Saudi Authority for Data and
Artificial Intelligence
Data Science Bootcamp

Arabic tweets sentiment analysis (MVP)

Done by: Sarah Hamad Alhussiny

1. Introdaction:

In the world of artificial intelligence, which is getting bigger day by day and knowledge of it increases uninterruptedly, Natural Language Processing (Which is part of artificial intelligence and intersects with machine learning and deep learning, as shown in Figure 1)[1] become an important technology and have many applications that are useful in many sides even in our daily life, such as texts translating, speech to text, etc.

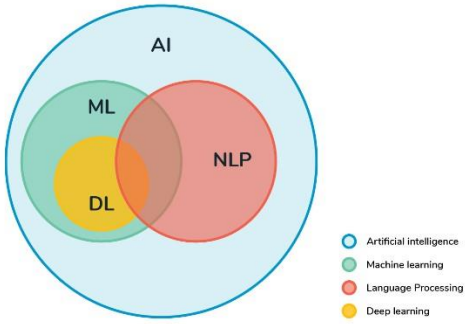

FIGURE 1: AI AND NLP

One of its most important tasks is text classification, which is the process of classifying texts or documents based on the content. Text classification has a variety of applications, such as classification of articles, classification of news, etc.[2].

FIGURE 2: TEXTS CLASSIFICATION

2. Proplome:

Often, it is difficult for a person to determine the sentiment of others, whether through their writings or voices. How if these statements are contained in thousands and millions of tweets. This makes it difficult for decision-makers to define them well and quickly. This sometimes leads to the anger of customers and users because of not paying attention to their opinions.

3. Aim:

The project aims to classify the tweets into negative or positive. So that, tweets sentiment analysis helps many people to save time and money as well as develop their businesses. For example, a company can find out what people think of their services by analyzing the tweets in their hashtag, A series director can know the audience's opinions about his series It is also possible to know the opinion of the people on the issue of public opinion, like, muting the sounds of mosque loudspeakers.

4. Dataset:

To solve this problem, I need the dataset that helps train the machine and build models. I searched for a lot of data and preferred it to be in Arabic to develop my skills in it. I found suitable data on the Kaggle website [3] (dataset link: https://www.kaggle.com/mksaad/arabic-sentiment-twittercorpus?select=test_Arabic_tweets_positive_20190413.tsv).

It initially consists of two features, a tweet, and the classification (positive or negative). I needed more features to build well-produced models, so I added many more features which is:

1- Tweet

3- Num of hashtags

4- word density

5- Sentence density

6- char_count

7- Word_count

8- Num of positive words

9- Num of negative words

9 features (columns) and 45275 tweets (rows).

5. Cleaning and Preprocessing

5.1. Text Cleaning

Data cleaning is the first step to prepare data for natural language processing. can help to reduce the noise present in text data so the machine can more easily detect the patterns in the data. Text data contains a lot of noise, this takes the form of special characters such as hashtags, punctuation, and numbers. which makes it difficult for the machine to understand.

The first step I make in cleaning data was removing extra spaces and repeated characters to remove the noise in the data, the data I use didn't need a lot of cleaning;. After that process, it is rare to find duplicate letters, spam text, and punctuation marks or noise in general, so I jump to the pre-processing.

5.2. Pre-Processing

In-Text preprocessing I start by identifying Arabic and English punctuations which are a set of symbols [!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~], also the stopwords, and number to remove them from the data to reduce the noise, also since I are working on Arabic text classification, I need to remove Arabic diacritics (Tashkeel) from the data since I will be focusing on the words and trying to find how much it's relevant to the category, so we are trying to normalize and clear the words to make it easier for the machine to understand the text, it also helps to increase the accuracy of the text classification.

5.3. Stemming

Stemming is the process of removing a part of a word or reducing a word to its stem or root (lemma), which is not always the dictionary root of the word. It helps in recognizing, searching, and retrieving more forms of words returns more results. When a form of a word is recognized, it can be used to determine domain vocabularies in domain analysis or text classification.

So, it will make it possible to put them in the same category that otherwise might be miss categorized.

5.4. CAMeL Tools

I use CAMeL Tools for pre-processing, which is a collection of open-source tools for Arabic natural language processing in Python developed by the CAMeL Lab at New York University Abu Dhabi. CAMeL

Tools currently provide utilities for pre-processing, morphological modeling, dialect identification, named entity recognition, and sentiment analysis.[3]

I use CAMeL functions such as:

1- Didacticization: which removes Arabic diacritics which occur infrequently in Arabic text and tend to be considered noise. These include short vowels, shadda (gemination marker), and the dagger alif

2- Orthographic Normalization: Due to Arabic's complex morphology, it is necessary to normalize text in various ways to reduce noise and sparsity (Habash, 2010). one of the most common of these normalizations is letter variant normalization.

3- Tokenization: is a way of separating a piece of text into smaller units called tokens. Here, tokens can be either words, characters, or subwords. where the main component is a lookup model of a word and its

most probable full morphological analysis. For a word that does appear in the lookup model, I use the morphological analyzer from Section 6.1 to generate all possible analyses or the word and chose the top-ranked analysis based on the pre-computed probabilistic score for the joint lemma and POS frequency.

6. Features Engineering:

- **Length**

  - Char count: number of characters for each word
  - Word count: number of words for each Sentence
  - Word density: number of characters/word count
  - Sentence density: number of words/numbers of sentence

- **PosWords & NegWords:**

PosWords = Num Of positive word in each Tweet.

NegWords = Num Of negative word in each Tweet.

```
positiveList=['الحمد','حل','العافيه','مبروك','اجمل','بالتوفيق','تفاءل','النصر',
'التعاون','صح','جميل','افضل','شكرا','سعاده','ايجابي','الخير','النجاح','اكثر','ر
اءع','متاز']
NegativeList=['اقل','سيء ه','عدم','المشكله','اعتداء','الهم','ضد','اهمال','جراح',
'خاب','غبي','محزن','سلبي','مءلم','فاشل','سيء','لاسف','خطا']
The lists from previses project worked in tweets sentiments.
```

After I build some models with these features the results were not good as shown in (figure:9) so I will not use it at all instead of this I use N-gram, AraVec, and TF-IDF to building the models.[5][6]

```
Accuracy :  66.70347874102706
-----------------------
Precision (macro avg) :  66.71105046236536
Recall (macro avg)    :  66.70808288538672
F-score (macro avg)   :  66.7028289848603
-----------------------
Precision (micro avg) :  66.70347874102706
Recall (micro avg)    :  66.70347874102706
F-score (micro avg)   :  66.70347874102706
-----------------------
Precision (weighted avg) :  66.71399823174134
Recall (weighted avg)    :  66.70347874102706
F-score (weighted avg)   :  66.7020005457477
```

FIGURE 1: NEW FEATURES RESULTS

- ## TF IDF

Short for term frequency (TF) –inverse document frequency (IDF) which is an information retrieval technique that weighs a word's frequency, and its inverse document frequency.

so, I generally compute a weight to each word which shows how important is the word in the text.

The value of it increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general.[7]

TF-IDF method is a widely used technique in Information Retrieval and Text Mining, however, it is good for text classification and to help the machine read words in numbers. but the problem with it is that TF-IDF doesn't work if you want to understand the meaning of the text (Tweets) or the document.

- ## N-Gram

N-Gram is a concept found in Natural Language Processing. It is used for a variety of things. You can think of an N-gram as the sequence of N words, by that notion, a 2-gram (or bigram) is a two-word sequence of words like "Sara Amal" and 3-gram (or trigram) is a three-word sequence of words like "student information technology". n-gram Can be more maybe 6 or 7. Advantages of using N-grams: Simple, easy, and cheap.

- ## AraVec

The data used for our project is in Arabic, so I used the AraVec for word embedding as shown in Figure 6. It is a pre-trained project by using many word embedding's enables users to embed their data by using it for free because it is open source. The first version of it was using the content of three Arabic content domains are: Tweets, World Wide Web pages, and Wikipedia.[8]

I used World Wide Web pages because it's more similar to our data and it has the largest number of tokens (Vocabularies = 234,961) and I integrate it with Spacy.io.

```
[31] AraVec_word_embedding = np.array([nlp(d).vector for d in balanced_data['Text'].values.astype('U')])

     # print the dimension of x
     print(AraVec_word_embedding.shape)

     (16261, 100)
```

FIGURE 2: APPLING ARAVEC TO OUR DATA

7. Modeling:

I use three models **KNN, SVM, and Logistic Regression then I enter** TF-IDF, AraVec, and N-gram to **each model to compare the best result between them.**

- **KNN:**

KNN is a simple algorithm, easy-to-implement supervised machine learning that can be used to solve both classification and regression problems, and a very popular algorithm for text classification. I used it with (TF-IDF, Aravec, and N-gram) to convert the words into a vector and then make all the nearby vectors into a group where the k was 40, p=2 which mean Euclidean distance: computes the root of square difference between co-ordinates. Of pair of objects. [9]

- **SVM:**

A Support Vector Machines (SVM) is a Supervised Machine Learning Algorithm With an associated learning algorithm that analyzes data for classification and regression. it is mostly used in classification problems. - SVM is a common learning algorithm for tasks like POS (Part of-speech) tagging for Natural Language Processing (NLP).

- **Logistic Regression:**

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. It is considered one of the simplest ML algorithms that can be used for various classification problems.

```
[ ] # function to train and test model: take 5 parameters model name, training and testing data. it return the evaluation report
    def building_model(classifier, X_train, y_train, X_test, y_test):

        # fit the training dataset on the classifier
        classifier.fit(X_train, y_train)

        # predict the labels on validation dataset
        prediction = dict()
        prediction["lr"] = classifier.predict(X_test)

        # report
        report= classification_report(y_test,prediction["lr"], output_dict=True)
        report2 = classification_report(y_test, prediction["lr"], labels=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,

        Precision_macro= report['macro avg']['precision'] * 100
        Recall_macro= report['macro avg']['recall'] * 100
        f1_macro= report['macro avg']['f1-score'] * 100
        Accuracy= report['accuracy'] * 100

        Precision_micro= report2['micro avg']['precision'] * 100
        Recall_micro= report2['micro avg']['recall'] * 100
        f1_micro= report2['micro avg']['f1-score']  * 100

        Precision_weighted = report['weighted avg']['precision'] * 100
        Recall_weighted = report['weighted avg']['recall'] * 100
        f1_weighted = report['weighted avg']['f1-score'] * 100


        return [Accuracy, Precision_macro,Recall_macro, f1_macro, Precision_micro, Recall_micro,f1_micro,Precision_weighted,Recall_weighted, f1_wei
```

I wrote this method to make it easier to build and evaluate models, as it receives data after splitting. And it returns the results in the order I chose.

8. Evaluation Results:

In the Evaluation Results I use SVM, logistic regression, and KNN then I enter it in Aravec, N-gram, and Tf-IDF to see the butter result.

Models	Based on Independent (input)	AraVec	N-gram	Tf-IDF
KNN	Accuracy	69.91	63.90	57.84
	Precision	70.00	72.01	74.25
	Recall	69.91	63.90	57.84
	F-score	69.89	60.36	49.45
Logistic Regression	Accuracy	67.23	64.66	77.30
	Precision	67.28	73.34	77.50
	Recall	67.23	64.66	77.30
	F-score	67.21	61.14	77.27
SVM	Accuracy	66.68	64.71	77.85
	Precision	66.84	73.58	77.91
	Recall	66.68	64.71	77.85
	F-score	66.61	61.16	77.85

TABLE 1: CLASSIFICATION MODELS RESULTS

As shown in Table1(1) The best model was SVM with using TF-IDF as input. When I train the models, I notice that the best result I got is by using TF-IDF whiteout [AraVec and N-gram] as input. so. For the SVM I enter AraVec, N-gram, and TF-IDF to see the better result and tf-idf gives us the better result. In the last TF-IDF was the best method although it is one of the oldest ways in word embedding methods.

9. Reference:

[1] https://devopedia.org/natural-language-processing

[2] https://towardsdatascience.com/text-classification-applications-and-use-cases-beab4bfe2e62

[3]: *(PDF) CAMeL Tools: An Open Source Python Toolkit for Arabic Natural Language Processing*. Available from: https://www.researchgate.net/publication/341542088_CAMeL_Tools_An_Open_Source_Python_Toolkit_for_Arabic_Natural_Language_Processing [accessed Mar 23 2021].

[4]https://towardsdatascience.com/text-analysis-feature-engineering-with-nlp-502d6ea9225d

[5]: (PDF) *An Efficient Feature Selection Method for Arabic Text Classification*. Available from: www.researchgate.net/publication/272863884_An_Efficient_Feature_Selection_Method_for_Arabic_Text_Classification [accessed Mar 24 2021].

[6]: Arabic Topic Classification On The Hespress News Dataset. Availabel from :https://towardsdatascience.com/arabic-topic-classification-on-the-hespress-news-dataset-7adceef12bed [acceseed Mar 24 2021].

[7] TF-IDF from scratch in python on real world dataset. Available from: https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089 [accessed Apr 4, 2021].

[8]: *(PDF) AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP*. Available from: https://www.researchgate.net/publication/319880027_AraVec_A_set_of_Arabic_Word_Embedding_Models_for_use_in_Arabic_NLP [accessed Mar 23 2021].

[9]: *(PDF) Comparison of A*, Euclidean and Manhattan distance using Influence Map in Ms. Pac-Man.* Available from: https://www.diva-portal.org/smash/get/diva2:918778/FULLTEXT02.pdf [accessed April 10, 2021].