Identify Arabic dialect using ensemble model

Mohammad Bani Younes (mohammad_aliyounes@yahoo.com)
Ajloun National University
Mutaz Bani Younes
Open-Insights
Nour Al-khdour
Tarjama

Research Article

Keywords: Arabic Dialects, Arabic NLP, Ensemble models, Machine Learning, n-grams

Posted Date: April 21st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1564007/v1

Identify Arabic dialect using ensemble model

Mohammad Bani Younes[1*], Mutaz Bani Younes[2†] and Nour Al-khdour[3†]

[1*]Dep. of Computer Science, Ajloun National University, Irbid, Jordan.
[2]Open-Insights, Amman, Jordan.
[3]Tarjama, Amman, Jordan.

[†]These authors contributed equally to this work.

Abstract

This paper proposes a model that classifies the Arabic dialect of 26 different dialects from a given text. We used the dataset provided by MADAR Shared Task on Arabic Fine-Grained Dialect Identification, sub-task 1. 19 teams participated in this task, and the highest accuracy achieved by the winning team was 67.33%. Our model's accuracy outperforms that by 0.785%. The proposed model consists of 3 classifiers and then ensembles the predictions from the classifiers to predict the result. We used both n-grams features and probability predictions from the 6-label dataset provided by the organizers. Our model achieves 68.15% accuracy and 68.01% macro F1-score.

Keywords: Arabic Dialects , Arabic NLP ,Ensemble models, Machine Learning, n-grams

1 Introduction

Natural Language Processing (NLP) is a branch of artificial intelligence (AI). NLP aims to find a complex meaning for the language. This field has many tasks, such as finding the similarity between two words or sentences, or even more complex tasks such as question answering. In the past several years, there has been interest from the researchers in processing the Arabic language. The Arabic language is one of the most spoken languages in the world. The Arabic language is spoken or written in three ways. The first one is the Modern

Standard Arabic (MSA), the second one is Classical Arabic (CA), which is the older style of Arabic, and the third one is dialects. The Arabic dialects are different from each other; for example, the used words are different for each geographical region, or even the word itself can be used in a different pronunciation. This exact situation led to an interest in investigating the features of each dialect in the Arabic language.

Dialectal Arabic classification is challenging due to the limited and unavailability of resources. In the 2019 MADAR project, the organizers provided two datasets distributed on two tasks for Arabic Fine-Grained Dialect Identification. Subtask 1: MADAR Travel Domain Dialect Identification, and subtask 2: MADAR Twitter User Dialect Identification. This research focused on subtask one that provides data for 25 Arab cities and MSA.

We propose a novel approach to identifying the dialect using the available corpus to train our ensemble model. Our model consists of features combinations of n-grams in both word and character level and Term Frequency–Inverse Document Frequency (TF-IDF) n-grams in both word and character level. We used two machine learning classifiers, Multinomial Naive Bayes(MNB) and Logistic Regression(LR), in addition to the OneVsAll strategy with MNB. Our model outperformed the highest accuracy obtained by the winning team in the MADAR Travel Domain Dialect Identification task.

This paper is organized to propose our model to identify the dialect of a piece of text using Machine Learning algorithms. In section II, we overview the previous research in dialect identification. In section III, we mentioned and analyzed the dataset. In section IV, we describe the methodology of this work. In section V we present and discuss the results. Finally, Section VI concludes this research.

2 Related work

Dialect identification is considered a challenge in Arabic Natural Language Processing (NLP) field; most of the researchers focused on Modern Standard Arabic (MSA) besides the difficulty to detect the dialect due to the large number of Arabic dialects that differ in linguistic analyses against MSA [1].

Some previous studies adopted Naive Bayes for dialect identification; Fatiha et al. [2] achieved an accuracy of 98% in identifying 18 dialects by using character-based n-gram Markov language models, specifically bi-gram model, and Naive Bayes classifier. Also, The authors [3] used a voting model over two simple (MNB, BNB) classifiers with two categories of lexical features unigrams as word level and 4, 5 grams as character level.

In [4], the authors generated new training samples using data augmentation; they generated six sentences from each sentence. They used n-gram features in both word and character levels; they also extracted a vector of size 26 for each sentence representing each dialect's language model probability. These features were used with a Multinomial Naive base classifier. The authors mentioned that using deep learning such as RNN, CNN, or BERT will not

achieve reasonable accuracy. Their highest accuracy score was 66.42%, 84.54% in Region-level Accuracy, and 74.71% in Country-level Accuracy.

The authors of [5] achieved the highest accuracy score in MADAR shared task last year. In the beginning, the authors reported their experiments using LSTM and Word2vec in this task, which got 58.33%, 49.74% F1-score, respectively. On the other hand, they achieved 67.31% F1-score, and 67.73% accuracy using several features to train their model; they used words uni-gram, words bi-gram, character level n-grams from 1 to 5 with and without considering space between words, and 26 likelihoods estimated by the 26 unigram language models. These n-gram values came after many experiments on the development_dataset. The final set of features was used with a Multinomial Naive Bayes classifier.

The authors of [6] proposed different models as an ensemble of deep learning models for the Multi Arabic Dialect Applications and Resources (MADAR) dataset. The best model among them achieved an F1 macro averaged score of 65.66%. This ensemble model consists of a neural network trained on Character TFIDF and Word TFIDF, with MNB using log probability averaging.

The authors in [7] participated in DSL 2016 competition [8] where they had to classify the Arabic dialect between six different dialects. The authors achieved the first rank in the competition using an SVM classifier with TF-IDF N-gram features. Their best feature set were character bi-grams, tri-grams, 4-grams, and 5-grams. Their highest accuracy was 51.36% on the test data.

The organizers of the DSL competition [8] published a report about all submissions. They allowed the teams to use outside data to train their models. One of the teams that used outside data increased their accuracy by 3.5% only. They achieved 49.7% using only the provided training data, and 53.2% when they used outside data with the provided training data. Also, it is worth mentioning that one of the teams used deep learning and achieved 43.8% accuracy, which is 7.6% less than the winning team.

The authors in [9] were the first researchers to work on a fine-grained Arabic dialect classification. They classified 25 labels, including the MSA (Modern Standard Arabic). Their best features were a combination of n-gram both on word and char level, in addition to char/word 5-gram language model. These features were used with Multinomial Naive Bayes and achieved a 67.9% accuracy score. They claim that this accuracy can be achieved by using sentences with an average of 7 words for training, while using sentences with 16 words average length can achieve 90% accuracy.

3 Dataset

The dataset is from MADAR Travel Domain Dialect Identification subtask 1 [10]. It is a parallel sentence in the Arabic language from the travel domain; the data was cleaned and balanced in each corpus. MADAR shared task aimed to classify the sentence into the related Arabic dialect or MSA from 26 labels. In our experiment, we used both corpus MADAR-Corpus-26 and

MADAR-Corpus-6 to train our model.

The MADAR-Corpus-26 dataset covers 25 Arab cities and MSA. Also, it was already divided into a train, validation, and test set. Table 1 shows the distribution of the data respectively were 41,600, 5,200, and 5,200 parallel sentences, and the number of instances on each class. The 25 Arab cities: (Aleppo, Alexandria, Algiers, Amman, Aswan, Baghdad, Basra, Beirut, Benghazi, Cairo, Damascus, Doha, Fes, Jeddah, Jerusalem, Khartoum, Mosul, Muscat, Rabat, Riyadh, Salt, Sana'a, Sfax, Tripoli, Tunis).

As for MADAR-Corpus-6, it covers 5 Arab cities and MSA. The 5 Arab cities are Beirut, Cairo, Doha, Tunis, and Rabat. It also was divided into a train and validation set, 54,000, 6,000 respectively parallel sentences, and the number of instances on each class is shown in Table 1.

Table 1 Number of instances in the dataset.

CORPUS	instances	Train	Dev	Test
CORPUS-26	overall #	41,600	5,200,	5,200
	per class	1,600	200	200
CORPUS-6	overall #	54,000	6,000	-
	per class	9,000	1,000	-

The provided data for the first task focuses on the travel domain, and thus, dealing with a specific domain should increase our chances of achieving a high score in the classification task.

4 Methodology

In this section, we present the pre-processing techniques, extracted features, and our good and bad models. The results are obtained after various tests using a different combination of various models and various types of features.

4.1 Deep learning models

Deep learning models are commonly used when working with text because they can handle the complexity of the language and capture the relationships between words within the text. However, in this task, deep learning models did not outperform the basic n-gram with the machine learning models method. We tested different embedding features such as BERT embedding, Aravec, Fasttext. Also, we tested the combination of BERT embedding with n-grams features. None of the embeddings got promising results. Some of the previous work, such as [6], reported that using some of the deep learning methods scored around 50% f1-score. Hence, we did not go further with the deep learning models and focused on improving the results using the standard machine learning models.

4.2 Preprocessing

We experiment with different pre-processing steps, and we ended up deleting the dots "." and the double spaces as a pre-processing step.

Many experiments are tested using our model to find the optimal pre-processing steps. When we trained the model with pre-processed data, the overall model's accuracy decreased. Meanwhile, the overall model's accuracy increased when we trained the models on both pre-processed and raw data. In detail, we trained LR on the pre-processed data; and we trained the OneVsAll MNB and MNB classifier on the raw data. This method increased our overall model by 5.8%.

4.3 Features Extraction

Based on the previous works that listed in section II, most of the research used N-gram features to train models for Arabic dialects classification such as [2] [5] [9] and TF-IDF [6]. As well [7] used a combination of them. We built our best model based on many experiments on different combinations of n-grams [11], and TF-IDF n-grams [12] in both word and character level. And then, we used FeatureUnion from sklearn [13] to concatenate all extracted features into a sparse representation.

Table 2 illustrates our experiments for this phrase, our final set of features are explained as follows:

4.3.1 TF-IDF n-grams

TF-IDF is a statistical measure that represents the importance of the words or characters in documents regards the corpus [12]. It generates a vector from the text by determining the parameters of the 'analyzer' for words or character (char), in addition to the 'char_wb' option, which generates character n-grams only from text inside word boundaries. Our experiments used TF-IDF n-grams features and set transformation weight for each one. For character level, we used a transformation weight of 0.9, while for the character level with word boundary, we used a transformation weight of 1.1 and (1-5) n-gram ranges for both of them. As shown in Table 2 the highest accuracy of 67.19 for test data achieved using TF-IDF 'char_wb' for (1-5) n_grams, and TF-IDF 'char' for (1-5) n_grams.

4.3.2 N-Grams

N-Grams [11] is a probabilistic language model that generates a contiguous sequence that reflects the occurrence of words in a text in a specific size of a window such as a unigram for size 1, bigram for size 2, or trigram for size 3. In our experiments, we create ngrams for word level with a transformation

weight of 0.6. The range of n-gram from 1 to 2, as shown in Table 2 the highest score achieved using the unigrams and bigrams (1-2).

N-Grams Features			Accuracy		F1-Score	
TFIDF char_wb	TFIDF char	N-Gram word	DEV	TEST	DEV	TEST
-	-	1-2	62.73	62.88	62.75	62.79
-	-	1-3	62.53	62.61	62.53	62.54
-	-	1-4	62.38	62.42	62.38	62.34
1-3	-	-	59.35	57.02	59.08	56.99
1-4	-	-	64.04	62.48	63.87	62.52
1-5	-	-	65.17	63.15	65.10	63.21
-	1-3	-	58.17	55.67	57.89	55.60
-	1-4	-	62.92	62.35	62.75	62.32
-	1-5	-	64.29	63.77	64.19	63.68
1-3	1-3	1-2	66.94	66.06	66.86	66.05
1-4	1-4	1-2	68.40	67.02	68.32	67.00
1-5	1-5	1-2	**68.67**	**67.19**	**68.70**	**67.20**

Table 2 This table shows MNB results using the following transformer weights 1.1, 0.9, and 0.6, respectively. These numbers are reported without using the six label probability features

4.4 Model Architecture

The following is a brief explanation of the machine learning classifiers used in our ensemble model:

Multinomial NB [14] Multinomial Naïve Bayes is a specific instance from the Naive Bayes classifier considered to be a supervised learning algorithm. Naïve Bayes [15] is derived from Bayes Theorem; it is used for calculating the conditional probability for each of the features in the model, the probability that something will happen, given that something else has already occurred. while Multinomial Naive Bayes classifier uses a multinomial distribution for each of the features [16].

Logistic Regression (LR) [17] [18] It is a machine learning algorithm for regression and classification. Logistic regression is commonly used for a binary classification task and can be used for multi-class classification. LR uses the logistic function to model a binary model and make the predicted value binary (either 0 or 1), the sigmoid function is applied to squash the values from values in the range (0-1) into binary. LR for multi-class classification can be done by setting 'ovr' as the 'multi_class' parameter. Based on that LR uses the one-vs-rest (OvR) scheme for classification, OvR is a model applied as a classifier for each class independently, which converts the problem into a binary classification as it predicts an example whether it belongs to a specific class or not, this process is repeated for all the classes, then it chooses the

class with the highest score for each example.

OneVsAll strategy This method's main idea is converting a multi-class classification problem into a binary classification problem. This is done by considering one class as the first class and the other classes as the second class.

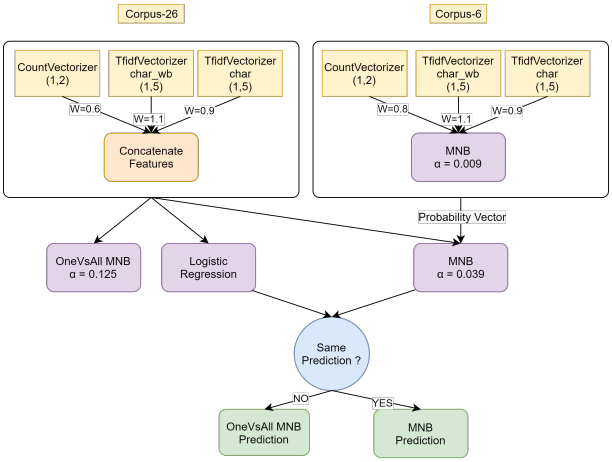

Fig. 1 Our proposed Model

For training the model, we used all the provided data from MADAR corpus-6 and corpus-26; the difference between them is illustrated in section III. In the beginning, we used corpus-6 to generate a six-label probability for the 26-corpus data, then we extracted the best features based on many experiments conducted as represented in Table 2. The combination of features with the transformation weight as follows: N-grams (unigrams and bigrams) shown in Figure 1 as CountVectorizer (1,2) with a transformation weight of 0.8, and TF-IDF n-grams (1,5) for character level as shown in Figure 1 as TfidfVectorizer char (1,5) with a transformation weight of 0.9, and TF-IDF n-grams (1,5) for char level with word boundaries as shown in Figure 1 as TfidfVectorizer char_wb (1,5) with a transformation weight of 1.1. Then we used MNB with an alpha value equal to 0.009 to predict the probability of the six classes for the Corpus-26 training and testing data. This model achieved an accuracy of 92.5%. Because of this high accuracy, these six numerical features increase our overall accuracy for the corpus-26.

At the same time, for corpus-26, we extracted the same features and transformation weights we used for corpus-6 except for the CountVectorizer (1,2); we used transformation weights of 0.6 instead of 0.8. Then we concatenated the features to be used for training the model. After that, we pass the concatenated features into our ensemble model, consisting of OneVsAll MNB with an alpha value equal to 0.125, Logistic Regression, and MNB with an alpha value

Model	Parameters	Accuracy	Precision	Recall	F1-Score
MNB	alpha = 0.039	67.46	67.54	67.46	67.37
OneVsAll MNB	alpha = 0.125	67.17	67.35	67.17	67.04
LR	multi_class='ovr' penalty='l2' tol=0.0001	63.46	65.28	63.46	63.48
Ensemble	-	**68.15**	**68.21**	**68.15**	**68.01**

Table 3 A comparison between each model's performance and the performance of the ensemble model. These results are reported using the test set.

equal to 0.039. The MNB with an alpha value equal to 0.039 is also trained on the probability vector from the previous model of corpus-6. Finally, we compare the results from both models, Logistic Regression and MNB; if they yield the exact prediction, then the prediction will be the MNB prediction; if not, the prediction will be the OneVsAll MNB predictions.

5 Discussion

Various classifiers were tested in order to find the highest accuracy. Our last selected classifiers are Multinomial NB with an alpha value of 0.039, One Vs. Rest Classifier with Multinomial NB with an alpha value of 0.125, and Logistic Regression. As shown in Figure 1, our ensembling method is simple; we consider our final prediction in two ways. The first one is that if the LR and MNB have the exact prediction, this would be our final prediction. Otherwise, we will consider the OneVsAll MNB prediction as our final prediction.

For example, we tested a sentence that was written in Moroccan delicate:
In MSA :
In Marrocon:

Multinomial NB classifier output is "Rabat", Logistic Regression classifier output is "Khartoum", and the One Vs. Rest Classifier with Multinomial NB output is "Rabat". Since the prediction from MNB and LR is not the same prediction, the final prediction will be OneVsAll MNB prediction which is "Rabat".

In the beginning, we trained our model without pre-processing the data; the final model did achieve a good accuracy of 67.75%. Then we tested the model with the pre-processed data in more than one way. The first one is to train one classifier on the pre-processed data while training the others on the raw data, and so on. Our final result was achieved by training LR on the pre-processed data and training the OneVsAll MNB and MNB on the raw data.

There is a slightly improvement on the results once we used the ensemble model over using the MNB classifiers alone or the LR classifier alone. The following table 3 presents a comparison between the results of using the ensemble models against using each model alone on the development data.

It is worth mentioning that the "alpha" parameter plays a huge role in improving the results for the MNB classifier

Team	Accuracy	F1	Precision	Recall
Our team	**68.15**	**68.01**	**68.21**	**68.15**
[19]	67.29	67.32	67.6	67.29
[5]	67.33	67.31	67.73	67.33
[20]	67.03	67.2	67.53	67.08
[10]	66.48	66.31	66.68	66.48
[21]	66.31	66.28	66.56	66.31
[9]	67.75	67.89	68.41	67.75
[22]	64.75	64.74	65.01	64.75

Table 4 Comparison between our proposed ensemble model and the top 5 teams participated in the MADAR competition. The last two rows are for the baselines

6 Results

Our ensemble model outperformed the results obtained by the MADAR competition participants, below is table 4 that shows the comparison between our score and their score on the same dataset.

7 Conclusion

This paper proposes an enhancement over the previous work on the Arabic dialect detection field. We used MADAR shared task data to train and test our model. Our best model achieves 68.1% accuracy. This model was built after many experiments to develop the best combination of classifiers, pre-processing steps, parameters, weights, and features.

Our ensemble model consists of MNB, Logistic Regression, and One Vs. All MNB. It is trained on a combination of extracted features with weights. Such as count vectorizer unigrams and bigrams with 0.6 as a weight for corpus-26 and 0.8 as a weight for corpus-6, TF-IDF character level (1-5) n_grams with 0.9 as a weight, and TF-IDF character level (1-5) n_grams with word boundaries with 1.1 as a weight. The final prediction depends on the equality of MNB and both Logistic Regression and One Vs. All MNB. If it is equal, then the final prediction is the MNB prediction, and if it is not equal, then the final prediction is One the Vs. All MNB predictions.

8 Future Work

This paper used one of the basic yet reasonably performing methods to detect a given text's dialect in the Arabic language. Many state-of-the-art models were proposed to the machine learning and AI world in recent years, such as BERT. For future work, we will consider training our own BERT model on a domain-specific dataset to make it understand the Arabic language in that specific domain better than the standard BERT models.

9 Compliance with Ethical Standards

The author(s) declare no competing interests.

The research does not involve Human Participants and/or Animals.

References

[1] Al-Sabbagh, R., Girju, R.: Yadac: Yet another dialectal arabic corpus. In: LREC, pp. 2882–2889 (2012)

[2] Sadat, F., Kazemi, F., Farzindar, A.: Automatic identification of arabic language varieties and dialects in social media. In: Proceedings of the Second Workshop on Natural Language Processing for Social Media (SocialNLP), pp. 22–27 (2014)

[3] Eltanbouly, S., Bashendy, M., Elsayed, T.: Simple but not naïve: Fine-grained arabic dialect identification using only n-grams. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop, pp. 214–218 (2019)

[4] Talafha, B., Fadel, A., Al-Ayyoub, M., Jararweh, Y., Mohammad, A.-S., Juola, P.: Team just at the madar shared task on arabic fine-grained dialect identification. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop, pp. 285–289 (2019)

[5] Meftouh, K., Abidi, K., Harrat, S., Smaïli, K.: The smart classifier for arabic fine-grained dialect identification. (2019)

[6] Fares, Y., El-Zanaty, Z., Abdel-Salam, K., Ezzeldin, M., Mohamed, A., El-Awaad, K., Torki, M.: Arabic dialect identification with deep learning and hybrid frequency based features. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop, pp. 224–228 (2019)

[7] Eldesouki, M., Dalvi, F., Sajjad, H., Darwish, K.: Qcri@ dsl 2016: Spoken arabic dialect identification using textual features. In: Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3), pp. 221–226 (2016)

[8] Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J.: Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. In: Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3), pp. 1–14 (2016)

[9] Salameh, M., Bouamor, H., Habash, N.: Fine-grained arabic dialect identification. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 1332–1344 (2018)

[10] Bouamor, H., Hassan, S., Habash, N.: The madar shared task on arabic fine-grained dialect identification. In: Proceedings of the Fourth Arabic

Natural Language Processing Workshop, pp. 199–207 (2019)

[11] Damashek, M.: Gauging similarity with n-grams: Language-independent categorization of text. Science **267**(5199), 843–848 (1995)

[12] Mayfield, J., McNamee, P.: Indexing using both n-grams and words. In: TREC, pp. 361–365 (1998)

[13] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)

[14] Small, K.A., Hsiao, C.: Multinomial logit specification tests. International economic review, 619–627 (1985)

[15] Rish, I., *et al.*: An empirical study of the naive bayes classifier. In: IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, vol. 3, pp. 41–46 (2001)

[16] Intelligence, A.: A modern approach. by Stuart Russell and Peter Norvig (2003)

[17] Wright, R.E.: Logistic regression. (1995)

[18] Kleinbaum, D.G., Dietz, K., Gail, M., Klein, M., Klein, M.: Logistic Regression. Springer, ??? (2002)

[19] Abu Kwaik, K., Saad, M.K.: Arbdialectid at madar shared task 1: Language modelling and ensemble learning for fine grained arabic dialect identification. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop (2019). Association for Computational Linguistics

[20] Ragab, A., Seelawi, H., Samir, M., Mattar, A., Al-Bataineh, H., Zaghloul, M., Mustafa, A., Talafha, B., Freihat, A.A., Al-Natsheh, H.: Mawdoo3 ai at madar shared task: Arabic fine-grained dialect identification with ensemble learning. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop, pp. 244–248 (2019)

[21] Mishra, P., Mujadia, V.: Arabic dialect identification for travel and twitter text. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop, pp. 234–238 (2019)

[22] Zaidan, O.F., Callison-Burch, C.: Arabic dialect identification. Computational Linguistics **40**(1), 171–202 (2014)

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- MadarSpringer.zip