Sentiment Analysis of Arabic Tweets

[Natural Language Processing Application]

By Mohamed Ahmed Hesham

Project Goals

- Creating the full pipeline of text sentiment analysis on any platform like: facebook or twitter
- Apply text pre-processing (cleaning, normalization, stemming, Word Embedding, ...etc.) on arabic language
- Using various Machine Learning and Deep Learning Techniques to get the highest sentiment analysis scores

Project Steps

1- Text Extraction
- Using Provided API (Post) to get balanced samples from each dialect for better training on all classes

2- Text Pre-processing
- Identification of Arabic letters
- Text Cleaning of unrequired words (e.g. email addresses, tags, hashtags, urls, punctuations, stop-words, non-arabic words)
- Text Preparation (e.g. Words Tokenization, Words Stemming, Word Embedding and Sequencing, Label Encoding of Classes)

3- Training on Text
- Using Suitable Machine Learning Algorithms
- Using Suitable Deep Learning Networks

Text Extraction

Due to slow extraction of tweets from API which require continuous extraction without any disconnection for about nearly 30 hours which is very hard to be done.

So, what should be done is to get balanced samples (1200 examples) from each dialect for better training on all classes to be totally 21.6K examples from overall 540K examples.

Text Pre-processing

- Text Cleaning of unrequired words (e.g. email addresses, tags, hashtags, urls, punctuations, stop-words, non-arabic words)
- Text Preparation
  Tokenization: converting sentences to vector of sentence words
  Stemming: converting every word to its root of meaning
  This process will help the model gain the dialect of each country and attitude of writing on twitter for easier classification
  Word Embedding: TF-IDF Vectorizer & Count-Vectorizer (explained in notebook)
  N-Gram Concept: sequence of N items from a given example of text to get the highest probabilities of followed sequence of words

Training on Text

- Machine Learning: As per the high no. of features and classes in selecting suitable algorithm and also, the used algorithm in the research paper (SVM and Logistic Regression Algorithms are used)

- Deep Learning: I have used fully connected neural network and its intended to use recurrent neural network to try its advantage of sequence probability calculation but, unfortunately RNN didn't ran due to google co-lab resources

Findings

- The scores are too low compared to models that are usable in real world due to the slow fetching of tweets as described before in the presentation
- The highest accuracies are achieved at unigrams whether in machine or deep learning while it should not be and the best should be in trigrams, but after thinking about the reason, I thought about that is due to compound verbs (e.g. Carry out, catch up, look for, ...etc.) that is occurring too much in English while it's not occuring at the same familiarity in Arabic language so, the best N-Gram model in Arabic text processing is Uni-Grams

Proof of Efficiency

To make sure of the efficiency of the model that can achieve high scores if there's a big dataset to be trained on the same preprocessing and modelling technique and changing some hyper-parameters to be compatible with modelling of this dataset,

This approach is found in proof_of_efficieny.ipynb , The scores in both Machine Learning Algorithms ~= 91 % and in Deep Learning Fully Connected Network, Accuracy is ~= 93 %

Proof of Efficiency

To make sure of the efficiency of the model that can achieve high scores if there's a big dataset to be trained on the same preprocessing and modelling technique and changing some hyper-parameters to be compatible with modelling of this dataset,

This approach is found in proof_of_efficieny.ipynb , The scores in both Machine Learning Algorithms ~= 91 % and in Deep Learning Fully Connected Network, Accuracy is ~= 93 %