Word Representation Models for Arabic Dialect Identification

Mahmoud S. Ali **Ahmed H. Ali** **Ahmed A. El-Sawy** **Hamada A. Nayel**
Department of Computer Science
Faculty of Computers and Artificial Intelligence
Benha University
{mahmoud.hassan,ahmed.ali,ahmed.el_sawy,hamada.ali}@fci.bu.edu.eg

Abstract

This paper describes the systems submitted by BFCAI team to Nuanced Arabic Dialect Identification (NADI) shared task 2022. Dialect identification task aims at detecting the source variant of a given text or speech segment automatically. There are two subtasks in NADI 2022, the first subtask for country-level identification and the second subtask for sentiment analysis. Our team participated in the first subtask. The proposed systems use Term Frequency Inverse/Document Frequency and word embeddings as vectorization models. Different machine learning algorithms have been used as classifiers. The proposed systems have been tested on two test sets: Test-A and Test-B. The proposed models achieved Macro-f1 score of 21.25% and 9.71% for Test-A and Test-B set respectively. On other hand, the best-performed submitted system achieved Macro-f1 score of 36.48% and 18.95% for Test-A and Test-B set respectively.

1 Introduction

Social media's widespread use has made it easy to collect user data in surpassing ways. These data can include behaviour and usage, content, and network (Abdul-Mageed et al., 2020). This work focuses on predicting social media user dialect based on language of his/her post. Dialect identification task comprises of some challenges such as finding the differences in writing style between men and women on social networks, ages of authors, or location (Abdul-Mageed et al., 2021b). The solutions to these questions are very important for new problems in the era of social networks such as fake news analysis, plagiarism detection, privacy and security issues.

The author profiling task aims at examining the written documents to extract pertinent demographic information from their authors (Aliwy et al., 2020). Lately, the research community concerning Arabic natural language processing started to pay attention to dialect identification. Nuanced Arabic Dialect Identification shared task (NADI 2021) aimed at predicting the dialect in Arabic Tweets (Abdul-Mageed et al., 2021b).

This work explores different vectorization techniques integrated with the various machine learning approaches. Term Frequency/Inverse Document Frequency (TF/IDF) and word embeddings have been used as vectorization models. Multinomial Naïve Bayes (MNB), Complement Naïve Bayes (CNB), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Decision Tree (DT), Random Forest (RF), and Multi-Layer Perceptron (MLP) have been used as classifiers due to their ability to deal with multi-class Classification problems.

The rest of the paper is organized as follows: section 2 presents the dataset; section 3 describes the system architecture. Experimental settings and results are given in section 4. Finally, conclusion and future work are presented in section 5.

2 Data

The NADI 2022 datasets that we used for building, developing, and evaluating the submitted systems were distributed by the task organizers (Abdul-Mageed et al., 2022).

The dataset targets nuanced Arabic dialect identification at the country level for Arabic tweets. It comprises training, development, and test sets. It covers 18 dialects (a total of approximately 20K tweets). The evaluation depends on two test sets, Test-A covers 18 country-level dialects, whereas the second test set (TEST-B) covers k country-level dialects. The value of k was kept unknown by the

Proceedings of the The Seventh Arabic Natural Language Processing Workshop (WANLP), pages 474 - 478
December 8, 2022 ©2022 Association for Computational Linguistics

task organizers

3 System Architecture

The general framework of our model is shown in Figure 1. The model consists of three main phases. The first phase is preprocessing where the raw data was prepared to further steps. The second phase is feature extraction and the third phase is training the model. After model construction, test set was fed to the model for model evaluation. The following are details of each phase.

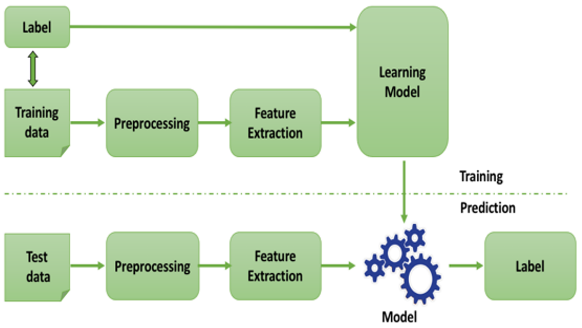

Figure 1: General architecture of the proposed models

3.1 Text Preprocessing

Text data sourced or generated by social media are unstructured and very noisy data. To overcome this issue some non-informative data or texts are removed such as emojis, Latin-characters, URLs, mentions, numbers, and non-Arabic characters. The preprocessing steps based on the work done by Nayel (2020); Ashraf et al. (2022a,b), have been applied to the tweets in detail:

- *Removing Non-Arabic letters* by deleting English letters, special symbols, numbers, Twitter markup, and Emoticons.
- *Text Normalization* by refining text to normalize different forms of some Arabic characters to unique form like, " ة " (an Arabic letter pronounced Haa) and " ه" to be " ه", removing redundant Arabic forms like, " ية" (pronounced al and it is used as determiner).
- *Removing punctuation marks* such as $\{'+','\#','-','\$',...\}$ which increase the redundant features resulting a huge feature space dimension.
- *Decreasing the letter repetition*, cleaning the tokens from the redundant letters helps in reducing feature space. In our work, the letter is assumed to be redundant if it is repeated more than two times. For example, the word " عاااالم " (i.e., "global" will be reduced to " عالم ", also the word " رااائع " (i.e., "wonderful") will be reduced to " رائع "

3.2 Feature Extraction

In this work, TFI/DF and word embeddings (Word2Vec) vectorization algorithms were used with unigram features (words or tokens) to describe each tweet as a feature vector.

3.2.1 TF/IDF

In TF/IDF, the value of each component in this vector represents the weight of the corresponding feature (word) within a tweet. Assuming the vocabulary set V=v_1, v_2, …., v_k that contains the unique tokens appeared in the corpus. Then, the tweet T_i can be represented as the following vector $T_i = < t_{i1}, t_{i2}, t_{i3}, \ldots, t_{ik} >$ and is calculated by the following formula:

$$t_{ij} = tf_{ij} * \log\left(\frac{N+1}{df_i+1}\right)$$

Where, t_{ij} is the weight of a word j in tweet i, tf_{ij} is the count of word j in tweet i, N is the total number of tweets, and df_i is the count of word i in all tweets. We used unigram model in TF/IDF algorithm, in which each feature is a single word (token). For example, the sentence means which " جمعة مباركة تصبح على خير " "Happy Friday good night", has the following set of features (tokens) " جمعة " , " مباركة ", " تصبح ", " على ", " خير "

3.2.2 Word Embeddings

Another approach for word representation is word embeddings (Mikolov et al., 2013). One of the most effective embeddings model is Word2vec. Word2vec has a neural network structure, proposed by Google, to processes the text data. Word2Vec includes two learning models, Continuous Bag of Words (CBOW) and Skip-gram. CBOW predicts the word given its context, but Skip-gram predicts the context given a word. Word2Vec generates the word vectors through feeding the text corpus (which was available in this task) to one learning model.

First, Word2Vec builds a vocabulary from training corpus, which obtained from NADI 2022 subtask1, and learns the vector representations of each word. Also, Word2Vec calculates the cosine distance among each word. We implemented Word2Vec using gensim, which is a python library. First, we used the vocabulary from the entire training data. Then, to generate the word vectors, we employ the CBOW as it has higher computing speed than Skip-gram. After training step, each word is represented by a vector.

Then, a high dimension matrix has been constructed. Each row in matrix represents a training sample and columns represent the generated word vectors. Now, each word has multiple degrees of similarity, it can be computed via a linear calculation.

After we create the feature vector matrix of all training samples using the two algorithms, we go to the classification step, which will be described

3.3 Classification

In this work, the classification step was accomplished by applying seven classifiers. Then comparing the performance of each classifier and the best performed classifier was chosen to submit. Word2vec and TF/IDF have been used to represent the tweet tokens for each classifier. The following list is the classifiers have been used in this model:

- *The Complement Naive Bayes (CNB)* classifier was designed to correct the "severe assumptions" accomplished by the standard Multinomial Naïve Bayes (MNB) classifier. It is particularly suitable for imbalanced datasets, and this is proved in our results.

- *Support Vector Machine (SVM)* is a linear classifier which uses training examples or support vectors close to the boundaries of classes. SVM also can be used for classifying nonlinear data using kernel functions such as, Linear, and RBF, which were used in this work.

- *K-NN* algorithm suppose that the similarity between the new example and available examples and put the new one into the category that is most similar to the available categories.

- *Decision Tree (DT)* classifier depends on the decision tree as a predictive model to go from observations about an item which represented in its branches to conclusions about the item's target value which represented in its leaves.

- *Random Forest (RF)* is a meta estimator that fits several decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

- *Multi-Layer Perceptron (MLP)* is a fully connected class of feedforward Artificial Neural Network (ANN). An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function

4 Experiments and Results

We proposed seven classifiers with TF/IDF, and five classifiers with Word2Vec. All algorithms were implemented on NADI 2022 shared task dataset for subtask1.

We calculated four evaluation metrics, Accuracy (Acc), Precision (P), Recall (R), and F1-score to measure the performance of our models. The macro-averaged f1-score is the official metric for subtask1.

Table 1 and Table 2 show the results for all runs of the development set classification using TF/IDF and Word2Vec representations respectively.

We implemented SVM with two different kernels, linear kernel and Radial Bases Function (RBF). Different numbers of hidden layers (h =10, 20, 30 and 40) have been implemented in MLP.

From Table 1 and Table 2, it is clear that MLP and CNB outperforms all other classifiers. We decided to submit the output of CNB, MLP (h = 20) and MLP (h = 30).

Table 3 and Table 4 shows the results of our submissions on Test-A and Test-B of subtask 1 respectively. For test-A set, MLP with 30 hidden layers and word embeddings (WE) outperforms all other classifiers. While accuracy of CNB with TF/IDF outperforms all other accuracies.

For test-B set, MLP with 30 hidden layers and word embeddings (WE) outperforms all other classifiers. While precision of CNB with TF/IDF outperforms all other precisions.

Algorithm	Accuracy	Precision	Recall	F1 (macro)
MNB	30.158	21.508	9.763	7.567
CNB	39.068	24.321	**19.893**	**20.475**
SVM (Linear)	**39.643**	**34.482**	14.893	13.407
SVM (RBF)	37.323	36.247	14.893	13.407
KNN	33.833	29.311	13.771	13.178
DT	25.662	12.740	12.005	11.920
RF	34.675	21.493	14.637	14.102
MLP (10 H)	31.102	16.984	16.181	16.222
MLP (20 H)	32.745	19.135	17.536	17.852
MLP (30 H)	32.622	18.276	17.328	17.457
MLP (40 H)	32.478	18.863	17.348	17.601

Table 1: Performance measure of the different classifiers on development set using TF/IDF for subtask 1.

Algorithm	Accuracy	Precision	Recall	F1 (macro)
SVM (Linear)	40.135	22.736	20.033	19.843
SVM (RBF)	**42.620**	**32.166**	14.647	12.804
KNN	35.024	25.217	14.647	12.804
DT	17.984	8.883	8.856	8.859
RF	37.056	18.579	13.937	11.162
MLP (10 H)	40.731	19.315	20.264	19.029
MLP (20 H)	38.883	21.710	**20.440**	**20.188**
MLP (30 H)	37.590	21.041	20.179	20.023
MLP (40 H)	36.769	20.414	19.740	19.620

Table 2: Performance measure of the different classifiers on development set using Word2vec model for subtask 1.

Algorithm	**Acc**	**P**	**R**	**Macro F1**
MLP(30)+WE	38.63%	**25.25%**	20.47%	**21.25%**
CNB+TF/IDF	**39.05%**	22.81%	**21.30%**	21.16%
MLP(20)+WE	38.97%	24.58	21.19 %	21.13%

Table 3: Results of our submissions on Test-A of Subtask 1.

Algorithm	**Acc**	**P**	**R**	**Macro F1**
MLP(30)+WE	**23.13%**	14.54%	**11.99%**	**9.71%**
MLP(20)+WE	22.73%	**16.88%**	11.80%	9.14%
CNB+TF/IDF	21.23%	11.41%	10.45%	7.78%

Table 4: Results of our submissions on Test-B of Subtask 1.

5 Conclusion and Future Work

In this paper, a simple framework for dialect identification has been introduced. Two main vectorization approaches (TF/IDF and Word Embeddings) have been compared. It is clear from results that word embeddings outperforms TF/ID. From this study, we can conclude that dialect identification of Arabic text is one of the most challenging tasks. The results of training using MLP (h=20 and h=30) with Word2Vec model achieved the best F1 macro-averaged score as the power of word embeddings in NLP. CNB with TF/IDF comes in the second as it can deal with unbalanced text data.

In future work, pre-trained models could be used to improve the performance of classification, such as BERT (Devlin et al., 2019), AraBERT (Antoun et al., 2020), MarBERT model (Abdul-Mageed et al., 2021a). Transfer learning can be applied that knowledge from one domain can be transferred to another domain.

References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021a. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7088–7105, Online. Association for Computational Linguistics.

Muhammad Abdul-Mageed, Chiyu Zhang, Houda Bouamor, and Nizar Habash. 2020. NADI 2020: The first nuanced Arabic dialect identification shared task. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 97–110, Barcelona, Spain (Online). Association for Computational Linguistics.

Muhammad Abdul-Mageed, Chiyu Zhang, AbdelRahim Elmadany, Houda Bouamor, and Nizar Habash. 2021b. NADI 2021: The second nuanced Arabic dialect identification shared task. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 244–259, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Muhammad Abdul-Mageed, Chiyu Zhang, AbdelRahim Elmadany, Houda Bouamor, and Nizar Habash. 2022. NADI 2022: The Third Nuanced Arabic Dialect Identification Shared Task. In *Proceedings of the Seven Arabic Natural Language Processing Workshop (WANLP 2022)*.

Ahmed Aliwy, Hawraa Taher, and Zena AboAltaheen. 2020. Arabic dialects identification for all Arabic countries. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 302–307, Barcelona, Spain (Online). Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Nsrin Ashraf, Fathy Elkazzaz, Mohamed Taha, Hamada Nayel, and Tarek Elshishtawy. 2022a. BFCAI at SemEval-2022 task 6: Multi-layer perceptron for sarcasm detection in Arabic texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 881–884, Seattle, United States. Association for Computational Linguistics.

Nsrin Ashraf, Hamada Nayel, and Mohamed Taha. 2022b. A comparative study of machine learning approaches for rumors detection in covid-19 tweets. In *2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pages 384–387.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Hamada Nayel. 2020. NAYEL at SemEval-2020 task 12: TF/IDF-based approach for automatic offensive language detection in Arabic tweets. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2086–2089, Barcelona (online). International Committee for Computational Linguistics.