Arabic Dialect Identification, Topic Classification and Sentiment Analysis using BERT Fine-Tuning

Youness Hourri[1], El Mokhtar Hribach[2], and Hicham El Moubtahij (Superviser)[3]

[1,2,3]Faculty of Sciences Dhar El Mahraz, B.P. 1796 Fez, Atlas, Morocco.
[1]youness.hourri@usmba.ac.ma
[2]elmokhtar.hribach@usmba.ac.ma
[3]hicham.elmoubtahij@usmba.ac.ma

June 24, 2022

1 Introduction

Arabic is the official language of 22 countries, spoken by more than 400 million speakers.It is recognized as the 4th most used language of the Internet. Arabic is classified in three main varieties: 1) CA, a form of Arabic language used in literary texts and Quran. 2) MSA, used for writing as well as formal conversations. 3) AD, used in daily life communication, informal exchanges, etc.. AD are mostly divided into six main groups: (1) Algerian, (2) Lebanon, (3) Morocco, (4)Tunisian, (5) Egyptian and Others contains the remaining dialect.

Social media users tend to use informal language to express their opinions. Arabic informal language combines a variety of dialects differ from each other such that same words or expressions may have drastically different sentiments.Tracking the public reactions and impressions against different events is very important.

In this paper, we first explore DziriBERT as a pre-trained model.Then, we fine-tune its parameters on the sentiment classification, topic classification, and Arabic dialect identification tasks in order to achieve new state-of-the-art results.We aim to handle the most encountered Arabic dialects challenges such as ambiguity and complexity in the context of sentence also in the way of constructing its by user.For this purpose, we adopt the key technical innovation language modeling DziriBERT.

In order to evaluate the usability of the collected data set, several studies have been performed on it, using different machine learning algorithms such as SVM, Naive Bayes Classifier, etc.The obtained representation is able to consider a sentence and capture the sentiment, topic, and which dialect a tweet belongs to. The main contributions of this work are as follows:

- we added new examples on the topic's dataset, to balance it.
- we added new examples on the sentiment's dataset, to balance it.
- we added also to the dataset of dialect 7000 exemples labeled by arabic

The rest of this work is organized as follows. The different dialects of spoken Arabic language are briefly described in section 2. Section 3 describes the proposed method in detail. We present our experiments and results in Section 4. Section 5 outlines the conclusion and future work 0

Dialectical Variability

The speakers of the Arabic language use in their daily discourses various dialects that can be considered as alternatives of the Modern Standard Arabic (MSA). Tunisian, Algerian and Moroccan dialects share several phonological traits among themselves thanks to their common history. Their lexicon contains several pronunciations inherited from other languages like Berber, French, Turkish, Italian and Spanish.

Also, Syrian, Palestinian and Egyptian dialects share a lot of phonological features. In the following sub-sections, we briefly describe these dialects.

- Tunisian Dialect

  Similar to other Maghrebi dialects, Tunisian vocabulary is generally Arabic, with some Berber words. However, it is morphologically and phonologically different from the MSA. The Tunisian dialect is very agglutinative: Speakers use often very few words where just one expresses a whole sentence. It differs from the MSA especially in its negation form where the markers are always agglutinated to other words as affixes or suffixes. Moreover, in the Tunisian dialect, several Arabic words are used with significant changes in their stem formation.

- Algerian Dialect
  The Algerian dialect is an informal spoken language, not used in official speech. Its vocabulary is roughly similar throughout Algeria. Nevertheless, in the east of the country, the dialect is closer to the Tunisian one whereas in the west it is closer to the Moroccan one. Most of the words of Algerian dialect come from the MSA [2], but there is a significant variation in vocalization in most cases, and some omission of some sounds in other cases. Contrary to the MSA, few sounds are not used in Algerian discourses like and , where most of the time they are respectively pronounced as and . Furthermore, the Algerian dialect uses some non-Arabic sounds like and .

- Moroccan Dialect
  The Moroccan dialect or the Moroccan Darija is a member of the Maghrebi Arabic language continuum spoken in Morocco.
  It is mutually intelligible to some extent with the Algerian dialect and to a lesser extent with Tunisian one. It has been significantly influenced by other vocabulary like Berber, Latin, French and Spanish.

- Egyptian Dialect
  The Egyptian Arabic is a North African dialect of the Arabic language, which is a branch of the Afro-Asiatic language. It originated in the Nile in Lower Egypt around the capital Cairo. Egyptian Arabic evolved from the Quranic Arabic, which was brought to Egypt during the seventh-century Muslim conquest that aimed to spread the Islamic faith among the Egyptians. The Egyptian dialect was very highly influenced by the Coptic language, which was the native language of the Egyptians prior to the Arab conquest [3], and later it was significantly influenced by other languages such as French, Italian, Turkish and English.

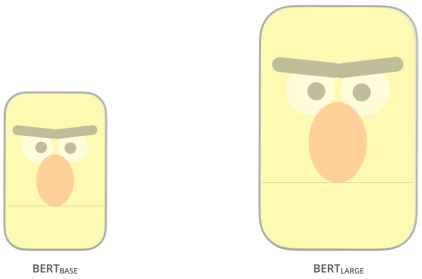
- Lebanon
  Lebanese Arabic shares many features with other modern varieties of Arabic. Lebanese Arabic, like many other spoken Levantine Arabic varieties, has a syllable structure very different from that of Modern Standard Arabic. While Standard Arabic can have only one consonant at the beginning of a syllable, after which a vowel must follow, Lebanese Arabic commonly has two consonants in the onset.

2 BERT language model

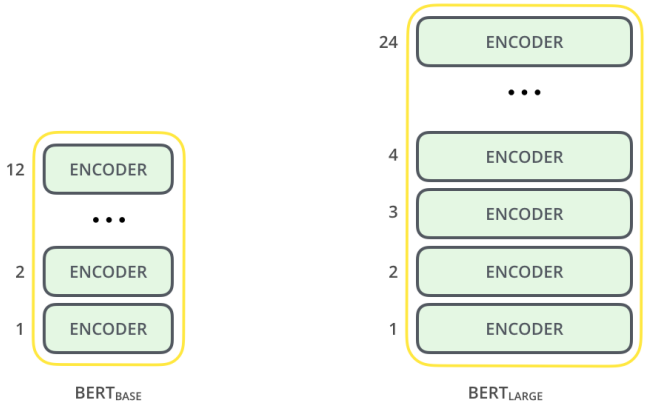
BERT is an open source machine learning framework for natural language processing (NLP). BERT is designed to help computers understand the meaning of ambiguous language in text by using surrounding text to establish context. The BERT framework was pre-trained using text from Wikipedia and can be fine-tuned with question and answer datasets.

BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection. (In NLP, this process is called attention.)

2.1 Model Architecture

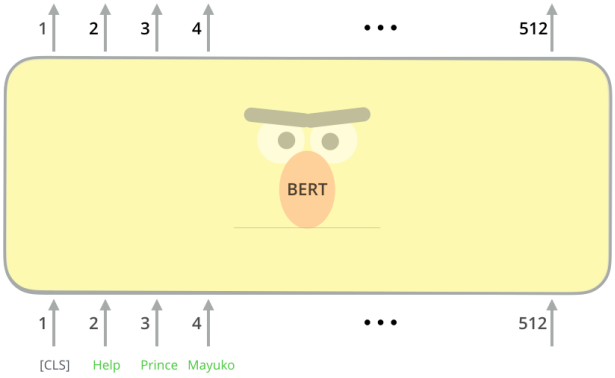


- BERT BASE – Comparable in size to the OpenAI Transformer in order to compare performance
- BERT LARGE – A ridiculously huge model which achieved the state of the art results reported in the paper[JD]



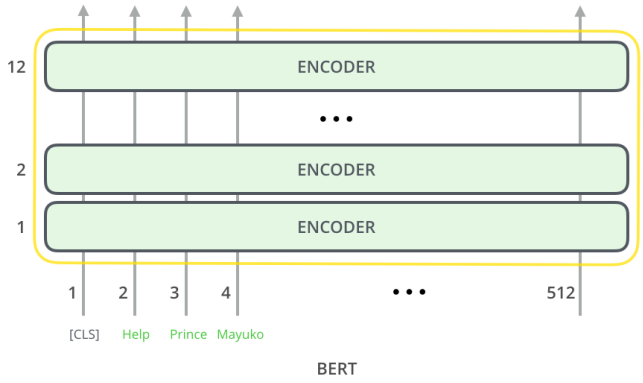
Both BERT model sizes have a large number of encoder layers (which the paper calls Transformer Blocks) – twelve for the Base version, and twenty four for the Large version. These also have larger feedforward-networks (768 and 1024 hidden units respectively), and more attention heads (12 and 16 respectively) than the default configuration in the reference implementation of the Transformer in the initial paper (6 encoder layers, 512 hidden units, and 8 attention heads).

2.2 Model inputs



The first input token is supplied with a special [CLS] token for reasons that will become apparent later on. CLS here stands for Classification.

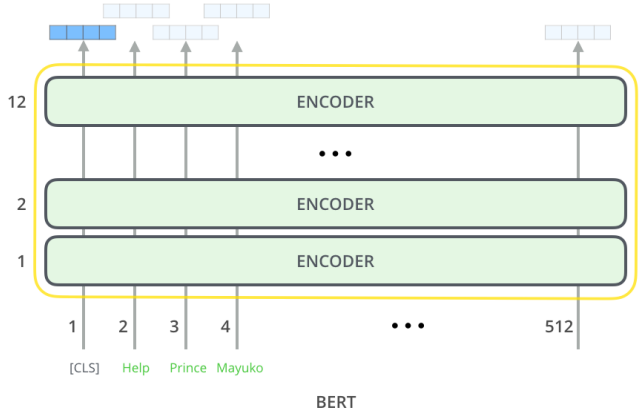
Just like the vanilla encoder of the transformer, BERT takes a sequence of words as input which keep flowing up the stack. Each layer applies self-attention, and passes its results through a feed-forward network, and then hands it off to the next encoder.



In terms of architecture, this has been identical to the Transformer up until this point (aside from size, which are just configurations we can set). It is at the output that we first start seeing how things diverge.

2.3 Model Outputs

Each position outputs a vector of size hidden-size (768 in BERT Base). For the sentence classification example we've looked at above, we focus on the output of only the first position (that we passed the special [CLS] token to).



2.4 Conclusion

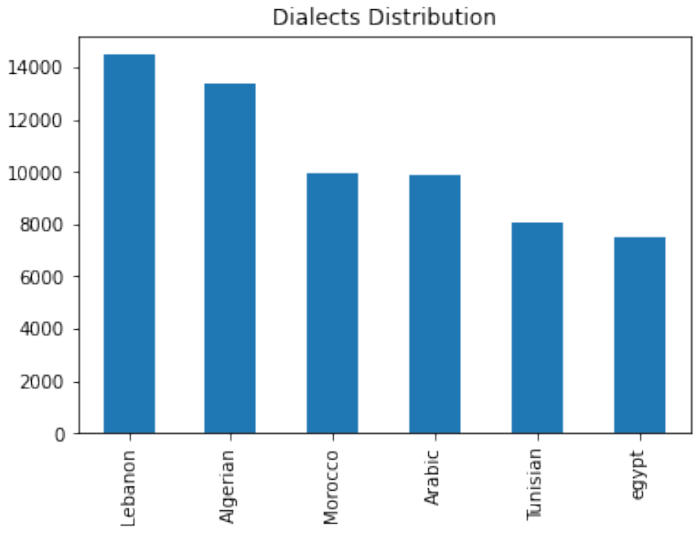
BERT is undoubtedly a breakthrough in the use of Machine Learning for Natural Language Processing. The fact that it's approachable and allows fast fine-tuning will likely allow a wide range of practical applications in the future.

3 Data Collection

The data set was created for this work from many sources, A simple pre-processing function was used to remove links, punctuation and stopwords. In this project, we present three main datasets:

3.1 Dialect Identification

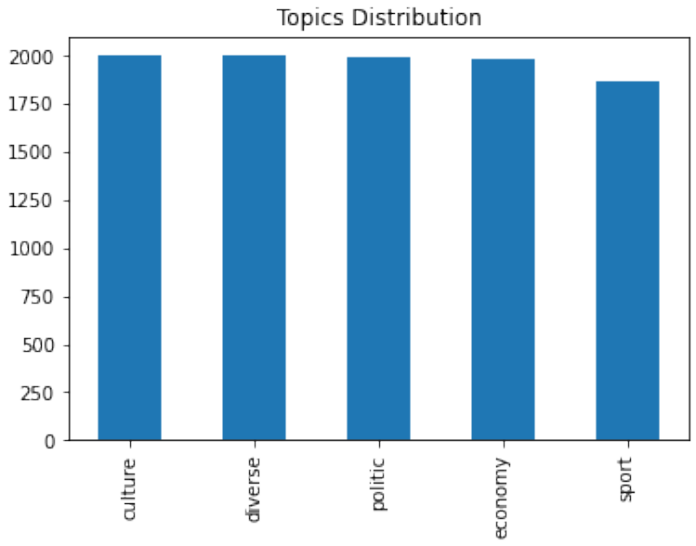
the total number of tweets is 63257 with 6 different labels (MSA, Moroccan, Algerian, Tunisian, Egyptian and Lebanese), it's a result of two data set, the first it was built by Mohamed VI Polytechnic University from this article [EB] and the second from newspapers articles which contains only MSA language.



We used pandas library to merge the datasets.

3.2 Topic Classification

The dataset is a collection of Arabic texts, which covers modern Arabic language used in newspapers articles. The text contains alphabetic, numeric and symbolic words. The existence of numeric and symbolic words in this dataset could tell the efficiency and robustness of many Arabic text classification and indexing documents. The dataset consists of 9854 sentences and 394,160 words structured in a csv file, and collected from 3 Arabic online newspapers: Assabah, Hespress and Akhbarona. The documents in the dataset are categorized into 5 classes: Sport, Politic, Culture, Economy and Diverse.



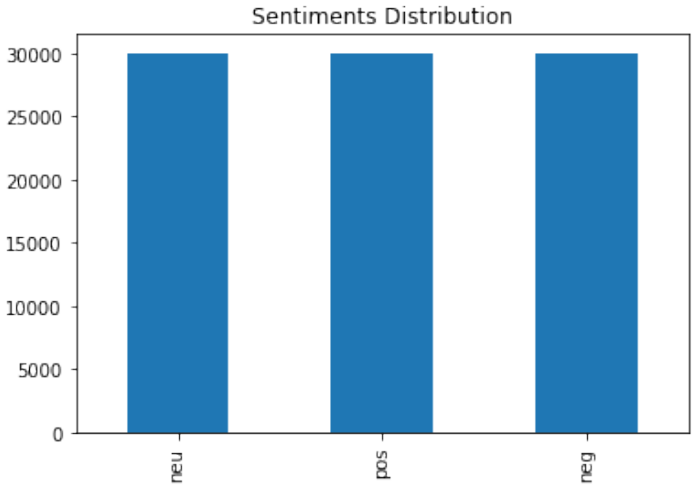
Despite the dataset contains only Arabic it working also with other dialects because the terms are same in the each single dialect, this dataset collected by Mohamed Biniz in this article [BIN]

3.3 Sentiment Analysis

The sentiment analysis dataset which collected by UM6P at first was contains 52210 labeled tweets (Positive, Negative and Neutral). Looking into the data set distribution in table below, we can notice that the data is totally imbalanced. For example, Neutral's tweets represent 57.52% of the data but Positive represents 13.01% of the data set. What makes the task even more challenging is the similarity between classes.

Label	Neutral	Negative	Positive
Num of Tweets	30033	15385	6792

This large variance makes our model weaker, to solve this we decided to add more tweets with positive and negative labels to balance our dataset, by using the Arabic sentiment corpus of Motaz Saad [Saa] with 58K of Arabic tweets annotated in positive and negative labels. By merging the two datasets we get a balanced dataset with 90k tweets, we used R programming language to transforme files to Dataframe afterwards we merge it with the old dataset



4 System Description

In this section, we present the various steps employed in our experiments, starting with model fine-tuning, followed by embedding the model to web application.

4.1 Fine-tuning

DziriBERT is a pre-trained model, it tested on one million Algerian tweets, it was introduced in this paper [AA] and is available on the Hugging Face Model Hub in this link , which means it is fairly easy to use and finetune the model.

In this report we will show how to finetune DziriBERT for sentiment analysis, it's the same thing for other models (dialect identification and topic classification) just changing dataset and some parameters.

In order to measure the performance of our model, we will compare it with other pre-trained models (DarijaBERT, MARBERT) and traditional machine learning algorithms (logistic regression, naive bayes)

we used several libraries, namely:

Transformers: provides APIs to easily download and train state-of-the-art pretrained models. Using pretrained models can reduce your compute costs, carbon footprint, and save you time from training a model from scratch.

Pandas: a fast and efficient DataFrame object for data manipulation with integrated indexing, provide tools for reading and writing data between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format

Sklearn: features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy

Torch: contains data structures for multi-dimensional tensors and defines mathematical operations over these tensors. Additionally, it provides many utilities for efficient serializing of Tensors and arbitrary types, and other useful utilities

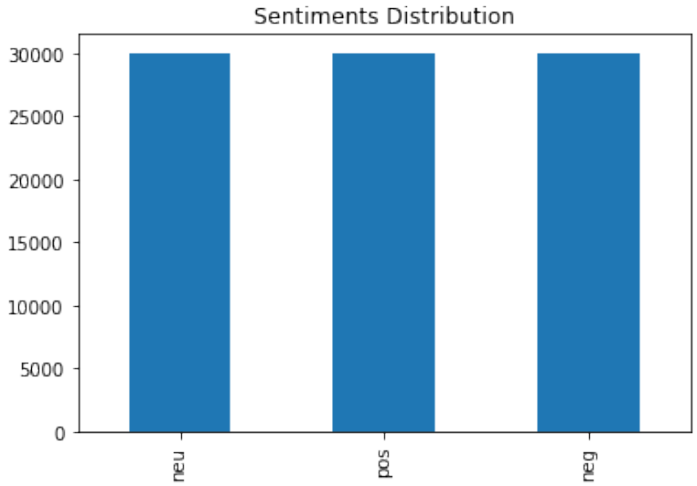
4.1.1 Data Importing

All datasets are available as a csv file

```
data = pandas.read_csv('Sentiment_Anaysis.csv')
```

This is the number of tweets for each class in our dataset

```
Data.label.value_counts().plot.bar(x=data.label.unique(), title='Tweet Distribution')
```



We encode the labels by mapping each dialect to an index (for exemple, assign 0 to Neutral, 1 to Positive and 2 to Negative)

```
sentiments = data['label'].unique()
lbl2idx = {d: i for i, d in enumerate(sentiments)}
data['label'] = data['label'].map(lbl2idx)
```

Using train_test_split() function from sklearn, we split the dataset randomly, 70% for training and 30% for testing

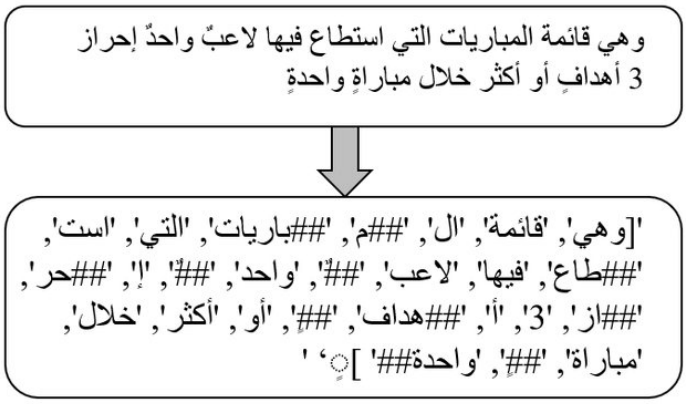
```
X_train, X_test, Y_train, Y_test = train_test_split(    data['Twits'],
                                                        data['label'],
                                                        test_size=0.3)
```

4.1.2 Tokenization

Tokenization is a way of separating a piece of text into smaller units called tokens. Here, tokens can be either words, characters, or subwords. Hence, tokenization can be broadly classified into 3 types – word, character, and subword (n-gram characters) tokenization.

We can use easly AutoTokenizer from transformers library, by specifying the name off model deployed in HuggingFace platform

```
from transformers import AutoTokenizer
tokenizer = AutoTokenizer.from_pretrained('alger-ia/dziribert')
```



We can see above that some words are split into pieces while some others are kept as they are. Some tokens are preceded by "##", this means that that is a completion of a word and should be attached to the previous token when decoding.

4.1.3 Encoding

Text encoding is a process to convert significant text into number/vector representation so as to preserve the context and relationship between words and sentences, such that a machine can understand the pattern associated in any text and can make out the context of sentences.

Before put in our data to the model, we need to convert them to vectors of numbers. What we see bellow is the index of each token in the vocabulary of the pretrained DziriBERT using WordPiece tokenizer

```
tokens = tokenizer.tokenize(arabic-sentence)
tokenizer.convert_tokens_to_ids(tokens)

output = [45796, 40053, 48544, 1821, 20759, 41048, 3985]
```

By visualising our data in a plot we can notice that average of sentences is 40 terms, so we can fix max_length parameter that represents the maximum sequence during encoding (maximum sequence length that BERT can take is 512)

```
train_encodings = tokenizer(    X_train.to_list(),
                                truncation=True,
                                padding=True,
                                max_length = 40)
```

The same about test and validation datasets

4.1.4 Model Training

Now is time to load the pretrained DziriBERT model from Hugging Face Hub
What AutoModelForSequenceClassification does is to remove the pretraining head of the model, and replace it with a classification head that will be initialized randomly.

```
from transformers import AutoModelForSequenceClassification
model = AutoModelForSequenceClassification.from_pretrained(
      model_name,
      num_labels=3)

# Define TrainingArguments
args = TrainingArguments(
    output_dir="output",
    evaluation_strategy="steps",
    eval_steps=500,
    per_device_train_batch_size=8,
```

```
    per_device_eval_batch_size=8,
    num_train_epochs=3,
    seed=0,
    load_best_model_at_end=True,)

# Define Trainer
trainer = Trainer(
    model=model,
    args=args,
    train_dataset=train_dataset,
    eval_dataset=val_dataset,)
```

4.1.5 Saving Model

We need to save the trained models and tokenizer in a file and restore them in order to reuse them to compare the model with other models, and to test the model on new data.

```
trainer.save_model('sentiment_model')
tokenizer.save_pretrained('sentiment_tokenizer')
```

4.2 Integrate the Models into Web Application

4.2.1 Model Loading

To use our trained model, we need to load both model and tokenizer:

```
SentimentTokenizer = AutoTokenizer.from_pretrained("sentiment_tokenizer")
SentimentModel = AutoModelForSequenceClassification.from_pretrained("sentiment_model")
```

For a given tweet. It's recommended to clean it, by remove non significant emojis, stop words remove long spacing and also non-Arabic characters (see source code for more details)

```
tweet = remove_emoji(tweet)
tweet = stopWords(tweet)
tweet = stemming(tweet)
tweet = cleanPunc(tweet)
tweet = clean_NonArabs(tweet)
clean_tweet = remove_spaces(tweet)
```

Then we can analyse the tweet by encode it using encode_plus() method, then extract prediction using torch library, also using a switcher to assign each index to its original label

```
def predict_sentiment(tweet):
    switcher = {
        "LABEL_0": "Neutral",
        "LABEL_1": "Positive",
        "LABEL_2": "Negative",
    }

    out = SentimentTokenizer.encode_plus(tweet, return_tensors='pt')
    SentimentModel.eval()

    with torch.no_grad():
      outputs = SentimentModel(**out)
      predictions = torch.nn.functional.softmax(outputs.logits, dim=1)
      labels = torch.argmax(predictions, dim=1)
      labels = [SentimentModel  .config
                               .id2label[label_id] for label_id in labels.tolist()]
      prediction = labels.pop(0)

    return switcher.get(prediction, "Neutral")
```

The same thing about dialect and topic classification (see source code for more details)

4.2.2 App Demonstration

5 Experimental Results and Discussion

Using our labeled data, we evaluate the performance of Arabic dialect identification, Arabic sentiment classification, and Arabic topic categorization systems with some standard and dialectal Arabic models (MarBERT, DarijaBERT).

for the three datasets, we allocated 70% for training,15% for validation and 15% for testing using a random stratified split.

the following tables present the obtained results on the sentiment and emotion classification tasks respectively. We calculated the accuracy and the macro averaged precision, recall and F1 score for each model on each dataset.

5.1 Dialect detection

Accuracy and macro averaged Precision, Recall and F1 score obtained by each model on the Dialect dataset..

Model	Acc.	F1	Pre.	Rec.
SGD Classifier	0.72	0.73	0.75	0.72
Logistic Regression	0.72	0.72	0.72	0.71
Naive bayes	0.75	0.75	0.80	0.71
Linear SVC	0.76	0.75	0.76	0.74
DarijaBERT	0.83	0.82	0.81	0.83
MARBERT	0.81	0.84	0.83	0.84
DziriBERT(old datasets)	0.85	0.83	0.81	0.85
DziriBERT(new datasets)	0.87	0.85	0.86	0.87

5.2 Topic detection

Accuracy and macro averaged Precision, Recall and F1 score obtained by each model on the topic dataset..

Model	Acc.	F1	Pre.	Rec.
Logistic Regression	0.82	0.82	0.59	0.59
SGD Classifier	0.84	0.72	0.65	0.45
Linear SVC	0.84	0.84	0.65	0.45
DarijaBERT	0.87	0.84	0.86	0.89
MARBERT	0.87	0.86	0.83	0.81
DziriBERT(old datasets)	0.87	0.88	0.83	0.88
DziriBERT(new datasets)	0.94	0.93	0.92	0.96

5.3 Sentiment analysis

Accuracy and macro averaged Precision, Recall and F1 score obtained by each model on the sentiment dataset..

Model	Acc.	F1	Pre.	Rec.
SGD Classifier	0.77	0.74	0.75	0.76
Logistic Regression	0.76	0.73	0.75	0.74
Naive bayes	0.73	0.67	0.75	0.64
Linear SVC	0.76	0.73	0.75	0.75
DarijaBERT	0.86	0.83	0.81	0.87
MARBERT	0.80	0.79	0.81	0.79
DziriBERT(old datasets)	0.86	0.86	0.81	0.85
DziriBERT(new datasets)	0.90	0.89	0.920	0.89

As shown in the tables above , DziriBERT outperforms all current multilingual, It is closely followed by MARBERT.

Our experiments confirm that pre-training a dedicated model for a given dialect on a small training set may give better results than pre-training a multi-dialectal model on much more data. In fact, MARBERT has been trained on 128 GB of text (almost x1000 times larger than our pre-training corpus). However, it has been trained all Arabic dialects which can be separated into +20 different groups, that vary from one region / country to another (Darwish et al., 2020). The same sentence may have different meanings according to the dialect spoken in each region. Therefore, we believe that the same sentence should have different representations according to the considered dialect.

6 Conclusion and Future Works

in this work we presented our efforts to construct a model for dialect identification sentiments analysis and topic classification.Sentiment analysis help businesses monitor brand and product sentiment in customer feedback, and understand customer needs.Also topic classification will help businesses make better decisions, optimize internal processes, identify trends, and provide all sorts of other advantages to make them more efficient and productive.We proposed to use the pre-trained DziriBERT in two different ways, transfer learning model and feature extractor. The first variant relied on the pre-trained DziriaBERT fine-tuning whereas the second one investigated two Arabic BERT models as feature extractors combined with several classifiers.The obtained results proved that the fine-tuned DziriBERT mode l outperformed mBERT, achieved new state-of-the-art results, and reached up to 87% in terms of accuracy on MSDA datasets. We intend to use DziriBERT for other NLP tasks.Besides, we will working on pretraining a new multiAraBERT model to understand different Arabic dialects not just this 5 dialects.

References

[AA] Mourad Oussalah Abdelouahab Moussaoui Amine Abdaoui, Mohamed Berrimi. Dziribert: a pre-trained language model for the algerian dialect.

[BIN] Mohamed BINIZ. Dataset for arabic classification.

[EB] Hamza Chataoui ElMehdi Boujou. An open access nlp dataset for arabic dialects : Data collection, labeling, and model construction.

[JD] Kenton Lee Kristina Toutanova Jacob Devlin, Ming-Wei Chang. Bert: Pre-training of deep bidirectional transformers for language understanding.

[Saa] Motaz Saad. Sentiment analysis in arabic tweets.