NLP Report 1
Emotion Detection in Arabic Tweets

Mohamed Zaki 46-1939
Ahmed Attawia 46-4982

May 26, 2023

1 Introduction

The Arabic language, owing to its complex nature, offers many challenges in NLP tasks in comparison to English. Arabic grammar is extremely complex when compared to other languages, and many Arabic words consist of multiple parts each having its own meaning. Not to mention words that are inferred from the context such as hidden subjects or omitted objects. These characteristics provide us with a great challenge in tackling NLP tasks in Arabic and is an area of research that is not heavily tackled. In this project, we attempt to tackle the under-researched problem of emotion detection in Arabic tweets.

2 The Dataset

The dataset we will use in this project is the SemEval-2018 dataset[4]. This dataset is a compilation of more than 22,000 tweets in three languages: English, Arabic, and Spanish. This dataset is further divided into 5 sections tackling different NLP tasks. In this project, we are mainly concerned with the Arabic section and its fifth partition specifically, which tackles emotion detection. This section, from here on referred to as the dataset, is a collection of 4,000+ tweets written in Arabic. Each tweet has a label for the following emotions: Anger, Anticipation, Disgust, Fear, Joy, Love, Optimism, Pessimism, Sadness, Surprise, and Trust.

2.1 Data Preprocessing

To be able to effectively use the dataset in our project, some preprocessing had to be done first. The data preprocessing pipleine is as follows:

1. Removing English characters.
2. Removing digits.
3. Removing stop words.
4. Removing diacritics.
5. Normalizing Arabic text.
6. Removing Emojis.
7. Removing non Arabic characters.

2.2 Data Analysis

In this section, we attempt to do some simple analysis on the dataset to gain some insights and find some limitations.

2.2.1 How many tweets are in each emotion label?

In this figure, we can see the distribution of emotions in our dataset. We can see that the emotions Anticipation, Trust, and Surprise are not well represented in this dataset, and this could become a limitation later when trying to predict them.

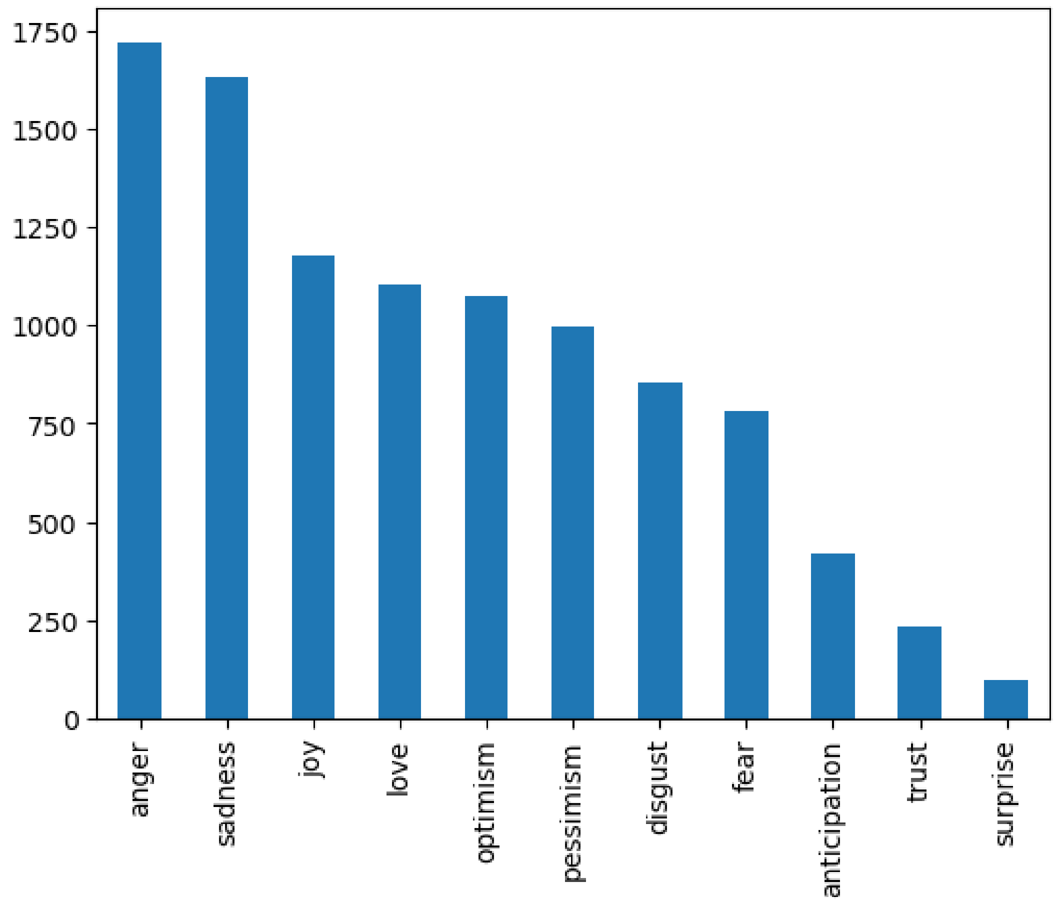


Figure 1: Tweet count for each emotion

0.2

2.2.2 The number of emotions per tweet

In this figure, we can see that the majority of tweets have more that one emotion detected. This can be beneficial as this way emotions are more accurately represented and more samples for each emotion is present.

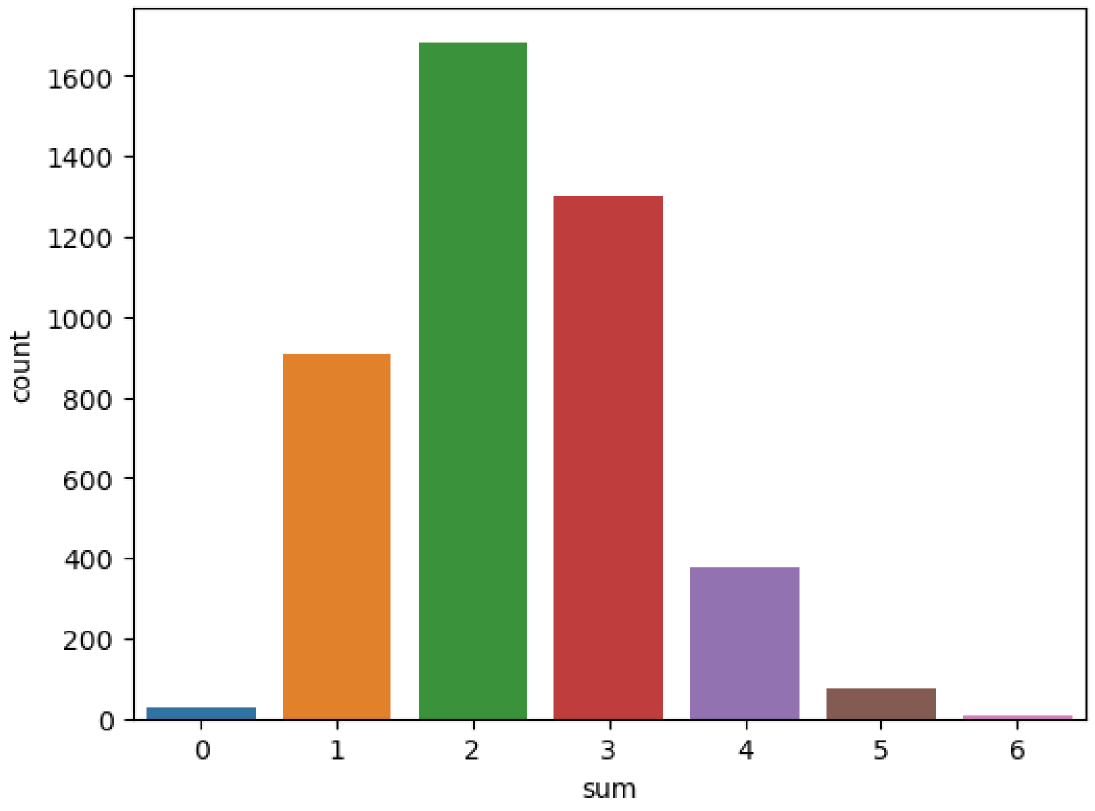


Figure 2: Count of emotion labels in tweets

2.2.3 Does tweet length correlate to number of emotions?

This figure was an attempt to find a relation between the length of the tweet and the number of emotions represented. The theory behind this was that as the tweet has more words in it, it could be describing more emotions. As we can see from the figure, this was disproven by the data.

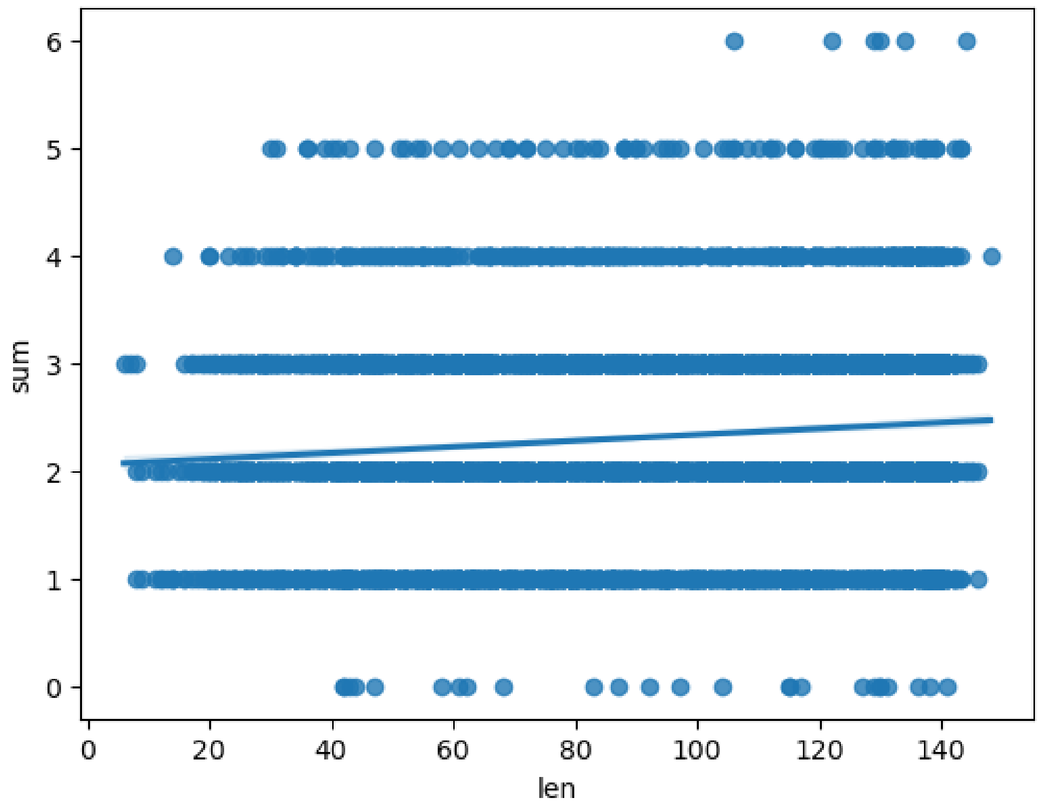


Figure 3: Relation between length of tweet and emotion count

3 System Architecture

Based on work from Mansy et al.[2] we will attempt to tackle this project by training three different models on the dataset to classify emotions, and then combining all 3 outputs using either an ensemble layer or a stacking approach where we feed the outputs into an ensemble layer and use that prediction as our final output. The models that will be used are as follows:

- **Bi-LSTM model:**Bi-directional LSTMs (Long Short-Term Memory) are a type of recurrent neural network that processes input sequences in both forward and backward directions using two separate hidden layers. This allows the network to capture both past and future context when making predictions, making it well-suited for tasks such as speech recognition and natural language processing.
- **Bi-GRU model:**Bi-directional GRUs (Gated Recurrent Units) are similar to bi-directional LSTMs in that they also process input sequences in both forward and backward directions. However, GRUs use a simpler architecture that replaces the LSTM's memory cell and output gate with a single update gate, making them quicker to train and more computationally efficient. Despite their simpler design, bi-directional GRUs can still be effective for tasks such as machine translation and speech recognition.
- **Pretrained language model (MARBERT Transformer):**A pretrained language model based on BERT has been introduced which is called MARBERT[1]. It was pretrained based on both MSA and Arabic dialects, unlike ARABERT which was pre-trained only on Arabic MSA. Because MARBERT transformer was used before in the emotion detection task, it has an emotion-related contextual understanding experience.

4 System Design

In this section, we will talk in further detail about the three models and the ensemble layer described in 3. We will attempt to walk you through our methodology and thinking for each model, concluding with the ensemble layer and the final output format.

4.1 The Bi-LSTM Model

As mentioned before, the first model in this system is a Bidirectional Long Short-Term Memory model. BiLSTMs have proven to be effective in NLP tasks owing to the fact that BiLSTMs can capture context in both the past and the future. This makes them suitable for emotions classification as the context of words related to the future and the past can have a great effect on their meaning and emotion. This model was designed using Keras in python, further details discussed below.

4.1.1 Model Design

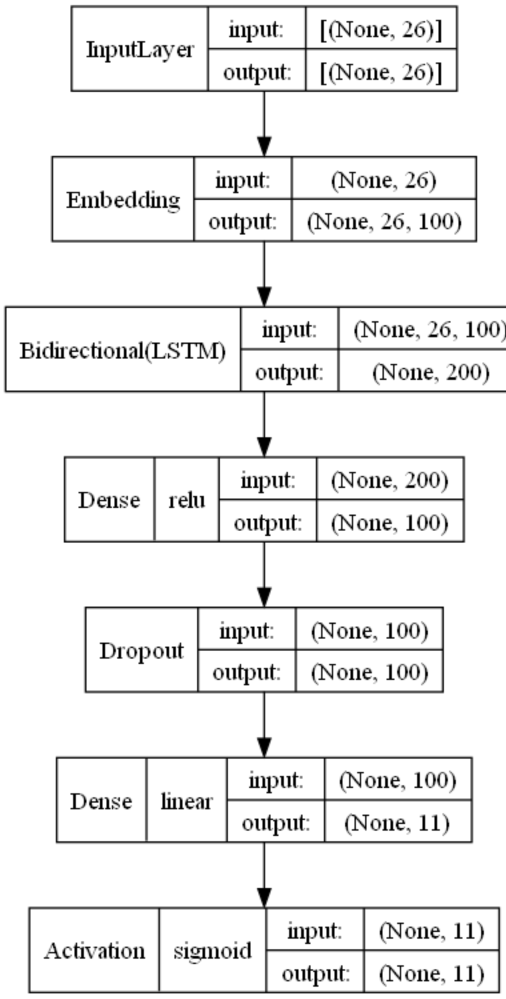

InputLayer | input: | [(None, 26)]
output: | [(None, 26)]
Embedding | input: | (None, 26)
output: | (None, 26, 100)
Bidirectional(LSTM) | input: | (None, 26, 100)
output: | (None, 200)
Dense | relu | input: | (None, 200)
output: | (None, 100)
Dropout | input: | (None, 100)
output: | (None, 100)
Dense | linear | input: | (None, 100)
output: | (None, 11)
Activation | sigmoid | input: | (None, 11)
output: | (None, 11)

Figure 4: LSTM Model design

As seen in 4, this model consists of 7 layers. The first layer in this model is an embedding layer. In this project, we used pretrained embeddings from the AraVec project[3]. Specifically, the AraVec unigrams Twitter-SkipGram with vector size 100. The use of a pretrained embedding in both our attempt and in Mansy et al.[2] is seen to improve the results from the model significantly.

After the embedding layer, we have the bidirectional LSTM layer itself. This layer is designed with a dropout value of 0.3 and a recurrent dropout of 0.2. This is followed by a dense layer with relu activation.

The next layer in this model design is a dropout layer. A dropout layer is used to attempt to tackle a problem we have in our dataset, which is that the data is unbalanced. The dropout layer tackles this by randomly setting some of the neurons in a layer to zero during each training iteration. This helps the network learn more robust features and reduces its reliance on any one set of inputs.

After the dropout layer we have another dense layer and then an activation layer with sigmoid activation. These two layers are the ones responsible for transforming the values currently in the neural network to an array of 11 binary values, one for each of the emotion classes we are trying to predict.

4.1.2 Input preparation & Training

To prepare the preprocessed input for this model, a tokenizer was designed over the train and test dataset. After converting all text inputs into their respective vectors using this tokenizer, post padding and truncating was applied with a target length of 26. This input was then fed into the model and trained for 20 epochs.

4.2 The GRU Model

In this section, we will discuss the second model in the architecture. This model is similar in its design and approach to the previous model. However, GRUs provide many advantages in comparison to LSTMs. One of the main benefits of using GRUs over LSTMs is that they have fewer parameters, which can make them easier to train and less prone to overfitting. This is done by combining the input and forget gates into a single update gate. Furthermore, This makes GRUs faster to train that LSTMs.

4.2.1 Model Design

As can be seen in 5. This model has a design very similar to the BiLSTM model. The only two differences in this model is the use of a Bidirectional Gated Recurrent Unit layer after the embedding layer, and the use of softmax activation in the last layer.

4.2.2 Input preparation & Training

The inputs to this model were prepared using the same tokenizer as the one used in the BiLSTM model. The model was trained for 10 epochs over the train data.

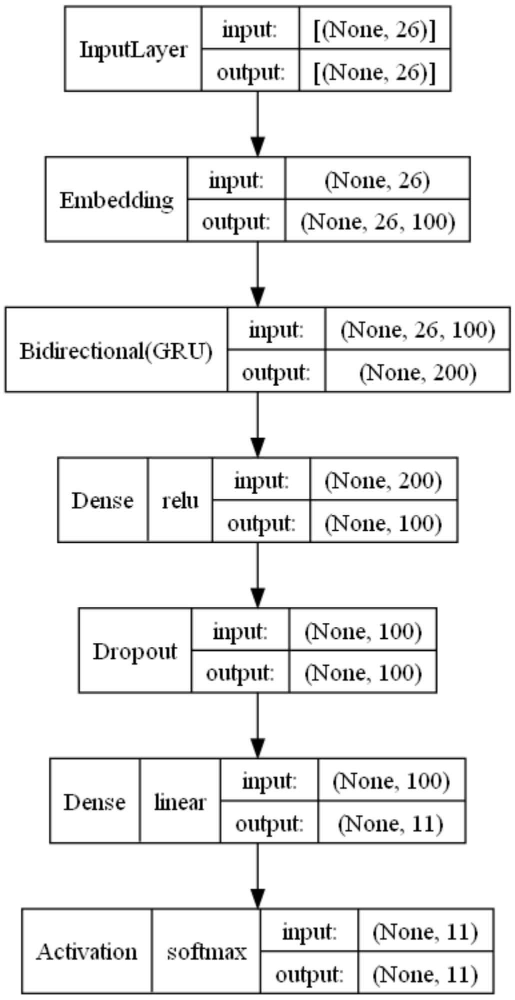


Figure 5: LSTM Model design

4.3 The MARBERT Model

MARBERT is a large scale pre-training masked language model focused on both Dialectal Arabic (DA) and Modern Standard Arabic (MSA). MARBERT is pretrained on a dataset of more that 1B tweets, and so it is very suitable to the task at hand.

4.3.1 Model Design, Input preparation, & Training

To prepare the MARBERT model for use in our architecture, We fine-tuned the model on our own training dataset for 4 epochs, using batch sizes of 16 and a learning rate of 1e-4. These hyperparameters are recommended by Google for fine-tuning BERT models, and have been shown to yield good results in a wide range of NLP tasks.

To prepare the input text, instead of using our own tokenizer, the MARBERT pretrained tokenizer is used instead. This ensures that the input text is properly tokenized and encoded, so that it can be effectively processed by the model.

4.4 The Ensemble Layer

The ensemble layer in this architecture is designed as a simple mathematical formula. This formula assigns a specific weight to each model from the 3 models above. This weight is then multiplied by the prediction labels resulting from each model. The results are then added together and a cutoff value of 0.5 is used to result in the final prediction. The weights used are 0.52, 0.32, and 0.23 for MARBERT, GRU, and LSTM respectively.

$$Pred = \left\{ \begin{array}{ll} 1, & \text{for } (0.52 * MARBERT + 0.32 * GRU + 0.23 * LSTM) \geq 0.5 \\ 0, & \text{for } (0.52 * MARBERT + 0.32 * GRU + 0.23 * LSTM) < 0.5 \end{array} \right\}$$

5 Results

The results from our approach, as seen below, show that the system architecture presented in this project is well suited to the task of emotion detection in arabic tweets. Furthermore, we can see that each individual model on its own is also suitable for the task.

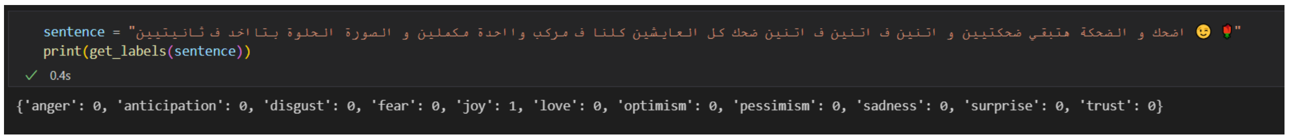
In the following table, we see the results in detail:

	Accuracy	**Precision**	**Recall**	**F1 Score**
BiLSTM	0.8365	0.6564	0.5058	0.5713
BiGRU	0.8246	**0.7849**	0.2554	0.3854
MARBERT	**0.8569**	0.7191	0.5505	0.6236
Ensemble	0.8567	0.7028	**0.5801**	**0.6356**

As we can see from the Table, all models performed well when evaluated over the dataset. We can see that the ensemble layer, while not the best in accuracy and precision, did perform extremely well overall and was not lacking in any metric. We can conclude from these results that the ensemble approach has benefits over using a single model in our task.

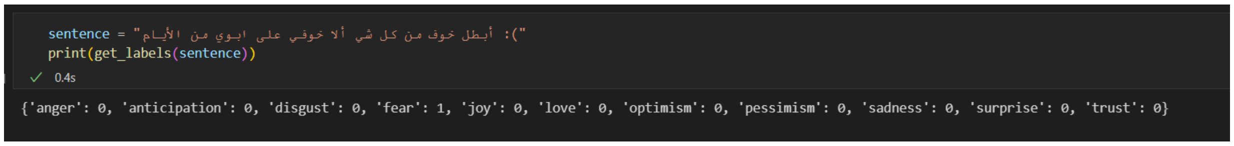
5.1 Examples

Here are some real examples of predictions done by our model:



```
sentence = "اضحك و الضحكة هتبقي ضحكتيين و اتنين ف اتنين ف اتنين ضحك كل العايشين كلنا ف مركب وااحدة مكملين و الصورة الحلوة بتااخد ف ثانيتيين 😉🌹"
print(get_labels(sentence))
✓ 0.4s
{'anger': 0, 'anticipation': 0, 'disgust': 0, 'fear': 0, 'joy': 1, 'love': 0, 'optimism': 0, 'pessimism': 0, 'sadness': 0, 'surprise': 0, 'trust': 0}
```

Figure 6: Example 1



```
sentence = "أبطل خوف من كل شي ألا خوفي على ابوي من الأيام :("
print(get_labels(sentence))
✓ 0.4s
{'anger': 0, 'anticipation': 0, 'disgust': 0, 'fear': 1, 'joy': 0, 'love': 0, 'optimism': 0, 'pessimism': 0, 'sadness': 0, 'surprise': 0, 'trust': 0}
```

Figure 7: Example 2

As we can see in the example, our model takes in as input an arabic sentence with no preprocessing, and returns a dictionary with the emotions and their predictions.

6 How to recreate

To recreate the results clone our repo present Here, download our MARBERT model available Here into the same directory, and download the dataset available Here into the same directory. After that, you can simply run the Ensemble.ipynb notebook to load all three models, generate predictions, and try out sentences from any external source. The other notebooks were the ones used in training each model, and preprocessing the data. You may use those if you want to edit the model designs or train for a different number of epochs, but you do not need to.

7 Limitations

In this project, we faced many computational limitations and as such could not find the best hyper-parameters for the MARBERT model and had to use the recommended ones. And we could not run an effective grid search for the ensemble weights and so we may not be using the most efficient ones.

References

[1] Abdul-Mageed, M., Elmadany, A., and Nagoudi, E. M. B. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (Online, Aug. 2021), Association for Computational Linguistics, pp. 7088–7105.

[2] Mansy, A., Rady, S., and Gharib, T. An ensemble deep learning approach for emotion detection in arabic tweets. *International Journal of Advanced Computer Science and Applications 13* (01 2022).

[3] Mohammad, A. B., Eissa, K., and El-Beltagy, S. Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science 117* (11 2017), 256–265.

[4] Mohammad, S. M., Bravo-Marquez, F., Salameh, M., and Kiritchenko, S. Semeval-2018 Task 1: Affect in tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)* (New Orleans, LA, USA, 2018).