1) Abstract

Our project is about sentiment analysis on tweets in the Algerian Arabic language. We are trying to solve the lack of sentiment analysis tasks done in this language. We solve this problem by using as many methods as we could to determine the best accuracy for this task. The methods we used include: Naive Bayes, SVM, Logistic Regression, Random Forest, LSTM, LSTM with Aravec, and 2-step Classification. Our models did better than we expected, reaching a highest score of accuracy at 70% using 2-step classification.

2) Introduction

Due to the widespread use of the Internet and social media platforms, most languages are becoming digitally available. This allows for various artificial intelligence applications that enable tasks such as sentiment analysis, machine translation, and hateful content detection. But, many african languages do not have the curated datasets available for developing such applications.

There have been datasets that were created by the Lacuna Fund for some African languages for researchers to determine the suitability of current NLP methods and the development of techniques to maximize applications of such datasets.

There is a growing interest in sentiment analysis and many tasks have been done in this topic. But none of them included African languages. We have decided to implement sentiment analysis for the Algerian Arabic language from tweets. The dataset that we used to do this was from a competition: "AfriSenti-SemEval 2023 Shared Task 12: Sentiment Analysis for Low-resource African Languages using Twitter Dataset"

3) Related Work

A lot of studies have been performed in the sentiment analysis field for English and Only a few were about arabic especially dialects. Here are a summary of some research related to sentiment analysis of arabic dialects.

After the proliferation of social networks, considerable researches have been carried out on sentiment analysis.

First, Rushdi et al collected 500 reviews from blogs in Arabic with 500 reviews. The data was balanced. The method they applied was very diverse and rich. Among the methods used there was “support vector machines” (SVM) and “Naïve Bayes” (NB) . After they preprocessed the collected dataset, they used manual spelling correction, stemming, stop-words removal and N-Gram tokenization to process the data. After training the model, they were able to achieve 90% as accuracy although the low size of the dataset. This proves how sometimes it is necessary to do manual spelling correction and further processing to get better results.
Secondly, Mataoui et al performed a lexicon-based approach for sentiment analysis on the Algerian Arabic dialect. They built three lexicons with a polarity computation. The first lexicon was for the negation words, the second for intensification-words and a list of emoticons with their assigned polarities, and the last is a dictionary of common phrases of the algerian dialect. The final model day an accuracy of about 79%.
Finally, Dahou et al used deep learning approach for binary sentiment analysis classification where they applied Neural Network method on existing dataset. For the preprocessing, they used the word embedding on a 3.4 billion words corpus from a web-crawled corpus to train a CBOW model. They also used the Word2vec tool. Overall they achieved 89.6% as accuracy.

4) Data

Data description:

The dataset was provided as a part of the competition: “AfriSenti-SemEval 2023 Shared Task 12: Sentiment Analysis for Low-resource African Languages using Twitter Dataset”. It is a collection of twitter tweets written in the Algerian dialect labeled by sentiment. The dataset consists of three columns: ID (the unique identifier of each row), tweet, and label.

	ID	tweet	label

0	dz_train_01499	@user عادي خويا اسحاق علابالي بلي مافهمتنيش\n...	positive
1	dz_train_01630	@user كاين بزاف ولكن الزين غالب	positive
2	dz_train_00011	@user تبهليل el garghotti يستمر مع المتبردعين.	negative
3	dz_train_01062	@user زعنا زعما راكي تشوفي لي كومونتار توعنا و...	neutral

The dataset contains 1651 total rows with 0 null values. The distribution of the labels is as follows: 892 negative tweets, 417 positive tweets, and 342 neutral tweets.

Data preprocessing:

Label preprocessing is straightforward, the label is encoded from text to numbers as follows:

Positive → 1

Neutral → 0

Negative → -1

The tweet preprocessing consisted of 7 steps:

1- Removing @user and RT (Twitter tokens)

2- Tokenization

3- Removing stopwords

4- Removing punctuation

5- Removing numbers

6- Normalizing emojis: If an emoji is repeated in a tweet, it gets reduced to one emoji. The goal of this step is to reduce diversity between tweets.

7- Removing empty string tokens

Data train/test split:

The data is split to training 90% and testing 10%. The test set is balanced with 60 rows for each label.

Data augmentation:

Since the quantity of our data is relatively low, we used a well known text data augmentation technique that consists of word replacement based on contextualized word embeddings. The augmentation process goes as follows: We loop through the sentence words n times where n is

equal to the number of words in the sentence. In each iteration, the n word is replaced with the mask token ([MASK]) and then the sentence is passed to a fill-mask pipeline using `dziribert` (https://huggingface.co/alger-ia/dziribert?text=tahya+el+%5BMASK%5D) and the top 3 predictions from the model are returned and stored as candidates for that n word. After all the iterations are done, 3 new augmented variations of the original sentence are created by replacing the n predicted word with the original word where n is the number of the augmented sentence (1 to 3) for each word of the sentence.
One thing to note is that the augmentation is done only on the training set and not on the testing set, because the testing set is supposed to simulate the real data to give an accurate assessment of the performance of the model.

4) Methodology and Model Description

Naive Bayes

Methodology: Firstly, there will be a function that gets the counts of each label called priorDict, then all of the words in the tweets will be placed in a list for each label and from that we will get the frequency of each word. After that, a naive bayes function will be created and apply the naive bayes formula on each tweet in the test split, and label them according to the prediction made.

Model Description:

The naive bayes classifier is an algorithm that is used to classify based on features. It is a classifier that assumes independence among predictors, so it basically assumes that the presence of a particular feature in a class is unrelated to the presence of other features and predicts based on the statistics of how frequent the feature was present in each class.

SVM

Methodology: Firstly we are creating features which is extracting the Tweet and label variables, then since the test and train split is already initialized, we directly move to initializing the SVM classifier model and fitting it, after learning and fitting the classifier we can go ahead and make predictions and evaluate the model’s performance.

Model Description:

SVM or support vector machine is a linear model that works well with linear and non-linear problems and is used with practical problems. It is a methodology that uses a hyperplane or a line to separate data from one dimension to high dimensional space.

Logistic regression

Methodology: Firstly, there will be a function that gets the counts of each label called priorDict, then all of the words in the tweets will be placed in a list for each label and from that we will get the frequency of each word. After that, a logistic regression is created from the sklearn library and apply on each tweet in the test split, and label them according to the prediction made.

Model Description:

Logistic regression is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set. A logistic regression model predicts a dependent variable by analyzing the relationship between one or more existing independent variables. For example, a logistic regression could be used to predict whether a political candidate will win or lose an election or whether a high school student will be admitted or not to a particular college. These binary outcomes allow straightforward decisions between two alternatives.

Random forest

Methodology: Firstly, we created features using the TfidfVectorizer. After that, a random forest model is created from the sklearn library and apply on each tweet in the test split, and label them according to the prediction made.

Model Description:

Random Forest is a robust machine learning algorithm that can be used for a variety of tasks including regression and classification. It is an ensemble method, meaning that a random forest model is made up of a large number of small decision trees, called estimators, which each produce their own predictions. The random forest model combines the predictions of the estimators to produce a more accurate prediction.

LSTM

Methodology: Firstly, the tweets are encoded into vectors with each word represented by its index in the vocabulary. Then, in order for the tweets to be represented by vectors with the same shape, all the vectors are padded with zeros until the length of all the vectors is equal to the maximum length of the non-padded vectors. After the vectorization step is done, the data is used to train an LSTM based model.

Model Description:

The model consists of a total of 4 layers: An input layer representing the vectorized tweets, an embedding layer that returns an embedding vector of size 300, an LSTM layer of size 68, and finally, an output layer that outputs a vector of shape (3,1) representing the 3 probabilities of the tweet belonging to one of the three layers.

LSTM with Aravec

Methodology: Firstly, the tweets are encoded into vectors with each word represented by its index in the vocabulary. Then, in order for the tweets to be represented by vectors with the same shape, all the vectors are padded with zeros until the length of all the vectors is equal to the maximum length of the non-padded vectors. After the vectorization step is done, the data is used to train an LSTM based model and fine-tune the Aravec embeddings layer.

Model Description: The model consists of a total of 4 layers: An input layer representing the vectorized tweets, an embedding layer with size 300 that is initialized with the Aravec embeddings and fine tuned throughout the training, an LSTM layer of size 68, and finally, an output layer that outputs a vector of shape (3,1) representing the 3 probabilities of the tweet belonging to one of the three labels.

2-step Classification

Methodology: Firstly, the same vectorization and padding steps mentioned in the previous methods are done to encode tweets into vectors. Then, unlike the previous two approaches, the data is used to train 3 different models: Negativity detector, Positivity detector, and the Judge. As the name suggests, the Negativity detector model predicts whether a tweet is negative or not, and the same logic applies to the Positivity detector. The judge model takes the output of the two previous models and predicts whether the tweet is positive, negative or neutral. All of the three models are trained on different variations of the original dataset. 3

Negativity Detector Model Description: The model consists of a total of 4 layers: An input layer representing the vectorized tweets, an embedding layer with size 300 that is initialized with the Aravec embeddings and fine tuned throughout the training, an LSTM layer of size 68, and finally, an output layer that outputs a number from 0 to 1 representing how negative the tweet is.
Positivity Detector Model Description: The model consists of a total of 4 layers: An input layer representing the vectorized tweets, an embedding layer with size 300 that is initialized with the Aravec embeddings and fine tuned throughout the training, an LSTM layer of size 68, and finally, an output layer that outputs a number from 0 to 1 representing how positive the tweet is.
Judge Model Description: This model is a simple Neural Network model that consists the input layer that takes the output of the Negativity Detector model and the Positivity Detector model, and 3 hidden layers with relu as an activation function, and finally, an output layer that outputs a vector of shape (3,1) representing the 3 probabilities of the tweet belonging to one of the three labels.

6) Evaluation and Results

For evaluation, we used accuracy as the testing set is balanced with equal amounts of examples from each label.

Results

	SVM	Logistic Regression	Random Forest	Naive bayes	LSTM	LSTM with Aravec	2-step Classification
non-Augmented	37%	48%	46%	49%	61%	63%	70%
Augmented	41%	57%	62%	54%	NA	NA	NA

8) Conclusion and Discussion

Overall, the model built relied a lot on the text pre-processed. We first ran the data against our model using it the way we found it. The base models(SVM, Naive bayes and random forest) gave us an accuracy under 50%. This is a very interesting result for a base model. The main model(LSTM) gave a score of 61% but the the logistic regression did really bad. After seing that, we used the LSTM with Aravec to get a better result which we did. However, the best performance is the 2-step classification(70%), this means that this solves our problems and can detect the sentiment in a text better than any other models we built.

This study is the baseline for our future work, looking ahead, we could have more data to learn from since we noticed that the augmented data gave a better performance for some of our models. Additionally, it will be better if we work more in the preprocessing by removing the stopwords. This could improve our scores by removing all the words we are not learning a lot from.

References

https://www.rcs.cic.ipn.mx/rcs/2016_110/RCS_110_2016.pdf#page=55
https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.21598
https://aclanthology.org/C16-1228/
https://huggingface.co/alger-ia/dziribert?text=%D8%B1%D8%A8%D9%8A+%5BMASK%5D+%D8%AE%D9%88%D9%8A%D8%A7+%D9%84%D8%B9%D8%B2%D9%8A%D8%B2