The IDC System for Sentiment Classification and Sarcasm Detection in Arabic

Abraham Israeli,[1,2] Yotam Nahum,[1] Shai Fine[1], Kfir Bar[1]
[1]The Data Science Institute, Interdisciplinary Center, Herzliya, Israel
[2]Department of Software and Information System Engineering
Ben-Gurion University of the Negev, Beer-Sheva, Israel

{abraham.israeli,yotam.nahum,kfir.bar}@post.idc.ac.il,
shai.fine@idc.ac.il

Abstract

Sentiment classification and sarcasm detection attract a lot of attention by the NLP research community. However, solving these two problems in Arabic and on the basis of social-network data (i.e., Twitter) is still of lower interest. In this paper we present designated solutions for sentiment classification and sarcasm detection tasks that were introduced as part of a shared task by Abu Farha et al. (2021). We adjust the existing state-of-the-art transformer pretrained models for our needs. In addition, we use a variety of machine-learning techniques such as down-sampling, augmentation, bagging, and usage of meta-features to improve the models performance. We achieve an F1-score of 0.75 over the sentiment classification problem where the F1-score is calculated over the positive and negative classes (the neutral class is not taken into account). We achieve an F1-score of 0.66 over the sarcasm detection problem where the F1-score is calculated over the sarcastic class only. In both cases, the above reported results are evaluated over the *ArSarcasm-v2*–an extended dataset of the ArSarcasm (Farha and Magdy, 2020) that was introduced as part of the shared task. This reflects an improvement to the state-of-the-art results in both tasks.

1 Introduction

Sarcasm and sentiment detection are two central tasks in the research field of natural language processing (NLP). Sentiment classification can be defined as the process of estimating emotion polarity of a given piece of text (Liu, 2012). Sarcasm detection is usually treated as the process of determining whether a piece of text was sarcastically written or not. The Free Dictionary[1] defines sarcasm as a form of verbal irony that is intended to express contempt or ridicule. According to Wilson (2006), sarcasm is correlated with expressing the author's opinion in an indirect form, where the intended meaning is different from the literal one.

Although these two tasks are highly related to each other and are usually solved by using a classification model, they differ in significant nuances. Sarcasm heavily dependent on *context* – in most of the cases, a given sentence can be interpreted as sarcastic only if it is written (or said) in a specific context. Moreover, a speaker will use sarcastic language in cases where she thinks that her words will be interpreted as so (Joshi et al., 2017). In many cases, sarcasm detection is treated as a sub-task of the broader problem of sentiment analysis. It does not mean that sarcasm detection becomes redundant if the sentiment classification problem is solved accurately enough, since sarcastic language remains the biggest challenge for sentiment-analysis models. The reason behind this is the fact that a sarcastic language usually carries a negative implicit sentiment, while it is expressed using positive expressions (Bouazizi and Ohtsuki, 2016). The nuanced differences between the two tasks motivated us to take the approach of building two distinct solutions.

There is an active research area on sarcasm and sentiment analysis, but it is mostly done on English (Joshi et al., 2017; Bharti et al., 2016; Zhang et al., 2018). The research efforts that focus on sarcasm and sentiment classification in Arabic is limited. However, with the recent growing interest in non-English NLP, processing Arabic becomes one of

[1]https://www.thefreedictionary.com/

Proceedings of the Sixth Arabic Natural Language Processing Workshop, pages 370–375
Kyiv, Ukraine (Virtual), April 19, 2021.

	Dialect (%)					Sentiment (%)			Sarcasm (%)		
Dataset	MSA	Egypt	Levant	Gulf	Magreb	Pos	Neu	Neg	True	False	Total
$train_{arsar}$	67.0	22.5	5.2	4.9	0.3	15.9	50.6	33.5	15.9	84.1	10587
$train_{\overline{arsar}}$	75.0	14.6	3.6	6.2	0.5	25.1	20.3	54.6	24.3	75.7	2001
$test$	77.4	10.2	1.6	10.7	0.1	-	-	-	-	-	3000

Table 1: Data statistics. *ArSarcasm-v2* (15587 tweets) is the complete corpus published by Abu Farha et al. (2021). It consists of three parts as demonstrated in the table. $train_{arsar}$ denotes $train$ data that is taken from the ArSarcasm dataset (Farha and Magdy, 2020) while $train_{\overline{arsar}}$ is the complementary part of the $train$ set. All numbers besides the last column are percentages. Last column is the actual number of instances in each dataset. The $test$ dataset sentiment and sarcasm annotation was unknown while building the models and was revealed after the shared task ended.

the most active research topics (Bouamor et al., 2018; Farha and Magdy, 2019; Obeid et al., 2020).

For the last two decades, social networks (e.g., Twitter,Reddit) became a central communication tool for humans. These data sources enable the usage of textual data for sarcasm detection and sentiment classification models building.

In this paper, we present our solution to the shared task of sarcasm and sentiment detection in Arabic, organized as part of the the sixth workshop for Arabic NLP (Abu Farha et al., 2021). We implemented two distinct algorithms for each task (sentiment and sarcasm). Both algorithms use some of the recent advanced transfer-learning models followed by traditional machine-learning techniques (e.g., usage of meta-feature, down-sampling).

2 Data

As part of the shared task, the *ArSarcasm-v2* dataset is introduced (Abu Farha et al., 2021). The dataset consists of two separate datasets–$train$ and $test$. We note that the ArSarcasm (Farha and Magdy, 2020) dataset (10587 tweets) is a *subset* of the $train$ dataset. We denote the two parts of the $train$ dataset as $train_{arsar}$ and $train_{\overline{arsar}}$ throughout the paper. Each tweet in the $train$ dataset is additionally tagged with a dialect value (five Arabic dialects), a sentiment value (three classes), and whether the tweet is sarcastic or not (two classes). The $test$ dataset contains 3K tweets for which only the dialect value is available. Dataset statistics are provided in Table 1. As observed in the table, $train_{arsar}$ and $train_{\overline{arsar}}$ distribute differently. For both tasks, a subset of the train dataset was used as an evaluation set (i.e, $eval$). However, the $train$ and $eval$ datasets were differently created for each of the tasks.

2.1 Sentiment Classification Data

For the sentiment classification task, we randomly split the original $train$ dataset into a 83%-17% $train$-$eval$ parts. We used stratified sampling to ensure an equal distribution of the label (i.e., sentiment) over the two datasets.

2.2 Sarcasm Detection Data

Due to the unique challenges of the sarcasm-detection task, we process the data in two steps: (i) Down-Sampling, and (ii) Augmentation.

Down-Sampling We build a topic model (Grootendorst, 2020) for each of the five dialects, to detect the different topics in the $train$ dataset. We found that topics are distributed significantly different over $train_{arsar}$ and $train_{\overline{arsar}}$. Therfore, we down-sample the majority class (not sarcastic) and keep only tweets, which their major topic is broadly represented in $train_{\overline{arsar}}$. This way, we ensure that irrelevant topics (assuming that $train_{\overline{arsar}}$ is the more relevant corpus) are removed. Overall, we remove 32% of the $train$ data in the down-sampling process. The majority (92%) of the removed data were selected from the MSA dialect.

Augmentation We use the suggested algorithms by Qiu et al. (2020), which allow usage of transformer models. We utilize the transformer model that is used for the actual classification (see Section 4) for the generation of augmented instances. Since most transformer models are trained using 'Mask-LM' (MLM) and 'Next Sentence Prediction' (NSP) (Devlin et al., 2018) they are capable of augmenting new sentences out of existing ones. We allow the augmentation algorithm to replace text within existing tweets, as well as to fill in "missing" words–injecting new words by the model to existing tweets. An example of the augmentation

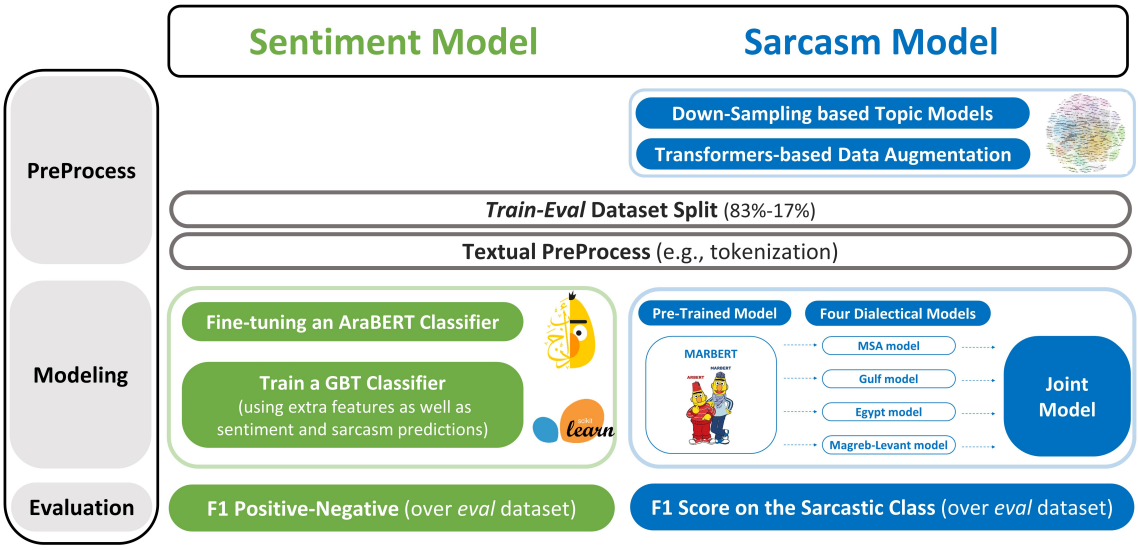

Figure 1: System pipelines. Both algorithms rely on transformer models followed by a tailor-made modeling layer as part of the prediction pipeline.

process over a single tweet is presented in Table 2. Using this novel approach, we reduplicate the corpus (after the down-sampling step).

3 System Description

For both tasks, we use a transformer architecture as a major part of the system. For sentiment classification we use AraBERT (Antoun et al., 2020) while for the sarcasm detection we use MARBERT (Abdul-Mageed et al., 2020). Both are recent pre-trained autoregressive models in Arabic. In both systems, we preprocess the data prior to training. The preprocessing steps include tokenization, removal of redundant text and punctuation marks (e.g., 'RT'), and replacement of links and mentions with placeholders. Figure 1 provides a high-level overview of both systems.

3.1 Sentiment Classifier System

The sentiment classifier uses two-steps modeling. In the first step, we fine-tune an AraBERT (Antoun et al., 2020) transformer model. We do the following adjustments for the particular task we face: (i) Since data are imbalanced (see Table 1), we re-weight the instances according to the inverse proportion of their class weight. (ii) We use the FocalLoss (Lin et al., 2017) as our loss function while training, as it has been shown to work well for imbalanced datasets. (iii) We set the $F1$ score over the positive and negative classes ($F1^{PN}$) as the evaluation criteria of the classifier (tested over the $eval$ dataset) as this is the target evaluation measure of the task. In the second step, we train a Gradient Boosting Trees (GBT) classifier as the final predictor (Friedman, 2002). The GBT model uses the transformer model predictions in addition to some other extra features (e.g., text length, number of emojis, number of hashtags). Another useful feature that we add to the GBT model is the prediction output of the sarcasm classifier (trained over *the same train* dataset as used for the sentiment task). We train the GBT model using 30 iterations (i.e., number of trees) and limit the maximum tree depth to three.

3.2 Sarcasm Detection System

The sarcasm classifier fine-tunes MARBERT (Abdul-Mageed et al., 2020) for each dialect. The training process is composed of two stages of fine-tuning. For every dialect, the first training epoch is performed on the full, multi-dialect training set, and from the second epoch onwards, the training data includes only the relevant examples (referring to the specific dialect). One exception is that we join together the the Levantine and Maghrebi dialects due to their relative small support. Following this way, four distinct classifiers are trained. We set the maximum epoch size to five, batch size to 16 and a relative low learning rate, which ranges ($4e^{-6}$ to $4e^{-7}$).

4 Results

The results of the sentiment classification and the sarcasm detection tasks are reported in Table 3

	Tweet (Arabic)	Tweet (English)
Original	ما اسخف مطاردة الارهاب و هو في مجلس الأمن!	What a ridiculous chase of terrorism in the Security Council!
Replacement	ما اسرع مطاردة الارهاب و العدالة في مجلس الامن!	What a quick chase of terrorism and justice in the Security Council!
Addition	ما اسخف تهمة مطاردة الارهاب للبنان و هو في مجلس الامن!	What a ridiculous charge to chase terrorism for Lebanon while it is in the Security Council!
Both	ما اسخف مطاردة الارهاب والعالم هو يتراس اليوم مجلس لمجلس الامن!	What a ridiculous chase of terrorism and the world now chairs the Security Council!

Table 2: Augmentation example. Replacement and addition of tokens is possible. We use both options in the preprocess steps of the sarcasm model. The augmentation allows us to enlarge the corpus and to add context.

	Acc.	Prec.	Recall	F1	$F1^{PN}$
GigaBERT	0.708	0.687	0.7	0.692	0.674
AraBERT	0.735	0.714	0.729	0.72	0.699
Mod. AraBERT	0.761	0.741	0.762	0.749	0.741
Mod. AraBERT +GBT	**0.77**	**0.757**	**0.763**	**0.758**	**0.75**

Table 3: Sentiment classification results. 'Acc.' and 'Prec.' stand for accuracy and precision respectively. Precision, recall and F1 are calculated using the 'macro' average. $F1^{PN}$ is calculated over the positive and negative classes only. 'Mod'. stands for Modified and it includes the adjustments we applied to the standard AraBERT model.

	Acc.	Prec.	Recall	F1	$F1^{P}$
MARBERT	0.832	0.724	0.771	0.74	0.583
MSA (56%)	0.852	0.747	0.791	0.762	0.61
Egypt (30%)	0.772	0.778	0.787	0.772	0.742
Gulf (7%)	0.771	0.733	0.732	0.731	0.634
Magreb - Levant (7%)	0.8	0.766	0.791	0.775	0.701
All	0.831	0.764	0.80	0.774	0.661

Table 4: Sarcasm detection results. 'Acc.' and 'Prec. stand for accuracy and precision respectively. Precision, recall and $F1^{P}$ are calculated over the positive class only (sarcastic tweets). The MARBERT model (first row) is a standard run using default parameters. Last row are the results over all dialects.

and Table 4, respectively. All reported results are measured on the *eval* dataset. It is important to note that we use a different *eval* dataset for each task (see Section 2). Therefore, it is impossible to compare the results of the two models. Table 3 includes the results of a GigaBERT classifier (Lan et al., 2020) as an additional baseline. Although we did not expand about the experiment done using the GigaBERT model, it is important to note that the model achieves impressive results and almost equivalent to the AraBERT classifier. As observed, in both algorithms we improve the state-of-the-art results (Lan et al., 2020; Abdul-Mageed et al., 2020). In Table 5 we present the confusion matrix of the best model we trained for the sentiment classification problem. As observed, the distinction between 'Negative' and 'Neutral' instances is the vulnerable point of the model.

4.1 Official Submission Results

As explained in the Section 2, the *test* dataset (3000 tweets) true annotations were unknown by the time the predictive models were created. Our two suggested algorithms (as well as other teams' solutions) were evaluated according to the *test* dataset by the deadline time of the shared task[2].

[2]A single submission per team was allowed

		Predicted Classes			
		NEG	NEU	POS	Total
Actual Classes	NEG	667	100	29	796
	NEU	174	697	94	965
	POS	48	53	272	373
	Total	889	850	395	2134

Table 5: Sentiment classification confusion matrix. Results are recorded over the *eval* dataset – 17% of the full *train* dataset.

The results of the sentiment classification and the sarcasm detection tasks over the *test* dataset are reported in Table 6. We were ranked in the 5'th place (out of 22 participation teams) in the sentiment classification task and 9'th place (out of 27 participation teams) in the sarcasm detection task. As observed, there is a significant detraction of both models' performance when evaluated over the *test* set. Based on the to the decisive evaluation metrics ($F1^{PN}$ in the sentiment classification and $F1^{P}$ in the sarcasm detection) there is a performance drop of 4% and 13.6% in both problems respectively.

	Acc.	Prec.	Recall	F1	$F1^*$
Sentiment Classification	0.692	0.643	0.658	0.645	0.719
Sarcasm Detection	0.767	0.706	0.702	0.704	0.568

Table 6: Official submission results. The reported results are over the $test$ dataset (see the third row in Table 1). F^* is the F-Score decisive metric. In the sentiment classification it is the $F1^{PN}$ while in the sarcasm detection it is the F-Score over the sarcastic class. The sentiment classification algorithm is ranked 5'th (out of 22 participants). The sarcasm detection algorithm is ranked 9'th (out of 27 participants).

5 Discussion

One of the challenges we were facing in both tasks is overfitting. We assume that the root cause of the problem is the relative small corpus size and label imbalance. To handle the challenge, We control the learning process in different ways (e.g., small number of training epochs, relative low learning rate). Another challenge we identified and dealt with is the significant difference between $train_{arsar}$ and $train_{\overline{arsar}}$. The difference is reflected through the dialect distribution as well as the label distribution (see Table 1). We assume that the root cause of this difference is either in the way that the data were collected or the way the data were annotated. In both cases, such a "data-mixture" situation raises modeling challenges that we had to handle, mainly in the sarcasm detection algorithm.

6 Conclusion

As observed in previous works as well as in this one, sarcasm detection is a "harder" task to solve than sentiment classification. In this work we introduced two different approaches for solving each of the tasks. Both approaches rely on transformer architectures. However, in both algorithms we adjust the transformer model by using machine-learning techniques (e.g., instance weighting, a more suitable loss function). In addition, in both solutions we add machine-learning components to the pipeline prior or subsequent to the transformer model (e.g., down-sampling, data augmentation). We showed that these techniques improve the existing baselines in both tasks when we evaluate results over the dataset provided as part of the shared task.

References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2020. Arbert & marbert: Deep bidirectional transformers for arabic. *arXiv preprint arXiv:2101.01785*.

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*.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*.

Santosh Kumar Bharti, Bakhtyar Vachha, RK Pradhan, Korra Sathya Babu, and Sanjay Kumar Jena. 2016. Sarcastic sentiment detection in tweets streamed in real time: a big data approach. *Digital Communications and Networks*, 2(3):108–121.

Houda Bouamor, Nizar Habash, Mohammad Salameh, Wajdi Zaghouani, Owen Rambow, Dana Abdulrahim, Ossama Obeid, Salam Khalifa, Fadhl Eryani, Alexander Erdmann, et al. 2018. The madar arabic dialect corpus and lexicon. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4:5477–5488.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ibrahim Abu Farha and Walid Magdy. 2019. Mazajak: An online arabic sentiment analyser. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 192–198.

Ibrahim Abu Farha and Walid Magdy. 2020. From arabic sentiment analysis to sarcasm detection: The arsarcasm dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 32–39.

Jerome H Friedman. 2002. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378.

Maarten Grootendorst. 2020. Bertopic: Leveraging bert and c-tf-idf to create easily interpretable topics.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. 2017. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)*, 50(5):1–22.

Wuwei Lan, Yang Chen, Wei Xu, and Alan Ritter. 2020. Gigabert: Zero-shot transfer learning from english to arabic. In *Proceedings of The 2020 Conference on Empirical Methods on Natural Language Processing (EMNLP)*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. Camel tools: An open source python toolkit for arabic natural language processing. In *Proceedings of the 12th language resources and evaluation conference*, pages 7022–7032.

Siyuan Qiu, Binxia Xu, Jie Zhang, Yafang Wang, Xiaoyu Shen, Gerard de Melo, Chong Long, and Xiaolong Li. 2020. Easyaug: An automatic textual data augmentation platform for classification tasks. In *Companion Proceedings of the Web Conference 2020*, pages 249–252.

Deirdre Wilson. 2006. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253.