WANLP 2021 Shared Task: Towards Irony and Sentiment detection in Arabic tweets using Multi-headed-LSTM-CNN-GRU and MaRBERT

Reem Abdel-Salam
Computer Engineering, Cairo University
reem.abdelsalam13@gmail.com

Abstract

Irony and Sentiment detection is important to understand people's behavior and thoughts. Thus it has become a popular task in natural language processing (NLP). This paper presents results and main findings in WANLP 2021 shared tasks one and two. The task was based on the ArSarcasm-v2 dataset (Abu Farha et al., 2021). In this paper, we describe our system Multi-headed-LSTM-CNN-GRU and also MARBERT (Abdul-Mageed et al., 2021) submitted for the shared task, ranked 10 out of 27 in shared task one achieving 0.5662 F1-Sarcasm and ranked 3 out of 22 in shared task two achieving 0.7321 F1-PN under CodaLab username "rematchka". We experimented with various models and the two best performing models are a Multi-headed CNN-LSTM-GRU in which we used prepossessed text and emoji presented from tweets and MARBERT.

1 Introduction

Text on social media as Twitter and Facebook involves the usage of expressive and figurative languages like irony and sarcasm. A text is considered ironic if it is intended to mean the opposite of what it literally expresses. While sentiment is the interpretation and classification of emotions (positive, negative, and neutral) within text. Irony detection has recently become more relevant due to its importance in extracting information from texts. For example, In order to move beyond the literal match of user queries, irony detection will enhance information retrieval with new operators to allow non-literal retrieval of creative expressions. Sentiment analysis has been an important topic in the literature for a while. The interest in Sentiment analysis arises due to the wide and increase in usage of social media platforms, and online shopping. Companies can benefit from understanding the feedback and thoughts of their customers. The sixth workshop for Arabic Natural Language Processing co-located with EACL 2021, had two shared tasks: Sarcasm and Sentiment Detection in Arabic tweets. The shared task on sarcasm detection is a binary classification where it is required to identify whether a piece of text is sarcastic or not. While shared task in sentiment detection is multi-class classification, where it is required to identify sentiment of a piece of text as positive or negative or neutral. In this paper, we describe the work we performed on both shared tasks. MARBERT, and Multi-headed-LSTM-CNN-GRU models [1] were used in this study. The rest of this paper is structured as follows: section 2 discusses the related work that has been performed on automatic sarcasm and Irony detection, section 3 describes the dataset used in this shared task, section 4 discusses the approach we used in our study, and section 6 discusses the results we obtained.

2 Related Work

There have been large attempts in the literature to tackle irony and sentiment detection. Most of the work is biased towards machine learning approaches, with few attempts to use deep learning approaches. A Recent work (Abdul-Mageed et al., 2021) proposed two powerful bidirectional based transformers models ARBERT and MARBERT. The models were trained and evaluated based on Arbenchmark which consists of 41 datasets targeting 5 different tasks/task clusters including irony and sarcasm tasks. In (Karoui et al., 2017), the authors used a random forest classifier based on some set of features, in which they believe are powerful, for irony detection. The set of features are surface features, Sentiment features, Shifter fea-

[1]The source code for the developed models can be found through: https://github.com/rematchka/Irony-and-Sarcasm-detection-in-Arabic-tweets

tures, and Contextual features. In another work (Khalifa and Hussein, 2019), the authors used ensemble learning in irony detection in Arabic tweets. The ensemble learning consists of three ensemble models. The first ensemble model uses different machine learning classifiers which uses TF-IDF word n-gram features, topic modeling features, bag-of-words, and sentiment features. The second ensemble model uses 8 different Word-level Bi-LSTM while the third ensemble model is a combination of both the first and second ensemble models. Moreover Alayba et al. (2018) proposed CNN-LSTM model for Arabic semantic analysis. The authors tested their model on two datasets, ArTwitter (Abdulla et al., 2013) and Arabic Health Services (Alayba et al., 2017) datasets, where they achieved accuracies of 88.1% and 94.3% respectively. Elshakankery and Ahmed (2019) proposed HILATSA for Arabic semantic analysis. HILATSA combines lexicon-based and machine learning approaches. The authors experimented with several datasets in their work as ASTD (Nabil et al., 2015) and ArTwitter. The authors made use of Different classifiers, which varied from using classical machine learning to deep learning models. In another study (Al-Smadi et al., 2019), the authors proposed two models: a character-level bidirectional LSTM along with a conditional random field classifier (Bi-LSTM-CRF) for aspect opinion target expressions extraction, and an aspect-based LSTM for aspect sentiment polarity classification. They tested their model on Arabic hotels' reviews (Elnagar et al., 2018) dataset, they achieved an F-score of 70%.

3 Data

This Section shows the used dataset in the shared task. We used ArSarcasm-v2 dataset (Abu Farha et al., 2021), which was provided in the training phase by ArSarcasm Shared Task. The dataset is a Twitter text with labels "FALSE" and "TRUE" for the first shared task and "POS", "NEG", and "NEU" for the second shared task. The dataset provided in the training phase contains 12548 training examples, while the test set provided in the test phase consists of 3000 examples.

Table 1 shows the statistics of ArSarcasm-v2 dataset. Most of the dialect examples presented in the dataset are either MSA or Egyptian, while there are few examples of the Maghrebi dialect. Based on labels distribution, the dataset provided is imbalanced, since it has 2168 examples for True around 17.3% and 10380 examples for False around 82.7%, in irony detection task, and 4621, 5747, 2180 around 36.8%, 45.8% and 17.4% for Negative, Neutral, Positive examples, in sentiment detection task, respectively. Emoji distribution in the provided training data was analyzed, it was found that there are 1524 examples that contain emojis, where the maximum length was 219 emojis. Therefore it could be concluded that there are examples of pure emoji or do not contain a large amount of text.

4 Methodology

This section discusses the details required for reproducing the results. It mentions the preprocessing steps, the architecture of the classifiers used, and hyperparameter values.

4.1 Preprocessing

Processing data is essential in order to extract useful information and improve data quality, which directly affects model performance. Especially when dealing with text messages from various dialects and non-standard language. Our processing pipeline goes as follows:

1. Removal of URLs, mentions, emails, dates, numbers, punctuation, English letters, stop words found in NLTK (Bird, 2006), and the emojis.
2. Diacritic removal, letter normalization
3. Conversion of emojis presented into text.
4. Unifying similar Arabic characters such as Hamza and Yaa

4.2 Proposed Models

In this section, we show the key models used in the two shared tasks. We used the Multi-headed-LSTM-CNN-GRU model and MARBET (Abdul-Mageed et al., 2021) as our baseline models. In addition to other experimental models that were not effective based on train-validation-test split used.

4.2.1 Multi-headed-LSTM-CNN-GRU Model

In this model, Arabic news embedding (Altowayan and Tao, 2016) was used as text representation in addition to emoji2vec (Eisner et al., 2016) for emoji information presented in the examples. These embeddings were then fed to a deep learning model which comprises the Bidirectional LSTM-CNN model, Bidirectional GRU-CNN model, and CNN-LSTM model. Processed clean text and emojis

Dialect	Non-Sarcastic	Sarcastic	Negative	Neutral	Positive	Total
msa	7634	928	2671	4486	1405	8562
egypt	1745	930	1376	793	506	2675
gulf	487	157	264	259	121	644
levant	486	138	285	197	142	624
magreb	28	15	25	12	6	43
Total	10380	2168	4621	5747	2180	12548

Table 1: Dataset statistics for sarcasm and sentiment over the dialects and labels

Model	Accuracy	F1-Score	Precision	Recall	Shared Task
MARBERT	0.944	0.899	0.892	0.908	1
MARBERT	0.8730	0.858	0.856	0.861	2
Multi-headed-LSTM-CNN-GRU	0.955	0.9259	0.928	0.924	1
Multi-headed-LSTM-CNN-GRU	0.873	0.795	0.699	0.923	2
Multi-headed-LSTM-CNN-GRU with TF-IDF 2-grams	0.822	0.762	0.776	0.752	1
CNN-LSTM with dialect information	0.836	0.834	0.834	0.834	1

Table 2: Results on our initial development set in shared task one and two, based on our splitting criteria when evaluating models

Model	F1-PN	Accuracy	Macro-F1	Precision	Recall
MARBERT	0.7321	0.6957	0.6587	0.6498	0.6748

Table 3: Leaderboard results on Provided test set for shared task two

Model	F1-Sarcastic	Accuracy	Macro-F1	Precision	Recall
MARBERT	0.5662	0.7803	0.7095	0.7231	0.7004
Multi-headed-LSTM-CNN-GRU model	0.1657	0.7047	0.4932	0.5497	0.5185

Table 4: Leaderboard results on provided test set for shared task one

were inputted to the network. Clean text is processed in two branches Bidirectional LSTM-CNN model and the Bidirectional GRU-CNN model. The output of both branches are concatenated and passed through the batch norm layer, then dense layer. emoji is processed using the CNN-LSTM model. The output of the processed emoji layer is concatenated with a dense layer. At this step, the output is passed to two fully connected layers as shown in figure1.

4.2.2 MARBERT Model

MARBERT model (Abdul-Mageed et al., 2021) was fine-tuned on the provided dataset. MARBERT model is a bidirectional transformer-based model.

4.2.3 Various Deep learning models

For shared task one and two, experiments were conducted on other models as CNN-LSTM model with Arabic news embedding, CNN-LSTM model with emoji embeddings, another version of Multi-headed CNN-LSTM-GRU where one input was processed text and the other input was dialect or TF-IDF of 2-grams.

4.2.4 Machine learning models

Experiments were conducted on SVM model and logistic regression model. The pipeline goes as follows: examples were processed as discussed in the previous subsection. For each example and its extracted emoji (if exists) we get their vector rep-

Model	F1-Sarcastic	Accuracy	Macro-F1	Precision	Recall
Multi-headed-LSTM-CNN-GRU model	0.42	0.69	0.60	0.43	0.41
Multi-headed-LSTM-CNN-GRU with TF-IDF 2-grams	0.30	0.72	0.56	0.47	0.23
Multi-headed-LSTM-CNN-GRU with Dialect information	0.18	0.73	0.51	0.54	0.11
SVM with concatenated features	0.47	0.52	0.51	0.34	0.79
Linear regression with add features	0.46	0.49	0.49	0.32	0.78

Table 5: Non official results on provided test set for shared task one, after competitions has ended

Model	F1-PN	Accuracy	Macro-F1	Precision	Recall
Multi-headed-LSTM-CNN-GRU model	0.55	0.527	0.48	0.48	0.49
SVM with concatenated features	0.54	0.55	0.47	0.47	0.46
Linear regression with concatenated features	0.5	0.53	0.45	0.45	0.46

Table 6: Non official results on provided test set for shared task two, after competitions has ended

resentation based on Arabic news embeddings and emoji2vec embeddings. We then obtain features either by concatenating examples vector representation with emoji vector representation or by adding them together.

5 Experimental Setup

Experiments were conducted via Python and Tensorflow framework, running on Google Colab resources, which are Nvidia Tesla P100-PCIE-16GB GPU, Intel ® Xeon ® CPU @ 2.20 GHz, and 12GB RAM. We used 70%-10%-20% strategy for train-validation-test splits respectively for the training phase. Eventually, we had 8783 examples for the training set, 753 examples for the validation set, and 3012 examples for the test set. All of the presented models were trained on the same dataset and splitting criteria. In the Multi-headed CNN-LSTM-GRU model, weights were initialized with He Initialization. Adam optimizer with a 0.0001 learning rate was used, with cross-entropy as a loss function. The model was trained for 150 epochs which took around 2-3 hours of training, with 32 batch sizes. For the MARBERT model, it was fine-tuned for 5 epochs using the initial learning rate 2e-06, with a batch size of 32 with Adam weight decay optimizer and cross-entropy loss function. Precision, recall, f-score (f1-sarcastic for shared task one, and macro average f1-score for shared task two), and accuracy were used as evaluation metrics.

6 Results and Discussion

6.1 Development phase results

Table 2 shows that Multi-headed-LSTM-CNN-GRU outperforms MARBERT based on our splitting criteria in shared task one, while MARBERT outperforms in shared task two. We believe that if ensemble learning were to be used, it will perform better as it will incorporate all information needed in both shared tasks from dialect, processed text, and emojis. We didn't include SVM and linear regression model scores as their F1-Sarcastic were below 0.2.

6.2 Official leaderboard results

The leaderboard results for shared task one official metric was based on F-score of the sarcastic class, while for shared task two, it was based on F-PN (Macro average of the F-score of the positive and negative classes). precision, recall, f-score, accuracy were also used for evaluation. As shown in table 4, based on the leaderboard in shared task one MARBERT model achieved 0.5662 F1-Sarcastic,

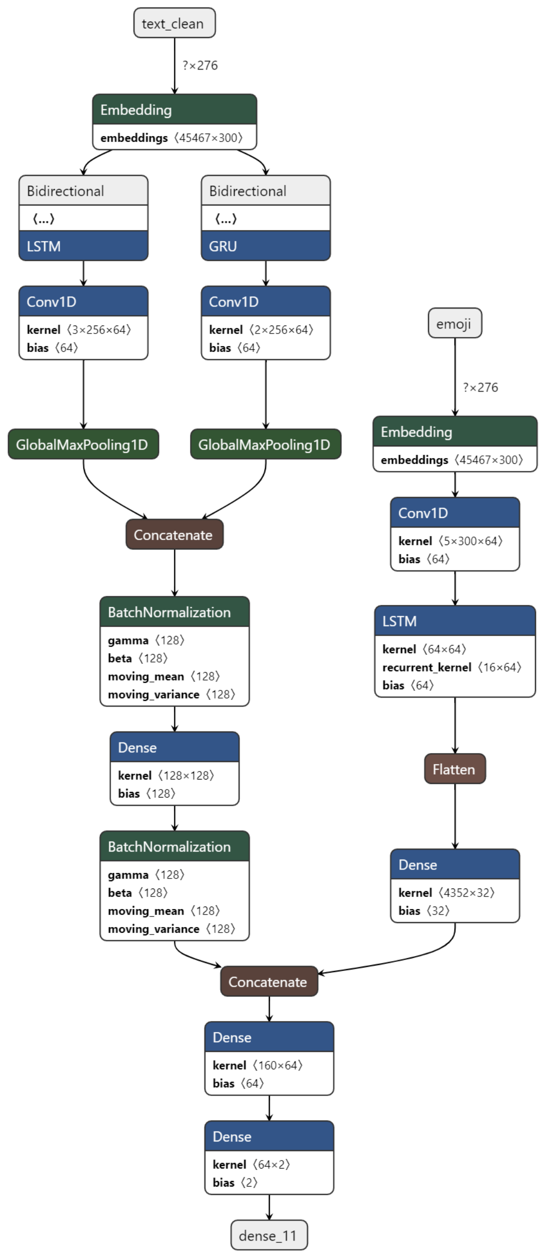

Figure 1: Model architecture

0.7803 Accuracy, 0.7095 Macro-F1, 0.7231 Precision, and 0.7004 recall, placing in 10th place among the submitted models, while Multi-headed-LSTM-CNN-GRU model achieved 0.1657, 0.7047, 0.4932, 0.5497, and 0.5185 for F1-Sarcastic, Accuracy, Macro-F1, Precision, Recall respectively, being placed in 27th place in the leaderboard.

The MARBERT model outperforms the Multi-headed-LSTM-CNN-GRU in F1-Sarcastic metric and has 8% accuracy loss on the test set. We believe that this shake in the leaderboard happened as MARBERT trained on multiple datasets and uses dialect information in addition to text, while Multi-headed-LSTM-CNN-GRU uses emoji and text. It was found that the distribution of emoji in the test set examples is not high as in train set examples, around 487 examples, with a maximum length of 71 emoji. In shared task two as shown in table 3 based on the leaderboard, MARBERT model achieved 0.732 F1-PN, 0.6957 accuracy, 0.6587 Macro-F1, 0.6498 precision, and 0.6748 recall on the leaderboard.

6.3 Non-official results

After the submission, we evaluated our models, on the provided test set and their annotations. As shown in table 5, for shared task one, a slightly modified version from the Multi-headed-LSTM-CNN-GRU model could achieve a higher F1-Sarcastic score. We believe that by fine-tuning the model by incorporating class imbalance, emoji distribution, and architecture tuning could achieve a competitive accuracy, F1-score as MARBERT model. Surprisingly, SVM model outperformed the Multi-headed-LSTM-CNN-GRU model on shared task one. For shared task two, table 6 shows Multi-headed-LSTM-CNN-GRU model, SVM, and logistic regression performance on test set. It could be concluded that most of models in shared task two have similar F1-PN, and accuracy score. Multi-headed-LSTM-CNN-GRU model performed well on shared task two than on shared task one.

7 Conclusion

In this paper, the results and the main findings of ArSarcasm shared task one and two, on identifying irony and sentiment in Arabic tweets were presented, in which different experiments were carried out with MARBERT and Multi-headed-LSTM-CNN-GRU model. Information presented in text as emojis were used to improve model performance. However, hyperparameter optimization and the usage of pre-trained word embeddings for the Multi-headed-LSTM-CNN-GRU network showed its effectiveness and impressive results for a small model compared to the pre-trained MARBERT model based on Bert and transformers. Future work will include investigating text representations, improving the deep learning model, and exploring different architectures.

References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. Arbert & marbert: Deep bidirectional transformers for arabic. *ArXiv*, abs/2101.01785.

Nawaf A. Abdulla, N. A. Ahmed, Mohammed A. Shehab, and Mahmoud Al-Ayyoub. 2013. Arabic sentiment analysis: Lexicon-based and corpus-based. *2013 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, pages 1–6.

Ibrahim Abu Farha, Wajdi Zaghouani, and Walid Magdy. 2021. Overview of the wanlp 2021 shared task on sarcasm and sentiment detection in arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*.

Mohammad Al-Smadi, Bashar Talafha, Mahmoud Al-Ayyoub, and Y. Jararweh. 2019. Using long short-term memory deep neural networks for aspect-based sentiment analysis of arabic reviews. *International Journal of Machine Learning and Cybernetics*, pages 1–13.

Abdulaziz M. Alayba, V. Palade, Matthew England, and R. Iqbal. 2018. A combined cnn and lstm model for arabic sentiment analysis. *ArXiv*, abs/1807.02911.

Abdulaziz M. Alayba, Vasile Palade, Matthew England, and Rahat Iqbal. 2017. Arabic language sentiment analysis on health services. *CoRR*, abs/1702.03197.

A. Aziz Altowayan and L. Tao. 2016. Word embeddings for arabic sentiment analysis. *2016 IEEE International Conference on Big Data (Big Data)*, pages 3820–3825.

Steven Bird. 2006. Nltk: The natural language toolkit. *ArXiv*, cs.CL/0205028.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bosnjak, and S. Riedel. 2016. emoji2vec: Learning emoji representations from their description. *ArXiv*, abs/1609.08359.

Ashraf Elnagar, Yasmin S. Khalifa, and Anas Einea. 2018. Hotel arabic-reviews dataset construction for sentiment analysis applications.

Kariman Elshakankery and M. Ahmed. 2019. Hilatsa: A hybrid incremental learning approach for arabic tweets sentiment analysis. *Egyptian Informatics Journal*, 20:163–171.

Jihen Karoui, F. Benamara, and Véronique Moriceau. 2017. Soukhria: Towards an irony detection system for arabic in social media. In *ACLING*.

M. Khalifa and Noura Hussein. 2019. Ensemble learning for irony detection in arabic tweets. In *FIRE*.

Mahmoud Nabil, Mohamed A. Aly, and A. F. Atiya. 2015. Astd: Arabic sentiment tweets dataset. In *EMNLP*.