Arabic Dialect Identification of Twitter Text Using PPM Compression

Mohammed Altamimi *Mh.altamimi@uoh.edu.sa*
College of Computer Science and Engineering
University of Hail
Hail, Saudi Arabia

William J. Teahan *W.j.teahan@bangor.ac.uk*
School of Computer Science and Electronic Engineering
University of Bangor
Bangor, United Kingdom

Abstract

This paper explores the use of the Prediction by Partial Matching (PPM) compression scheme for Arabic dialect identification of Twitter text. The PPMD variant of the compression scheme with different orders was used to perform the categorisation. We present experimental results identifying single tweet and multiple author tweets from five major Arabic dialect regions: Gulf; Egyptian; Levantine; Maghrebi; and Iraqi; in addition to Modern Standard Arabic (MSA) and Classical Arabic (CA). We used the Bangor Twitter Arabic corpus (BTAC) which we built for dialect research. We also applied different machine learning algorithms such as Multinomial Naïve Bayes (MNB), K-Nearest Neighbours (KNN), and an implementation of Support Vector Machine (LIBSVM) using several N-grams features. PPMD shows significantly better results in comparison to the other machine learning algorithms achieving 74.1% and 87.1% accuracy for single and multiple tweets dialect identification respectively.

Keywords: Arabic Dialect Identification, Data Compression, Machine Learning, Natural Language Processing.

1. INTRODUCTION

Arabic dialects are the spoken variations of Arabic language which differ according to the geographical region. Although the written form of Arabic is still considered the standard form of the language which is used formally, a dialect is more often used unofficially among people from a specific region on a daily basis. Recently, it has been noticed that the written form of dialects is being used more frequently for informal written communication on the web (Hamdi et al. 2015). However, most of the early research on languages has involved identification between the main languages. Dialect identification in social media is a recently emerging phenomenon in the field of text categorisation. Meanwhile, a common way to identify the source of a tweet is to trace the location of the tweet (latitude and longitude). The problem is that researchers rely heavily on the location of the tweet rather than on identifying the dialect from the content of the text. With the existence of many fake accounts and bots in social media which use hidden locations to spread rumours or start political propaganda, it is becoming vital to study the language used besides other elements to identify the actual source of the tweets rather than relying on misleading location information.

The rest of the paper is organized as follows: section two provides the background and related work on dialect identification with an emphasis on Arabic dialects studies; section three describes the dataset used in the experiments; section four discusses our methodology; section five lists our experimental results; section six discusses our findings; and finally, section seven provides the conclusion and suggests future work.

2. RELATED WORK

In the case of the English language, Lui and Cook (2013) investigated cross-domain three-way national dialect classification between Australian, British and Canadian English. Their results demonstrated that there are lexical and syntactic characteristics of each national language variation that exist across several data sources such as web data, web government pages, and tweets. They found that the SVM classifier using bag-of-words outperformed features based on syntax or character sequences when differentiating between Australian, British and Canadian English.

Ljubesic et al. (2007) used a character N-gram model in combination with a most frequent words list to distinguish between Croatian, Serbian and Slovenian-related languages using 13 thousand documents. Their study achieved high accuracies of over 99%. This research led to further work by Tiedemann and Ljubešić (2012) on investigating Bosnian, Croatian, and Serbian-related languages using a total of 600 documents. They performed an experiment using a Naïve Bayes classifier with word unigram features, achieving accuracies of 95%. More recently, Ljubešić and Kranjčić (2015) distinguished Twitter users by language using very similar South-Slavic languages – Bosnian, Croatian, Montenegrin and Serbian. They applied the supervised machine learning approach by annotating a subset of 500 users from an existing Twitter account collected by user language. They showed that by using a simple bag-of-words model, and the univariate feature, they were able to achieve a 98% user classification accuracy using Multinomial Naïve Bayes.

Zampieri and Gebre (2012) explored computational techniques for automatic dialect identification of two variations of the Portuguese language – Brazilian Portuguese and European Portuguese. A character-based model that used 4-grams was reported to perform best compared to character N-grams of other lengths from 1-6. They used data collected from newswire containing one thousand documents divided between the two variations, and reported an accuracy of 99.8%. Later, Zampieri et al. (2013) applied an N-gram language model on four Spanish variations, Espagne, Argentine, Mexique and et Pérou, with a total of one thousand documents from newswires. They found that word 2-grams outperformed character N-grams of any length from 1 to 5. They also found that binary classification settings achieved significantly better results reporting an accuracy of 96.9% whereas, in comparison to the 4-way classification, this achieved an F-measure of 0.876.

In the Chinese context, Xu et al. (2017) performed 6-way, 3-way, and 2-way classifications in various greater Chinese dialects such as Mainland China, Hong Kong, Taiwan, Macao, Malaysia, and Singapore. They found that character bi-grams and segmented words work much better in Chinese then character unigrams do. This indicates that such longer units are more meaningful in Chinese and can better reflect the characteristics of a dialect. They performed 6-way classification via the linear kernel support vector machine using the LIBLINEAR library, achieving an accuracy of 82% on a total of 15 thousand text sentences collected from newswires.

In the Indian language, Kumar et al. (2018) identified Indian variations of Modern Standard Hindi (MSH), Braj, Awadhi, Bhojpuri, and Magahi using 10 thousand sentences of each variation. The study demonstrated that character N-gram were more effective than word N-gram features. However, combining both character and word n-grams led to better results, achieving an accuracy of 96.4%.

2.1 Arabic Language Research

Arabic Dialect identification is a crucial topic for most Arabic NLP research because of the diversity of Arabic dialects and the fact that Arabic is spoken in 20 different countries in the Middle Eastern region. Most of the early work on Arabic focused only on Modern Standard Arabic. Recently, there has been an increase in studies focusing on Arabic dialect identification due to the availability of NLP tools and resources that support the Arabic language.

The early work on Arabic was started by Zaidan and Burch (2011) which is considered one of the first attempts to investigate the Arabic dialects in depth. They created an Arabic Online Commentary dataset with a total of 108 thousand sentences which were labelled for MSA and three dialects – Levantine, Gulf, and Egyptian. The study reported an accuracy of 69.4% from a 4-way classification. However, for a 2-way classification using character-based N-gram and word-based N-gram features between Egyptian Arabic and MSA, the accuracy reached 87.9% using word-based unigrams (Zaidan and Callison-Burch, 2014). Likewise, the study by Elfardy and Diab (2013) performed 2-way classification of the Egyptian dialect and MSA using the AOC dataset. They applied the Naïve Bayes classifier using tokenisation to the sentence level, scoring an accuracy of 85.5%.

Darwish et al. (2014) performed a study of 2-way classification on Egyptian dialects and MSA. The study included a range of lexical and morphological features to classify a total of 700 tweets annotated evenly between the two variations. The accuracy reached 95% using the Random Forest classifier. Moreover, the research by Malmasi et al. (2015) examined a 6-way classification task using two thousand sentences of a multidialectal Arabic dataset (Bouamor et al. 2014). Various character-based N-gram and word-based N-gram features were examined. The best result showed that by using the LIBLINEAR SVM classifier combining all character and word features, an accuracy of 74% was achieved.

El Haj et al. (2018) presented experimental results from automatically identifying dialects. They performed 5-way classification tasks with a total of 16 thousand sentences using SVM. The study used subtractive bivalency profiling features combined with grammatical and stylistic features. The results showed that their classification methods can reach more than 76% accuracy using 10-fold cross validation. Also, they tested on completely unseen data using SVM, and achieved an accuracy of 66%.

Furthermore, Sadat et al. (2014) conducted an experiment using Markov models. Their result showed that the Naïve Bayes classifier performs better than the character N-gram Markov models for most Arabic dialects. The experiment was performed using data collected from 18 Middle Eastern countries with a total of 63 thousand sentences. They reported an accuracy of 98% at distinguishing among all the dialectal datasets. The study by Alshutayri and Atwell (2017) reported a classification accuracy of 79%. They performed the classification using Multinomial Naïve Bayes (MNB) using the WordTokenizer feature in Weka. Their training data contained 8,090 tweets, and testing on 1,764 tweets divided unequally between the five main Arabic dialects.

3. EXPERIMENTAL DATASET

The dataset that we built for our research covers five major Arabic dialect groups: Gulf; Egyptian; Levantine; Maghrebi; and Iraqi; in addition to Modern Standard Arabic (MSA) and Classical Arabic (CA). The Bangor Twitter Arabic corpus (BTAC) which is purposely designed for dialect research (Altamimi et al. 2018). The corpus contains over 120K tweets annotated according to the different Arabic dialects. The corpus is collected from 101 authors from the Middle East, with all the tweets manually annotated according to the dialects and verified independently by two experts. An explicit testing set has been created for testing purposes rather than splitting the training set. The test set was collected from the same users for three different time periods to verify that there is no overlap between the training and testing set, and to reflect a real-world data collection scenario. The total number of tweets collected for each dialect including training and testing sets are listed in Table 1.

Dialects	MSA	CA	Gulf	Egyptian	Mixed	Levantine	Maghrebi	Iraqi
Train	42,658	31,006	9,148	9,057	8,343	7,857	3,980	1,884
Test	3,395	1,484	621	567	149	364	167	86

TABLE 1: Breakdown of the tweets used for the dialect identification experiments.

4. METHODOLOGY

The Prediction by Partial Matching (PPM) text compression technique for lossless data is based on the adaptive context modelling family which uses a fixed number of preceding characters according to a selected maximum fixed order to predict the coming character. For example, if the selected maximum order is three, the prediction of the following character will be based on the previous three characters. PPM moves from the maximum highest order down to lower orders using the escape mechanism whenever a previously unseen symbol is encountered. This process will be continued until the lowest default order of -1 is reached, where all character probability are equiprobable. It has shown excellent performance in many natural language processing tasks, such as text correction and language identification (Teahan & Cleary 1997).

PPM has gone through many developments with variations such as PPMA and PPMB (Cleary & Witten 1984), PPMC (Moffat 1990), PPMD (Howard 1993), PPM* (Cleary & Teahan 1997) and PPMO (Wu & Teahan 2008). For PPMC, the probability P_{PPMC} for the next character φ is given by:

$$P_{PPMC}(\varphi) = \frac{c_d(\varphi)}{T_d}$$

where the currently used coding order is specified by d, the total amount of times that the current context $c_{i-5} \cdots c_{i-1}$ has occurred is indicated by $T_d(c_{i-5} \cdots c_{i-1})$. $c_d(c_i|c_{i-5} \cdots c_{i-1})$ represents the total number of occurrences for the symbol c_i in the current context. The estimation of the escape probability E by PPMC is as follows:

$$E_{PPMC} = \frac{t_d}{T_d}$$

where the total number of times that a unique character has occurred following the current context is represented by t_d.

PPMD is a slight variant of PPMC invented by Howard (Howard 1993) which often results in better compression. The formula for estimating the probability P for the following character φ is given by:

$$P_{PPMD}(\varphi) = \frac{2c_d(\varphi)-1}{2T_d} \qquad (1)$$

and the estimation of the escape probability is as follows:

$$E_{PPMD} = \frac{t_d}{2T_d}. \qquad (2)$$

Table 2 below shows an example of how the PPMC processes the string “I have a dream. I have a dream. I ha” using different orders K= 2, 1, 0 and -1, where K means the prediction of the upcoming character will be estimated based on the (number of K) preceding characters. Usually, each character will be encoded arithmetically with the probability estimated by the model (Witten et al. 1987). Although for the purposes of classification, the arithmetic coding step can be eliminated since the physical process of writing to a file on disk is not required and only the modelling step is required.

PPM is used for classification by simply selecting the class related with the model that best compresses the text. The main idea is to predicate the correct dialect of text T using the formula:

$$\hat{\theta}(T) = argmin_c H(T|S_c)$$

where $H(T|S)$ is some approximation of relative entropy of text T with respect to text S and the class c is chosen from the model with the minimum value. In this case, it is estimated using the PPM compression scheme i.e. for an order five model, it is calculated using the following formula:

$$H(T|S) = -\sum_{i=1}^{n} \log_2 P(c_i|c_{i-5} \cdots c_{i-1})$$

where n is the length of the text and the probabilities for each character are calculated using the PPM Markov-based modelling method which estimates the probability of the next character (see formulas (1) and (2) for PPMD) based on the context of the previous five characters.

Order K=2				**Order K=1**			**Order K=0**			**Order K= -1**		
Prediction	c	p		Prediction	c	p	Prediction	c	p	Prediction	c	p
I□ →h	3	3/4		I →□	3	3/4	→I	3	$\frac{3}{46}$	→ A	1	$\frac{1}{\|A\|}$
→esc	1	1/4		→esc	1	1/4						
□h →a	3	3/4		□ →h	3	3/13	→□	9	$\frac{9}{46}$			
→esc	1	1/4										
□a →□	2	2/3		→a	2	2/13	→h	3	$\frac{3}{46}$			
→esc	1	1/3										
□d →r	2	2/3		→d	2	2/13	→a	7	$\frac{7}{46}$			
→esc	1	1/3										
□I →□	2	2/3		→I	2	2/13	→v	2	$\frac{2}{46}$			
→esc	1	1/3										
ha →v	2	2/3		→esc	4	4/13	→e	4	$\frac{4}{46}$			
→esc	1	1/3										
av →e	2	2/3		h →a	3	3/4	→d	2	$\frac{2}{46}$			
→esc	1	1/3		→esc	1	1/4						
a□ →d	2	2/3		a →v	2	2/9	→r	2	$\frac{2}{46}$			
→esc	1	1/3										
am →.	2	2/3		→□	2	2/9	→m	2	$\frac{2}{46}$			
→esc	1	1/3										
ve →□	2	2/3		→m	2	2/9	→.	2	$\frac{2}{46}$			
→esc	1	1/3		→esc	3	3/9						
e□ →a	2	2/3		v →e	2	2/3	→esc	10	$\frac{10}{46}$			
→esc	1	1/3		→esc	1	1/3						
ea →m	2	2/3		e →□	2	2/6						
→esc	1	1/3										
dr →e	2	2/3		→a	2	2/6						
→esc	1	1/3		→esc	2	2/6						
re →a	2	2/3		d →r	2	2/3						
→esc	1	1/3		→esc	1	1/3						
m. →□	2	2/3		r →e	2	2/3						
→esc	1	1/3		→esc	1	1/3						
.□ →I	2	2/3		m →.	2	2/3						
→esc	1	1/3		→esc	1	1/3						
				. →□	2	2/3						
				→esc	1	1/3						

TABLE 2: The generation of PPMC model after processing the string "*I have a dream. I have a dream. I ha*" using maximum order 2. (The space is represented by □ in this figure).

Imagine three scenarios where two subsequent letters – "ve", "te", and "rm" – are encountered after the sentence "I have a dream. I have a dream. I ha" has already been seen (see Table 3). First, for encoding "ve" following "ha", using maximum order of two, in this situation the probability is estimated as $\frac{2}{3}$ for each letter ('v' and 'e'), since the context and predictions, ha→v, and av→e

are found in order two (K=2) context (see Table 2). This requires 1.21 bits [$-\log_2\left(\frac{2}{3}\times\frac{2}{3}\right)$] to encode. (As stated, PPM normally uses arithmetic coding to physically encode the probabilities which results in the code length being close to the theoretical optimum which is $-\log_2 p$ where p is the probability being encoded. However, when using PPM for text classification purposes, there is no need to physically encode the probabilities and instead, PPM computes the theoretical code lengths directly and uses that as the categorisation measure.)

However, if "te" needs to be encoded following "ha", the escape probability of $\frac{1}{3}$ will be encoded from order two because the letter "t" was not seen in that context after following the "ha". Then the process will move down to order one, and the escape probability $\frac{3}{9}$ will need to be encoded again because the letter "t" was also not seen in order one after following "a". Next, the escape probability $\frac{10}{46}$ will be encoded a third time because the letter "t" was also not seen in order zero. Finally, the process will move down to order -1 where the letter "t" is found, so the encoded probability will be $\frac{1}{A}$ where A is the alphabet size (256 for a standard byte-based encoding 8 bits). Moreover, the second letter "e" will be encoded with probability $\frac{1}{3}$ after escaping because the context "at" was not seen in order two. After that, the escape probability $\frac{3}{9}$ will be encoded again because the letter "e" was also not seen in order one after following the "a". Next, the escape probability of $\frac{10}{46}$ will be encoded again because the letter "e" was also not seen in order zero. Finally, the process will move down to order -1 where the letter "e" is found, so the encoded probability will be $\frac{1}{A}$.

The total probability for encoding the letter "t" is ($\frac{1}{3}(esc)\times\frac{3}{9}(esc)\times\frac{10}{46}(esc)\times\frac{1}{A}$), and the letter "e" is ($\frac{1}{3}(esc)\times\frac{3}{9}(esc)\times\frac{10}{46}(esc)\times\frac{1}{A}$). which requires 26.9 bits to encode both letters (see Table 3).

Finally, If the aim is encoding the two letters "rm" after seeing "ha", then the escape probability will be $\frac{1}{3}$ for order two and $\frac{3}{9}$ for order one because the letter "r" was not seen in both orders, before the letter is found in order zero where the encoded probability will be found $\frac{2}{46}$. Similarly, the letter "m" will result in the encoding of an escape probability of $\frac{1}{3}$ for order two, $\frac{3}{9}$ for order one, until the letter is seen in order zero where the encoded probability will be found $\frac{2}{46}$. The total probability to encode the letter "r" is ($\frac{1}{3}(esc)\times\frac{3}{9}(esc)\times\frac{2}{46}$), and the letter "m" is ($\frac{1}{3}(esc)\times\frac{3}{9}(esc)\ \frac{2}{46}$) which requires 15.5bits to encode both characters (see Table 3).

Text	Subsequent letters	Codelength being used
..dream I ha..	ve	$-\log_2\left(\frac{2}{3}\times\frac{2}{3}\right)=1.21$ bits
	te	$-\log_2\left((\frac{1}{3}\times\frac{3}{9}\times\frac{10}{46}\times\frac{1}{\mathrm{A}})(\frac{1}{3}\times\frac{3}{9}\times\frac{10}{46}\times\frac{1}{\mathrm{A}})\right)=26.9$ bits
	rm	$-\log_2\left((\frac{1}{3}\times\frac{3}{9}\times\frac{2}{46})(\frac{1}{3}\times\frac{3}{9}\times\frac{2}{46})\right)=15.5$ bits

TABLE 3: Encoding Sample Characters using PPMC.

5. EXPERIMENTS RESULTS

Two main experiments were performed for this study. The first experiment involved single tweets dialect identification to determine whether the system was able to identify the dialect from a single tweet, knowing that a single tweet is written in 140 characters or less. This is considered challenging as the system is being provided with minimal dialectal context. The second

experiment involved combining test set tweets for each author in order to investigate dialect identification in relation to each author. Each author was labelled according to which dialect appeared the most frequently in the author's training file. The goal of this experiment was to identify the main dialect used by each author.

5.1 Dialect Identification of Single Tweets

In this experiment, a multiclass (7-way) classification task was investigated involving the identification of each single tweet's dialects (Modern, Classic, Gulf, Egyptian, Levantine, Maghrebi, Iraqi). This was performed using different orders of PPMD from order 2 to order 13. Each tweet is split into a single file; over 6573 tweets were tested in three different test sets. While other research experiments perform dialects classification using machine learning algorithms, this experiment is novel in its approach to dialect classification for Arabic text as it investigates the use of the character-based text compression scheme PPM. Table 4 shows that the best result that was obtained with order 7, achieving an accuracy of 74.1%.

Orders	Accuracy (%)	Recall	Pression	F-measure
Order 2	61.3	0.53	0.44	0.48
Order 3	66.2	0.60	0.50	0.54
Order 4	70.2	0.65	0.55	0.59
Order 5	72.2	**0.66**	0.57	0.61
Order 6	74.0	**0.66**	0.60	**0.63**
Order 7	**74.1**	0.64	0.60	0.62
Order 8	73.8	0.63	0.60	0.62
Order 9	73.8	0.63	0.60	0.61
Order 10	74.0	0.63	0.60	0.61
Order 11	74.0	0.63	0.60	0.61
Order 12	73.7	0.62	0.60	0.61

TABLE 4: Dialect identification of Arabic single tweets using PPMD.

Three machine learning classifiers were also investigated using Weka (Hall et al. 2009): Support Vector Machines specifically the LIBSVM package (Chang & Lin 2011); Multinomial Naïve Bayes (MNB); and k-nearest neighbours (KNN). In order to classify text for machine learning algorithms using Weka, training and testing sets need to be run through a string-to-word-vector filter. The filter for this experiment was built using the common term frequency-inverse document frequency (tf-idf) measure. Multiple tokenisers were also used such as word N-grams and character N-grams; however, no further pre-processing of the data was carried out, such as stemming, tokenisation, and removal of stop words, as the intent was to mimic the same approach used for PPM.

Classifiers		MNB				LibSVM				KNN 1			
Features		Acc. (%)	Rec.	Prec.	F-Meas.	Acc. (%)	Rec.	Prec.	F-Meas.	Acc. (%)	Rec.	Prec.	F-Meas.
word	Unigrams	**68.8**	**0.69**	**0.68**	**0.71**	67.3	0.64	0.67	0.67	53.4	0.53	0.53	0.53
	Bigrams	57.6	0.54	0.57	0.56	50.8	0.34	0.50	0.43	**56.4**	**0.52**	**0.56**	**0.54**
Character	Unigrams	56.2	0.54	0.56	0.55	58.5	0.54	0.58	0.55	48.5	0.49	0.48	0.50
	Bigrams	64.0	0.64	0.64	0.64	69.1	0.68	0.69	0.68	20.2	0.28	0.20	0.64
	Trigrams	68.3	0.68	0.68	0.70	**72.8**	**0.72**	**0.72**	**0.73**	20.2	0.22	0.20	0.60
	4-grams	67.9	0.68	0.67	0.70	71.8	0.71	0.71	0.72	26.2	0.30	0.26	0.56
	5-grams	66.4	0.66	0.66	0.68	68.2	0.66	0.68	0.68	37.2	0.41	0.37	0.56
	6-grams	64.2	0.64	0.64	0.64	63.2	0.58	0.63	0.65	53.3	0.52	0.53	0.52

TABLE 5: Dialect identification of Arabic single tweets using different features.

Table 5 shows that character trigrams feature identified single dialect tweets using LibSVM achieving the best result with an accuracy of 72.8%. MNB identified tweets best with the unigram feature, achieving an accuracy of 68.8%. However, the KNN1 classifier found word bigram performed best with an accuracy of 56.4%. Overall, in terms of F-measure, LibSVM was found to perform very well with results as high as 0.73.

Table 6 shows the confusion matrix for PPMD order 7 which achieved the highest score when compared with other the orders, as shown in Table 4. The confusion matrix shows a close relation between CA and MSA as both are considered formal. However, there are a few uses of words interchangeably which leads to some tweets being mis-classified. Also, the results show more mis-classifications of MSA tweets with other dialects; this shows that there is some overlap between MSA and other dialects which makes it difficult to classify.

The table also shows less confusion between the Egyptian, Levantine and Maghrebi dialects which may be due to the clear features and that make it easier for the classifier to distinguish between these dialects. On the other hand, more confusion for both Iraqi and Gulf dialects can be seen; this is due to the overlap between Gulf and Iraqi dialects as south of Iraq are influenced by the gulf dialects which also lead to confusion between the two dialects. Finally, it is important to note that the classifier used in this experiment has managed to distinguish between dialects despite the data being imbalanced.

PPMD 7	CA	Egyptian	Gulf	Levantine	MSA	Iraqi	Maghrebi
CA	1220	8	12	3	206	6	6
Egyptian	8	418	24	23	73	5	13
Gulf	21	36	363	59	91	22	21
Levantine	11	18	39	228	37	7	18
MSA	507	89	121	33	2503	24	51
Iraqi	3	7	13	16	16	25	4
Maghrebi	4	10	6	13	12	7	113

TABLE 6: Confusion matrix for single tweets dialect identification using PPMD order 7.

5.2 Author Dialect Identification

As well as classifying single tweets, classification of multiple tweets from the same author were also investigated. Each author was labelled according to the highest dialect found in the training set. This classification task is different to the problem of authorship attribution which was previously investigated (Altamimi & Teahan 2017). The intention is to classify text according to dialects used by the author, not to classify text according to which author the tweets belongs to.

Measures	Accuracy	Recall	Precision	F-measure
MNB	86.1	0.861	0.870	0.862
LibSVM	71.2	0.713	0.663	0.672
KNN 1	34.6	0.347	0.402	0.190
PPMD 5	**87.1**	**0.840**	**0.915**	**0.876**

TABLE 7: Results of author dialect identification using machine learning algorithms and PPMD.

Table 7 shows the results of classifying a total of 101 authors according to their dialect using MNB, LibSVM, KNN, and PPMD. The best result is achieved using PPMD order 5 reporting an accuracy of 87.1%, slightly better than MNB achieving an accuracy of 86.1%. LibSVM reported an accuracy of 71.2%. Finally, KNN 1 achieved an accuracy of 34.6%. However, various word and character features were also applied to the machine learning classifiers in Table 8. The

results show that MNB produces the best result using word unigram features, whereas LibSVM achieved the best result using character 4-grams features, with an accuracy of 78%. Finally, KNN 1 achieved best results using character unigram features achieving 49.5%.

Classifiers		**MNB**				**LibSVM**				**KNN 1**			
Features		Acc.	Rec.	Prec.	F-Meas.	Acc.	Rec.	Prec.	F-Meas.	Acc.	Rec.	Prec.	F-Meas.
Word	Unigrams	**86**	**0.86**	**0.87**	**0.86**	71	0.71	0.66	0.67	34.6	0.34	0.40	0.19
	Bigrams	80	0.80	0.81	0.79	36	0.36	0.40	0.22	40.0	0.40	0.59	0.29
Character	Unigrams	52	0.52	0.64	0.49	54	0.54	0.58	0.51	**49.5**	**0.49**	**0.57**	**0.48**
	Bigrams	68	0.68	0.73	0.68	60	0.60	0.63	0.55	4.09	0.05	0.28	0.02
	Trigrams	76	0.76	0.78	0.75	73	0.73	0.70	0.70	9.09	0.09	0.29	0.03
	4-grams	80	0.80	0.82	0.79	**78**	**0.78**	**0.74**	**0.75**	9.09	0.09	0.29	0.03
	5-grams	80	0.80	0.81	0.80	75	0.75	0.74	0.72	9.09	0.09	0.29	0.03
	6-grams	80	0.80	0.83	0.80	76	0.76	0.75	0.74	34.6	0.34	0.40	0.19

TABLE 8: Results for dialect identification of Arabic author using different features.

Table 9 shows the confusion matrix for the PPMD order 5 classifier which achieved the highest scores in all the experiments as shown in Table 7. The confusion matrix shows that there was some confusions between CA and MSA due to the overlap of some features they both use. This supports the earlier informal observation when classifying single tweets. Also, similar to previous experiments performed using single tweets, less confusion among Egyptian, Levantine, and Maghrebi dialects was observed. In contrast, a close relationship can be seen between the Iraqi and Gulf dialects as three Iraqi authors were classified as using the Gulf dialect, so this was examined further in the following section.

PPMD 5	**CA**	**Egyptian**	**Gulf**	**Levantine**	**MSA**	**Mix**	**Iraqi**	**Maghrebi**
CA	26	0	0	0	2	0	0	0
Egyptian	0	8	0	0	0	0	0	0
Gulf	0	0	7	0	0	0	1	0
Levantine	0	0	0	5	0	0	0	0
MSA	3	1	1	0	31	1	0	2
Mix	0	0	0	0	0	5	0	0
Iraqi	0	0	0	0	0	0	3	0
Maghrebi	0	0	0	0	1	0	0	3

TABLE 9: Confusion matrix for author dialect identification using PPMD order 5.

5.3 Limitation of Twitter Text Dialects Classification

We noted a few of the longer tweets were mis-classified. These tweets were then analysed in more depth to help better understand how difficult the task is. This showed that identifying dialects within tweets can be complicated even for a native speaker, for the following reasons:

- There are no defined boundaries between dialects and modern standard Arabic when dealing with text; for instance, some tweets are influenced by the standard modern Arabic regardless of the dialects used.
- The classification is affected by the topic bias of the tweets; for instance, the tweet might be classified by the tweet's topic regardless of the dialects being used.
- The cost of encoding the text can be dominated by people's names or location when classifying tweets.

- Classifying single tweets is more challenging due to the fact that some tweets contain less dialect content.

6. DISCUSSION AND FINDINGS

Most of the work on dialect identification performed binary classification to identify two dialects (Darwish et al. 2014; Elfardy & Diab 2013; Zaidan & Callison-Burch 2014). Other studies were performed with more dialects by using 4-way and 5-way classifications (Zaidan & Callison-Burch 2011; El Haj et al. 2017). However, according to Katakis et al. (2008), the more labels there are to categorise, the more complicated the identification task becomes. In contrast, the experimental results reported below investigated a 7-way classification task including five dialects in addition to Modern Standard Arabic and Classical Arabic.

There are two experimental settings used in most dialect studies: the first involves identifying short text represented by a single sentence or tweet. The second setting involves large text represented by paragraphs, multiple sentences, or multiple tweets. Most previous work on identifying Arabic dialects has involved classification of short text represented by sentences or tweets (Zaidan & Callison-Burch 2011; Elfardy & Diab 2013; Sadat et al. 2014; Darwish et al. 2014; Malmasi et al. 2015; El Haj et al. 2018).

Studies have performed experiments using different data sizes; for example, Darwish et al. (2014) performed the experiments using a total of 700 tweets. Malmasi et al. (2015) used sentences collected from the multidialectal parallel corpus of Arabic (MPCA). A total of 2000 sentences were translated by native speakers into five Arabic dialects. The studies by El Haj et al. (2018) and Sadat et al. (2014) used a total of 16,000 tweets and 63,000 sentences, respectively, to perform their experiments. Other researchers (Zaidan & Callison-Burch 2011; Elfardy & Diab 2013) used the AOC dataset, which is the closest dataset to the corpus of this current study, consisting of 108 thousand sentences collected from the commentary sections in popular Arabic newspapers.

However, the experiments in this paper investigated a 7-way classification task including five main Arabic dialects in addition to Modern Standard Arabic and Classical Arabic. In addition, two experimental settings were used in this study: identification of short text represented by single tweet; as well as large text represented by multiple tweets composed by the same author. Furthermore, this study used over 112 thousand tweets from BTAC as a training set, and also performed the testing on an unseen test set consisting of over 6500 tweets. Results for a number of Arabic dialect identification experiments have been presented using Prediction by Partial Matching (PPM) employing the character-based approach. These results have also been compared with those from other machine learning algorithms using character-based and word-based approaches.

The findings demonstrated the utility of the selected corpus BTAC for experiments for Arabic dialect identification. Single tweets identification achieved an accuracy of 74% and a F-measure of 0.630 for PPM. This result compares with other Arabic dialect identification studies performed on single tweets. For instance, the study by Zaidan and Callison-Burch (2011) used a dataset size of over 108 thousand sentences. They reported an accuracy of 69.4% performed on a 4-way classification task. Also, the research by Abu Kwaik et al. (2018) yielded an accuracy of 52% performed on a 4-way classification task.

Although other researchers have produced better results than those from this study, for this study we trained and tested on a larger dataset. El Haj et al. (2018) performed their study using 16 thousand tweets. Their study achieved an accuracy of 76.2% and an F-measure of 0.78. They performed a 5-way classification task using SVM. Moreover, Sadat et al. (2014) performed Arabic dialect identification on 18 Arabic variations. A total of 63 thousand sentences were used for training data, with the testing set consisting of 100 sentences for each dialect. They reported an overall F-measure of 0.80 and an accuracy of 98% using the character bi-gram model. Malmasi et al. (2015) obtained similar results to this current study achieving an accuracy of 74% from a

total dataset consisting of just two thousand sentences. The study by Alshutayri and Atwell (2017) reported 5-way classification accuracy of 79%. Their training data contained 8,090 tweets, and testing was done on 1,764 tweets.

In addition, this work has also investigated classification of author dialects yielding an accuracy of 87%. When comparing this with other machine learning algorithms, MNB performed the best with an accuracy of 86%. In addition, the inaccurately classified authors were highlighted, and it was found that the mis-classification was due to either fewer tweets by the author or that some authors changed their style of writing in the testing set. We also found out that the classification of the text can be dominated by people's names or location or the tweet's topic.

In general, it was also found that character N-grams identify Arabic dialects best, similar to the results reported by Darwish et al. (2014) and Sadat et al. (2014). Specifically, we found that longer sequence of characters such as orders 5, 6, and 7 capture the dialect features best in Arabic for PPM. This is in contrast to other experiments on multiple tweets classification which reported that word-based approaches identified Arabic dialects best (Zaidan & Callison-Burch 2014; Salama et al. 2014; Harrat et al. 2017; Alshutayri & Atwell 2017; Abu Kwaik et al. 2018). However, the above-cited studies used machine learning algorithms which are known for their ability to perform well with word-based approaches.

7. CONCLUSION AND FUTURE WORK

In this paper, Prediction by Partial Matching was employed to identify Arabic text from our Twitter dataset for both: single tweets and multiple (author) tweets. The research achieved accuracies of 74% and 87% on both experiments. The results were also compared with the benchmark machine learning algorithms. In addition, various features such as character-based and word-based approaches were applied. It was found that the character-based PPM classifier consistently outperformed the machine learning character-based and word-based classifiers.

There are a number of possible directions for future work in this project. The accuracy can be improved by increasing the size of the training data for both the Maghrebi and Iraqi dialects. In this regard, the relatively high classification accuracy of the compression-based approach is reassuring, given the restricted amount of training data available. Furthermore, the generalizability of the system needs to be investigated with a much greater number of authors in the author dialect identification experiment in order to determine how well the system scales up with real case scenarios.

8. REFERENCES

[1] Abu Kwaik, K. et al., 2018. Shami: A Corpus of Levantine Arabic Dialects. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

[2] Alkhazi, I.S. & Teahan, W.J., 2019. Compression-Based Parts-of-Speech Tagger for The Arabic. *International Journal of Computational Linguistics (IJCL)*, 10(1).

[3] Alshutayri, A.O.O. & Atwell, E., 2017. Exploring Twitter as a Source of an Arabic Dialect Corpus. *International Journal of Computational Linguistics (IJCL)*, 8(2), pp.37–44.

[4] Altamimi, M., Alruwaili, O. & Teahan, W.J., 2018. BTAC: A Twitter Corpus for Arabic Dialect Identification. In *of the 6th Conference on Computer-Mediated Communication (CMC) and Social Media Corpora (CMC-corpora 2018)*. p. 5.

[5] Altamimi, M. & Teahan, W.J., 2017. Gender And Authorship Categorisation Of Arabic Text From Twitter Using PPM.

[6] Bouamor, H., Habash, N. & Oflazer, K., 2014. A Multidialectal Parallel Corpus of Arabic. In *LREC*. pp. 1240–1245.

[7] Chang, C.-C. & Lin, C.-J., 2011. LIBSVM: a Library for Support Vector Machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3), p.27.

[8] Cleary, J. & Witten, I., 1984. Data compression using adaptive coding and partial string matching. *IEEE transactions on Communications*, 32(4), pp.396–402.

[9] Cleary, J.G. & Teahan, W.J., 1997. Unbounded length contexts for PPM. *The Computer Journal*, 40(2 and 3), pp.67–75.

[10] Darwish, K., Sajjad, H. & Mubarak, H., 2014. Verifiably effective arabic dialect identification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pp. 1465–1468.

[11] Elfardy, H. & Diab, M., 2013. Sentence level dialect identification in Arabic. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. pp. 456–461.

[12] El Haj, M., Rayson, P.E. & Aboelezz, M., 2018. Arabic Dialect Identification in the Context of Bivalency and Code-Switching. *Proceedings of the 11th International Conference on Language Resources and Evaluation, Miyazaki, Japan.. European Language Resources Association*.

[13] Hall, M. et al., 2009. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1), pp.10–18.

[14] Hamdi, A. et al., 2015. POS-tagging of tunisian dialect using standard arabic resources and tools. In *Workshop on Arabic Natural Language Processing*. pp. 59–68.

[15] Harrat, S., Meftouh, K. & Smaili, K., 2017. Creating Parallel Arabic Dialect Corpus: Pitfalls to Avoid. In *18th International Conference on Computational Linguistics and Intelligent Text Processing (CICLING)*.

[16] Howard, P.G., 1993. The Design and Analysis of E cient Lossless Data Compression Systems.

[17] Kumar, R. et al., 2018. Automatic Identification of Closely-related Indian Languages: Resources and Experiments. *arXiv preprint arXiv:1803.09405*.

[18] Ljubešić, N. & Kranjčić, D., 2015. Discriminating between closely related languages on twitter. *Informatica*, 39(1).

[19] Ljubesic, N., Mikelic, N. & Boras, D., 2007. Language indentification: How to distinguish similar languages? In *Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on*. IEEE, pp. 541–546.

[20] Lui, M. & Cook, P., 2013. Classifying English documents by national dialect. In *Proceedings of the Australasian Language Technology Association Workshop 2013 (ALTA 2013)*. pp. 5–15.

[21] Malmasi, S., Refaee, E. & Dras, M., 2015. Arabic dialect identification using a parallel multidialectal corpus. In *International Conference of the Pacific Association for Computational Linguistics*. Springer, pp. 35–53.

[22] Moffat, A., 1990. Implementing the PPM data compression scheme. *IEEE Transactions on communications*, 38(11), pp.1917–1921.

[23] Sadat, F., Kazemi, F. & Farzindar, A., 2014. Automatic identification of arabic dialects in social media. In *Proceedings of the first international workshop on Social media retrieval and*

analysis. ACM, pp. 35–40.

[24] Teahan, W.J. & Cleary, J.G., 1997. Models of English text. In *Proceedings DCC'97. Data Compression Conference*. IEEE, pp. 12–21.

[25] Tiedemann, J. & Ljubešić, N., 2012. Efficient discrimination between closely related languages. *Proceedings of COLING 2012*, pp.2619–2634.

[26] Witten, I.H., Neal, R.M. & Cleary, J.G., 1987. Arithmetic coding for data compression. *Communications of the ACM*, 30(6), pp.520–540.

[27] Wu, P. & Teahan, W.J., 2008. A new PPM variant for Chinese text compression. *Natural Language Engineering*, 14(03), pp.417–430.

[28] Xu, F., Wang, M. & Li, M., 2017. Sentence-level dialects identification in the Greater China region. *arXiv preprint arXiv:1701.01908*.

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

[30] Zaidan, O.F. & Callison-Burch, C., 2011. The arabic online commentary dataset: an annotated dataset of informal arabic with high dialectal content. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, pp. 37–41.

[31] Zampieri, M. & Gebre, B.G., 2012. Automatic identification of language varieties: The case of Portuguese. In *KONVENS2012-The 11th Conference on Natural Language Processing*. Österreichischen Gesellschaft für Artificial Intelligende (ÖGAI), pp. 233–237.

[32] Zampieri, M., Gebre, B.G. & Diwersy, S., 2013. N-gram Language Models and POS Distribution for the Identification of Spanish Varieties (Ngrammes et Traits Morphosyntaxiques pour la Identification de Variétés de l'Espagnol)[in French]. *Proceedings of TALN 2013 (Volume 2: Short Papers)*, 2, pp.580–587.