is a class of artificial neural network consisting of a number of layers of nodes through which information flows from an input data set to a desired output recursively. Each layer consists of a number of neurons that change from one layer to another, and the output of the system is formed in the neurons of the last layer. [21]

Hyperparameter tuning for MLP [14]

- hidden_layer_sizes: With this parameter we can specify how many layers and how many nodes we want to have in the Neural Network Classifier.
- activation: Function used for the hidden layer neurons.
- solver: Specifies the algorithm used to solve optimization problems.
- alpha: Strength of the L2 regularization term.
- learning_rate_init: The initial learning rate that controls the weight update rate.
- max_iter: Specifies the maximum number of iterations. The training is repeated until the completion of convergence or the use of early stop according to the max_iter.

B. Methodology of classification

1) Classification process

The proposed methodology for classifying comments is based on the scikit-learn library packages. The classification process is illustrated in Figure 1.

This process takes place in three main steps. The first one, is about vectorizing our dataset using the TF-IDF technique. The objective of this step is to calculate the TF-IDF score for each word in our dataset. During the second step, we build our models based on the training dataset (78% of the initial dataset). Indeed, we used eight different classification approaches and we applied hyperparameters tuning. The last step is dedicated to the evaluation of the created models using different performance measures. The evaluation is applied on the test dataset (22% of the initial dataset).

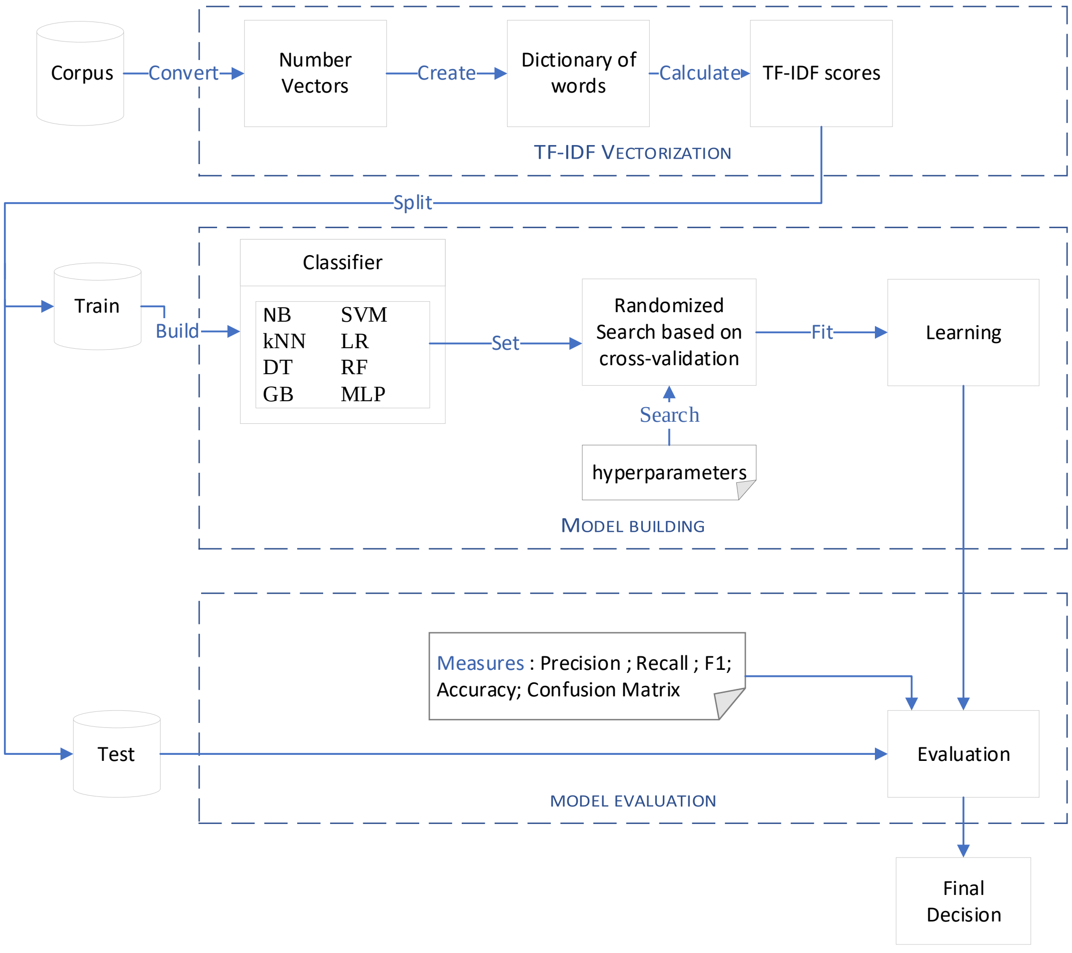


Figure 1. The process of the methodology followed in the classification of comments

2) Performance evaluation of classifiers

In order to well evaluate our classifiers, and because of imbalance of our dataset, we had to use multiple metrics to assess the performances of the different models that we created. The evaluation metrics used are the following: precision(P), accuracy(A), F-Measure(F1), recall(R), confusion matrix as well as macro-average measures.

Accuracy: Accuracy is the proportion between the correct classification and the total number of predictions, expressed as a decimal number between 0 and 1.

$$\text{Accuracy}=\frac{\text{number of correct predictions}}{\text{total number of predictions}} \quad (2)$$

$$A=\frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

***Recall**:* Recall being the proportion of comments correctly classified by the classifier compared to all comments in the class Ci.

$$\text{Recall }(C_i)=\frac{\text{number of comments correctly classified in } C_i}{\text{number of class comments } C_i} \quad (4)$$

$$R_i=\frac{TP_i}{TP_i+FN_i} \quad (5)$$

Precision: Precision is the proportion of correctly classified comments among those classified by the classifier in Ci.

$$\text{Precision }(C_i)=\frac{\text{number of comments correctly classified in } C_i}{\text{number of comments classified in } C_i} \quad (6)$$

$$P_i=\frac{TP_i}{TP_i+FP_i} \quad (7)$$

***F-measure**:* The F-measure is the harmonic mean between recall and precision. We generally choose to give the same importance to the two criteria, so the measure is thus noted F1 which is written:

$$F_1=\frac{2*P*R}{P+R} \quad (8)$$

***Confusion matrix*:** The confusion matrix is a matrix containing counts of a classifier's correct and incorrect comments. The values on the diagonal of this matrix correspond to the data having been correctly classified while the off-diagonal elements correspond to the data incorrectly classified by the classifier. The confusion matrix provides a detailed view of the errors made by our classifier as well as the types of errors committed, allowing us to easily identify the extent of confusion between the classes.[13]

***Macro-average*:** Measures of the macro average type correspond to an average that does not take into account the size of the classes. Macro-averaging first evaluates each class independently. After that, the average of the individual measurements is calculated, by which the overall performance of the workbook is calculated. The different classes then have the same importance. Macro-average precision and recall are calculated as follows:

$$P=\frac{\sum_{i=1}^{|C|} P_i}{|C|} \quad (9)$$

$$R=\frac{\sum_{i=1}^{|C|} R_i}{|C|} \quad (10)$$

Where C = {DAL, DAA, ARC, FRN, ANG, ESP, AUT} is the set of classes defined in our classification task, and TP, FP, TN and FN refer respectively to the number of true positive, false positive, true negative, and false negative responses.

IV. Results and Discussion

A. Runtime environment

Regarding the runtime environment, we used MARWAN-HPC [22] which is a High-Performance Computing system remotely accessible and made up of 38 nodes interconnected by a very low latency network at 100 Gbps. The MARWAN-HPC services are provided to Moroccan researchers by the National Center for Scientific and Technical Research (CNRST). This infrastructure offers the following capacity:

- 1672 CPU Cores (165 TFlops)
- 396 TB Storage
- 10.4 TB RAM
- 4 GPUs

Our experiments are carried out in a Python environment. There are many reasons for this choice. The language offers great flexibility, an important source of documentation and a large library of research in the NLP field.

B. Datasets

This work is carried out on the basis of a dataset of Facebook comments [4]. The dataset was manually annotated by using seven different language labels (DAL: Dialect in Latin Alphabet, DAA: Dialect in Arabic Alphabet, ARC: Classical Arabic, FRN: French, ANG: English, ESP: Spanish and AUT: Others).

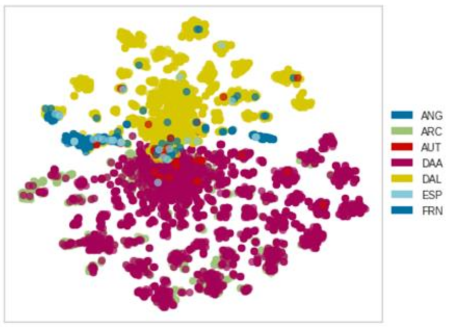


Figure 2. Visualization of the dataset by t-SNE

TABLE 1 DISTRIBUTION OF OUR DATASET COMMENTS

Classes	**The dataset contains a total of 5917 comments**
ANG	62
ARC	1144
AUT	34
DAA	2512
DAL	1944
ESP	28
FRN	193

The dataset was divided into two parts: a training dataset composed of 78% of the initial dataset comments, and a testing dataset that contains the remaining comments (22%).

1) Data collection

To build our corpus, we have chosen Facebook since it is one of the most active social media platforms in Morocco. We started by selecting pages with a large number of members and posts related to Moroccan contexts and users. When creating a page on Facebook, one has to define the category of his page. This categorization helps us to select six different types of pages to build our dataset which are Celebrities, Community, Entertainment, Media, Place and Sport. For each category, we have isolated ten collective pages according to the number of Moroccan fans. We have collected a total of 4,822,723 comments between 11 September and 18 November 2018. The total number of comments collected is equal to 157.7M. Then, we cleaned up the comments by deleting the duplicate and empty comments. After that, we have selected a set of 5917 comments for the manual annotation.

2) Data Annotation

Each comment in our selection is annotated using one among seven labels.

TABLE 2 LIST OF LANGUAGE LABELS

Class	**Description**
DAL	Comment in Dialect Latin Alphabet
DAA	Comment in Dialect Arabic Alphabet
ARC	Comment in Classic Arabic
FRN	Comment in French
ANG	Comment in English
ESP	Comment in Spanish
AUT	If the comment: •Written by other language (example: یہی کام باقی ہے کیا..اتنے) •Contains only Facebook Username (example: Ghita Bennani) •Contains both Arabic and Latin characters (example: hak next gen قد بلغ السيل ما بلغ) •Contains only named entities (example: omar, canada, احمد, طنجة) •Others: punctuation, numbers, emoticons, URL, hashtag, etc. •Contains only other strings : (example : hhhhhhhh, hahhahaaah, هههههه, up, اب) •Is ambiguous: where the type of comment cannot be determined.

C. *Detailed results*

TABLE 1 RESULTS OF NAÏVE BAYES CLASSIFIER

Types	• ***var_smoothing*** ϵ ***[0 .. -5]***	• ***alpha*** ϵ ***[1.0e-100 .. 1e-1]***	**accuracy**	**Weighted avg**			**Running time**
				precision	***recall***	***f1-score***	
Gaussian	0.00016	-----	0.8280	0.85	0.83	0.84	1min30s
Multinomial	------	0.1	0.8564	0.85	0.86	0.85	43s
Complement	------	0.0276	0.8379	0.88	0.84	0.85	44s
Bernoulli	------	1e-30	0.8356	0.86	0.84	0.85	1min10s

TABLE 2 RESULTS OF SVM CLASSIFIER

kernel	• ***gamma*** ϵ ***[0.01,0.1, 1, 10, 100]*** • ***C*** ϵ ***[0.01,0.1, 1, 10, 100]***	**accuracy**	**Weighted avg**			**Running time**
			precision	***recall***	***f1-score***	
Linear	gamma: 1 ; C: 10	0.8502	0.85	0.85	0.84	18h2min13s
rbf	gamma: 0.01 ; C: 100	0.8548	0.86	0.85	0.85	19h12min1s
poly	gamma: 'auto' ; 'C: 0.01	0.4247	0.18	0.42	0.25	14h11min21s
sigmoid	gamma: 0.01 ; C: 100	0.8510	0.86	0.85	0.84	13h31min39s

TABLE 3 RESULTS OF KNN CLASSIFIER

algorithm	• *metric ϵ [minkowski, euclidean, Manhattan, chebyshev]* • *Weights ϵ [uniform,distance]* • *p ϵ [1,2]* • *n_neighbors ϵ [1..30]*				**accuracy**	**Weighted avg**			**Running time**
						precision	***recall***	***f1-score***	
auto	euclidean	uniform	2	5	0.8349	0.83	0.83	0.83	2h36min 35s
ball_tree	euclidean	distance	2	8	0.8364	0.83	0.84	0.83	6h14min 48s
brute	minkowski	distance	2	9	0.8395	0.83	0.84	0.83	2h41min 59s
kd_tree	euclidean	distance	1	9	0.8403	0.84	0.84	0.83	6h58min 35s

TABLE 4 RESULTS OF LOGISTIC REGRESSION CLASSIFIER

solver	• *Penalty ϵ [l1, l2, elasticnet]* • *C ϵ [0.01 .. 1000]*		**accuracy**	**Weighted avg**			**Running time**
				precision	***recall***	***f1-score***	
Lbfgs	'l2'	100	0.8671	0.87	0.87	0.86	4min20s
liblinear	'l2'	1000	0.8664	0.86	0.87	0.86	4h9min24s
newton-cg	'l2'	100	0.8671	0.87	0.87	0.86	2min44s
sag	'l2'	100	0.8671	0.87	0.87	0.86	6h25min13s
saga	'l2'	1000	0.8664	0.87	0.87	0.86	158h39min20s

TABLE 5 RESULTS OF DECISION TREE CLASSIFIER

• *Splitter ϵ [best , random]* • *Criterion ϵ [gini, entropy]* • *max_depth ϵ [1-3000]* • *min_samples_split ϵ [1-10]* • *min_samples_leaf ϵ [1-10]*					**accuracy**	**Weighted avg**			**Running time**
						precision	***recall***	***f1-score***	
best	gini	default	5	3	0.7174	0.73	0.72	0.71	12min54s
best	entropy	[1-500] → 372	2	2	0.7189	0.72	0.72	0.71	12min0s
random	entropy	[501-1000] →795	7	1	0.7481	0.74	0.75	0.73	11min31s
random	gini	[1001-1500] →1251	5	2	0.7335	0.74	0.73	0.72	11min59s
best	gini	[1501-2000] →1926	4	1	0.7565	0.75	0.76	0.74	13min14s
best	entropy	[2001-2500] →2128	5	1	0.7358	0.73	0.74	0.72	12min12s
best	gini	[2501-3000] →2903	7	1	0.7527	0.75	0.75	0.74	12min52s

TABLE 6 RESULTS OF RANDOM FOREST CLASSIFIER

• *n_estimators ϵ [10-3000]* • *Criterion ϵ [gini , entropy]* • *max_depth ϵ [10-2000]* • *max_features ϵ [auto , sqrt]*				**accuracy**	**Weighted avg**			**Running time**
					precision	***recall***	***f1-score***	
687	entropy	474	auto	0.8195	0.82	0.82	0.80	1h18min30s
1531	entropy	500	sqrt	0.8164	0.81	0.82	0.80	5h22min3s
3000	entropy	371	sqrt	0.8034	0.82	0.80	0.79	8h29min58s
531	entropy	1537	auto	0.8103	0.81	0.81	0.79	2h8min8s
1947	entropy	1417	sqrt	0.8065	0.80	0.81	0.79	7h0min12s
2322	entropy	914	sqrt	0.8065	0.81	0.81	0.79	10h18min3s

TABLE 7 RESULTS OF GRADIENT BOOSTING CLASSIFIER

learning_rate	• *n_estimators ϵ [10 - 1000]* • *subsample ϵ [0.2 - 0.9]* • *max_depth ϵ [1 - 500]* • *max_features ϵ ['auto', 'sqrt','log2']*				**accuracy**	**Weighted avg**			**Running time**
						precision	***recall***	***f1-score***	
0.0001	890	0.5	76	sqrt	0.4616	0.52	0.46	0.33	7h30min1s
0.005	880	0.7	338	log2	0.8349	0.84	0.83	0.82	5h21min17s
0.001	920	0.7	368	sqrt	0.8088	0.83	0.81	0.80	14h57min2s
0.01	360	0.5	459	sqrt	0.8318	0.84	0.83	0.82	10h30min48s
0.1	530	0.7	142	log2	0.8464	0.84	0.85	0.84	6h53min48s
1	70	0.6	454	sqrt	0.8234	0.82	0.82	0.82	1h56min54s

TABLE 8 RESULTS OF MULTILAYER PERCEPTRON CLASSIFIER

activation	• *Solver ϵ ['lbfgs', 'sgd', 'adam']* • *random_state ϵ [0,1,2,3,4,5,6,7,8,9]* • *max_iter ϵ [50 – 5000]* • *hidden_layer_sizes ϵ np.arange(10, 15)* • *alpha ϵ [0.00005 - 0.01]*					**accuracy**	**Weighted avg**			**Running time**
							precision	***recall***	***f1-score***	
relu	adam	3	4500	25	5e-05	0.8648	0.86	0.86	0.86	6h14m27s
identity	lbfgs	6	1500	11	0.5	0.8656	0.86	0.87	0.86	4h47m53s
logistic	lbfgs	5	1600	27	0.5	0.8648	0.86	0.86	0.86	1h0m30s
tanh	adam	1	1400	13	5e-05	0.8679	0.86	0.87	0.86	5h4m33s

D. Discussion

TABLE 9 EVALUATION METRICS OF ALL CLASSIFIERS SORTED BY ACCURACY

Classifier	**Accuracy**	**Precision**	**Recall**	**F1**
MLP	86,79%	0.86	0.87	0.86
LR	86,71%	0.87	0.87	0.86
NB	85,64%	0.85	0.86	0.85
SVM	85,48%	0.86	0.85	0.85
GB	84,64%	0.84	0.85	0.84
kNN	84,02%	0.84	0.84	0.83
RF	81,95%	0.82	0.82	0.80
DT	75,65%	0.75	0.76	0.74

Table 11 shows the most efficient version (which show the best performance results) of each model. These results were obtained by using the Randomized Search approach based on cross-validation to find the optimal hyperparameters.

The classification results led to the following findings:

- The accuracy is greater than 75% for all the classifiers.
- Six classifiers show an accuracy score over than 84%.
- We notice that the MLP and logistic regression achieved respectively a success rate of 86.79% and 86.71%. In addition, they obtained the best F-measure (86%).
- The MLP classifier records the best percentage (86.79%) of correctly classified comments.
- The Decision Tree classifier records the lowest rate of correctly classified comments with 75.65%.

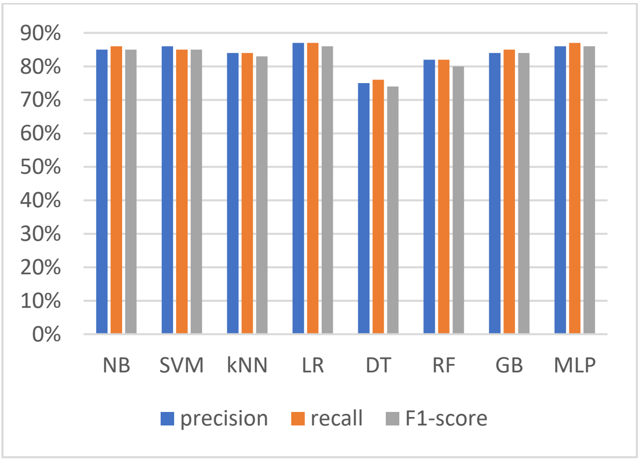


Figure 3. Recall, precision and F-Measure of the different classifiers.

Figure 3 is a histogram comparing the different values of precision, recall and F-measure of table 10. Note that the values of the three measurements are almost similar for all the algorithms.

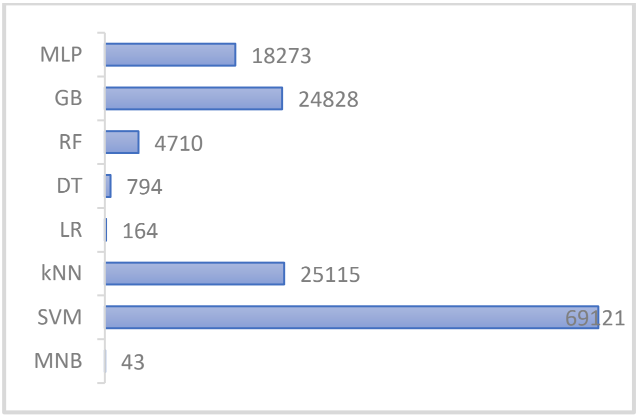


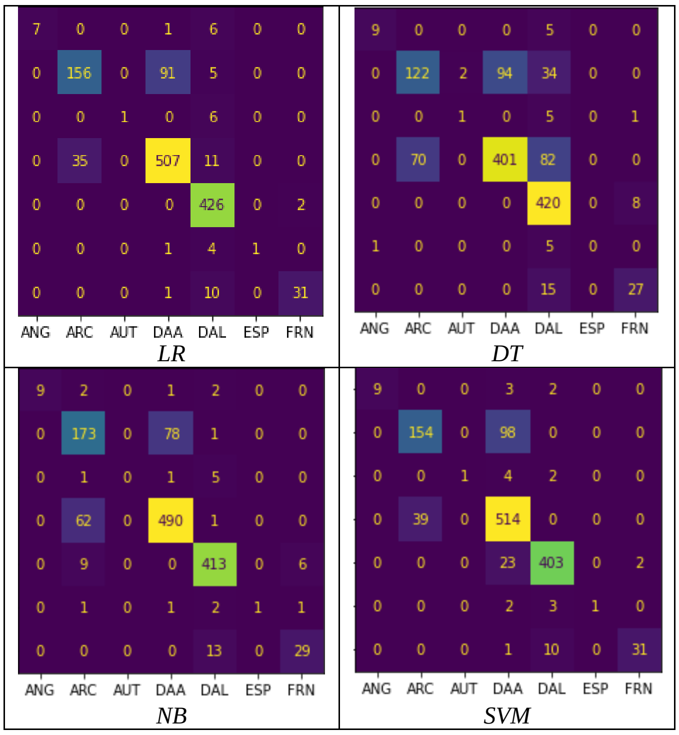
Figure 4. Training time (in second) of each classifier

Figure 4 shows a histogram of the average training time of the eight classifiers.

The results showed that the classifiers that performed the hyperparameters tuning in less time are MNB and LR.

The main weakness of the SVM classifier consist of the initial data sorting which requires a very long calculation time.

While the kNN classifier gives good results, it relatively takes a long time during the hyperparameters finetuning step that can be reduced by using the subsampling of the dataset (a smaller dataset).



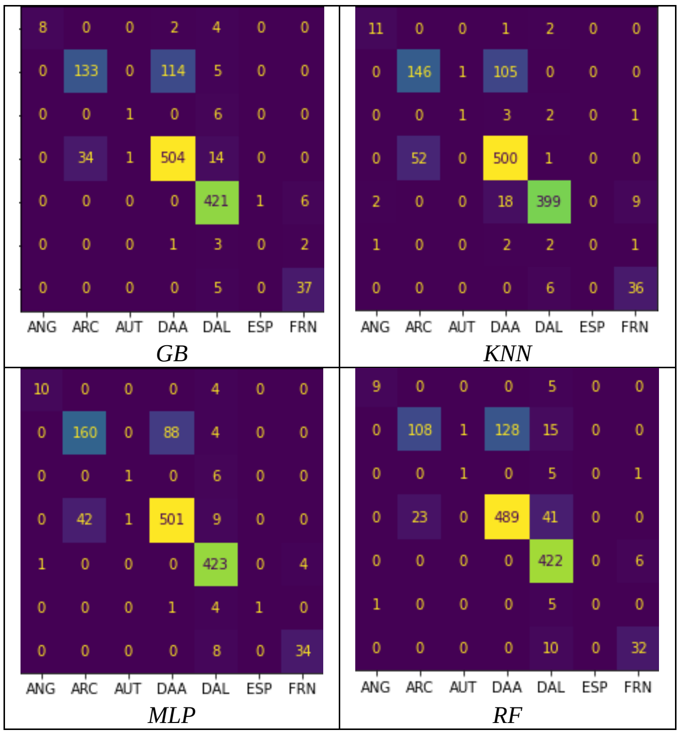


Figure 5. Confusion matrix of the different classifiers

In our case, the overall performance depends widely on the results related to the most representative classes, i.e, DAA, DAL and ARC. Therefore, the discussion of the results will focuses mainly on these three classes.

The confusion matrices indicate that there are errors for the three classes, more particularly the ARC class which recorded an error rate between 31% and 52% (incorrectly classified as DAA). This can be explained by the nature of Moroccan dialect which contain several words from Standard Arabic (ARC).

For example, in the matrix of the GB classifier, in the second row, 252 comments were recorded, 133 of them were correctly classified, 114 being assigned to the DAA class and 5 to the DAL class. Similarly, in the second line of the matrix that represents the NB classifier, among 252 comments, 173 were correctly classified, and 79 were not.

The errors that have been recorded for the DAL class do not exceed 7% for all classifiers.

In the matrix that represents the examples predicted by the LR classifier, at the fifth line there were 428 comments of DAL class, only 2 of them were misclassified.

For the DAA class, except the DT algorithm which recorded a success rate of 73%, all the other classifiers exceeded the value of 88%.

For the ARC class, the best success rate was scored by the NB classifier, about 69%. Concerning the DAA class, the best result was obtained by the SVM classifier with 93% of well-ranked comments. The highest score was obtained by the LR classifier for the DAL class which recorded a success rate of almost 100%.

Finally, we can notice that the models predict less accurately the ARC class, otherwise they predict more precisely the DAL and DAA classes. The results also indicate that the main cause of errors is due to the confusion that exists between the classification of ARC and DAA classes.

V. Conclusions

In this paper, we presented the methodology we followed to classify the language used by Moroccans to write comments in social networks (Facebook). Our dataset was manually annotated with seven different language classes. In order to find out the most accurate algorithm to perform this kind of classification, we compared eight different classifiers. To achieve our goal, we used the TF-IDF vectorization technique. Then, we applied eight classifiers during the training phase (NB, SVM, kNN, LR, DT, RF, GB and MLP). Finally, we compared the results obtained using Accuracy, Recall, Precision, F-measure and confusion matrix measures. The results indicate that the NB, the MLP, the RL and the SVM classifiers achieved quite impressive performances. Actually, they obtained more than 85% for the different performance metrics (precision, accuracy, f1 and recall). The results also indicate that the confusions (errors) relate to the two classes (ARC) and (DAA). Furthermore, our future work will focus on extending our dataset and trying other classification techniques especially those related to neural networks and deep neural networks.

Acknowledgment

This research was supported through computational resources of HPC-MARWAN (www.marwan.ma/hpc) provided by the National Center for Scientific and Technical Research (CNRST), Rabat, Morocco.

References

[1] O. F. Zaidan and C. Callison-Burch, “Arabic Dialect Identification,” *Comput. Linguist.*, vol. 40, no. 1, pp. 171–202, 2014, doi: 10.1162/COLI_a_00169.

[2] M. El-Haj, P. Rayson, and M. Aboelezz, “Arabic dialect identification in the context of bivalency and code-switching,” in *LREC 2018 - 11th International Conference on Language Resources and Evaluation*, 2019, pp. 3622–3627.

[3] F. Huang, “Improved Arabic dialect classification with social media data,” *Conf. Proc. - EMNLP 2015 Conf. Empir. Methods Nat. Lang. Process.*, no. September, pp. 2118–2126, 2015, doi: 10.18653/v1/d15-1254.

[4] O. Moussaoui, Y. El Younoussi, and C. Azroumahli, “Creating a Corpus of Moroccan comments by exploring Facebook,” 2021.

[5] H. Elfardy and M. Diab, *Sentence level dialect identification in arabic*, vol. 2. 2013.

[6] F. Sadat, F. Kazemi, and A. Farzindar, *Automatic Identification of Arabic Language Varieties and Dialects in Social Media*. 2014. doi: 10.3115/v1/w14-5904.

[7] K. Darwish, “Arabizi Detection and Conversion to Arabic,” *ANLP 2014 - EMNLP 2014 Work. Arab. Nat. Lang. Process. Proc.*, pp. 217–224, 2014, doi: 10.3115/v1/w14-3629.

[8] S. Malmasi, E. Refaee, and M. Dras, *Arabic Dialect Identification Using a Parallel Multidialectal Corpus*. 2015. doi: 10.1007/978-981-10-0515-2_3.

[9] A. Alshutayri, E. Atwell, A. Alosaimy, J. Dickins, M. Ingleby, and J. Watson, *Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts*. 2016. [Online]. Available: https://www.aclweb.org/anthology/W16-4826

[10] P. Mishra and V. Mujadia, *Arabic dialect identification for travel and twitter text*. 2019. doi: 10.18653/v1/w19-4628.

[11] A. Aliwy, H. Taher, and Z. AboAltaheen, “Arabic Dialects Identification for All Arabic countries,” *Proc. Fifth Arab. Nat. Lang. Process. Work.*, no. December, pp. 302–307, 2020.

[12] H. Nayel, A. Hassan, M. Sobhi, and A. El-Sawy, “Machine Learning-Based Approach for Arabic Dialect Identification,” *Proc. Sixth Arab. Nat. Lang. Process. Work.*, no. April, pp. 287–290, 2021, [Online]. Available: https://www.aclweb.org/anthology/2021.wanlp-1.34

[13] J. Wu, S. Pan, X. Zhu, Z. Cai, P. Zhang, and C. Zhang, “Self-adaptive attribute weighting for Naive Bayes classification,” *Expert Syst. Appl.*, vol. 42, no. 3, 2015, doi: 10.1016/j.eswa.2014.09.019.

[14] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, no. 85, pp. 2825–2830, 2011, [Online]. Available: http://jmlr.org/papers/v12/pedregosa11a.html

[15] L. K. Ramasamy, S. Kadry, Y. Nam, and M. N. Meqdad, “Performance analysis of sentiments in Twitter dataset using SVM models,” *Int. J. Electr. Comput. Eng.*, vol. 11, no. 3, 2021, doi: 10.11591/ijece.v11i3.pp2275-2284.

[16] A. M. Elmogy, U. Tariq, A. Ibrahim, and A. Mohammed, “Fake Reviews Detection using Supervised Machine Learning,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 1, 2021, doi: 10.14569/IJACSA.2021.0120169.

[17] R. D. Joshi and C. K. Dhakal, “Predicting type 2 diabetes using logistic regression and machine learning approaches,” *Int. J. Environ. Res. Public Health*, vol. 18, no. 14, 2021, doi: 10.3390/ijerph18147346.

[18] J. R. Quinlan, “Induction of Decision Trees,” *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986, doi: 10.1023/A:1022643204877.

[19] A. Cutler, D. R. Cutler, and J. R. Stevens, “Random forests,” in *Ensemble Machine Learning: Methods and Applications*, 2012. doi: 10.1007/9781441993267_5.

[20] J. H. Friedman, “Stochastic gradient boosting,” *Comput. Stat. Data Anal.*, vol. 38, no. 4, pp. 367–378, Feb. 2002, doi: 10.1016/S0167-9473(01)00065-2.

[21] T. T. Ngoc, L. van Dai, and D. T. Phuc, “Grid search of multilayer perceptron based on the walk-forward validation methodology,” *Int. J. Electr. Comput. Eng.*, vol. 11, no. 2, 2021, doi: 10.11591/ijece.v11i2.pp1742-1751.

[22] “HPC.” https://www.marwan.ma/index.php/en/services/hpc (accessed May 17, 2022).