Proceedings of the OSACT 2022 Workshop @LREC2022, pages 167–175
Marseille, 20 June 2022

GOF at Arabic Hate Speech 2022: Breaking The Loss Function Convention For Data-Imbalanced Arabic Offensive Text Detection

Aly Mostafa, Omar Mohamed, Ali Ashraf
Department Of Computer Science, Faculty of Computers and Artificial Intelligence, Helwan University
Helwan, Egypt
{alymostafa, omar_20170353, aliashraf }@fci.helwan.edu.eg

Abstract

With the rise of social media platforms, we need to ensure that all users have a secure online experience by eliminating and identifying offensive language and hate speech. Furthermore, detecting such content is challenging, especially in the Arabic language, due to several challenges and limitations. Generally, one of the challenging issues in real-world datasets is long-tailed data distribution. We report our submission to the Offensive Language and hate-speech Detection shared task organized with the 5^{th} Workshop on Open-Source Arabic Corpora and Processing Tools Arabic (OSACT5). In our approach, we focused on how to overcome such a problem by experimenting with alternative loss functions rather than using the traditional weighted cross-entropy loss. Finally, we evaluated various pre-trained deep learning models using the suggested loss functions to determine the optimal model. On the development and test sets, our final model achieved 86.97% and 85.17%, respectively.

Keywords: Arabic, Offensive Language Detection, Class imbalance

1. Introduction

Offensive speech has expanded at an unprecedented rate in today's digital age since most communication has transitioned to digital. Because of the lack of limits on users on social media platforms, the role of detecting offensive speech arises. In general, offensive speech is a public communication that displays hatred or urges violence against a person or group based on characteristics such as race or religion. Detecting offensive speech can be beneficial for filtering out inappropriate content. However, the detection process is difficult for several reasons. First, offensive information is classified into several categories, and not all of them have the same negative impact. Second, most research efforts are directed toward the English language (Hada et al., 2021), (Gupta et al., 2021), (Agrawal and Awekar, 2018), (Davidson et al., 2017); however, offensive detection research in the Arabic language is still in its early stages, with very few notable works (Mubarak and Darwish, 2019), (Mubarak et al., 2017), (Mubarak et al., 2020), (Mubarak et al., 2022). This is due to several challenges, namely a lack of pre-trained models, a small dataset size, and multiple dialects with no dataset that spans all of them; additionally, most social media content is written in Colloquial Arabic, which is not a formal language. It is written in Colloquial Arabic, which differs significantly from Modern Standard Arabic (MSA) since it does not always follow certain grammatical rules and has various word pronunciations. Finally, the Arabic offensive datasets have a long-tailed data distribution (i.e., a few classes account for the majority of the data, while most classes are under-represented), which adds difficulty because most learners will exhibit bias towards the majority class, and in extreme cases, may ignore the minority class entirely. Most offensive/hate speech classification researches ignore this issue since they utilize the traditional/naive technique of assigning sample weights inversely proportionately to the class frequency in the cross-entropy loss. This basic heuristic strategy is commonly used (Huang et al., 2016), (Wang et al., 2017). However, when training deep neural networks on large-scale, real-world, long-tailed datasets, weighted cross-entropy reveals poor performance (Mahajan et al., 2018), (Mikolov et al., 2013). In addition, recent studies (Cao et al., 2019) (Kini et al., 2021) suggest that weighted cross-entropy has little value for balanced accuracy and that alternative strategies based on margin adjustment can be more beneficial, mainly by ensuring that minority classes are further away from the decision boundary. As a result of the aforementioned causes, an important question is raised: How can we address this issue through improved class-balanced loss? In this study, we analyze five distinct loss functions and their variants in the text classification task across three different pre-trained Arabic language models. The experiments revealed that employing the suggested loss functions instead of standard weighted cross-entropy improved the model's macro f1 score metric by 0.5-2.0%. To summarise, our final model was an ensemble learning model composed of three different models (MARBERT, MARBERTV2, and QARiB) (Abdul-Mageed et al., 2021), (Abdelali et al., 2021a), all of which were trained using suggested loss functions and achieved 86.97% and 85.17% on the development and test sets, respectively.

The rest of the paper is organized as follows. section 2 provides a review of previous Arabic offensive text detection literature. section 3 describes the proposed dataset. section 4 proposes the model of the offensive detection. section 5 discusses the results and performance evaluation. Finally, we conclude in section 6.

2. Related Works

This section discusses previous research addressing offensive detection challenges in the Arabic language, the methodologies, strengths, and drawbacks. All of the following studies were conducted on SemEval 2020 Arabic offensive language dataset (Mubarak et al., 2020).

(Husain, 2020) An intensive cleaning strategy was proposed. First, emojis and emoticons are converted to an Arabic textual label that explains their content. Second, they normalise Arabic words with diacritics. Then, stopwords, HTML tags, URLs, mentions, and punctuation marks are removed. Finally, they employ Count Vectorizer as a feature extractor and character-based features to train a Support Vector Machine (SVM)-based classifier. For offensive language detection, they obtained an F1 score of 89.82%.

(Hassan et al., 2020), An ensemble learning model was proposed using four distinct classifiers, two of which are SVM as a classifier and a combination of Mazajak word embedding, character level, and word-level features. The Feed-forward Neural Network (FFNN) was the third classifier, and it used a combination of character-level and word-level features. The final classifiers were Convolutional Neural Network (CNN) and Mazajak as pre-trained word embeddings. As an ensemble learning technique, they used Majority voting and achieved an F1 score of 90.51%.

(Keleg et al., 2020) BERT-generated contextualised word embeddings were used for Arabic offensive detection. Furthermore, a morphological technique for augmentation strategy was used to increase the dataset sample size by employing a list of 87 bad words that were augmented to reach 5497 unique terms. Their strategy yielded an 89.57% of F1 score.

(Saeed et al., 2020) As the stacking classifier, they propose an ensemble of multiple models created from four different Deep Learning models. First and foremost, CNN, Bi-LSTM, Bi-GRU, and CNN-Bi-LSTM are trained as Deep Learning Architectures. Second, they experiment with several word embeddings to see which one may improve the classifier's performance. According to their findings, they combined FastText word embeddings (Bojanowski et al., 2017) with existing Deep Learning architectures. Finally, they use an ensemble stacking approach using five distinct Machine Learning classifiers to obtain the final predictions. Their methodology received an F1 score of 87.37%.

(Haddad et al., 2020) A Deep Learning technique for detecting offensive language is proposed. With the Word2Vec Arabic embedding, they use a bidirectional Gated Recurrent Unit (GRU) with attention layers (AraVec) (Soliman et al., 2017). Also, they employed an oversampling strategy. They added some offensive and inoffensive comments from an already created Arabic dataset derived from YouTube comments (Alakrot et al., 2018). Their strategy received an F1 score of 85.90%.

(Abdellatif and Elgammal, 2020) They proposed ULMFiT (Howard and Ruder, 2018) pre-trained from scratch on the Arabic Wikipedia corpus, then they fine-tuned their model on the Arabic offensive dataset achieving a 77.83% F1 score.

(Djandji et al., 2020) Due to the limited sample size in both tasks, they presented a multi-task learning strategy to train jointly offensive and hate speech detection. Their multi-task learning architecture goes as follows: They apply data pre-processing methods to the offensive tweet input before using the AraBERT model as a shared layer between the two tasks. Finally, for each task, two dense layers are used as task-specific layers. Their architecture achieved an f1 score of 90.04%.

(Elmadany et al., 2020) They used the BERT Multilingual model to leverage an effective offensive detection method. In addition, they use an oversampling approach to obtain negative sentiment tweets and label them as offensive or hate speech based on a lexical seed. Their method received 77.38% of the f1 score.

(Farha and Magdy, 2020) They presented a CNN-BiLSTM-based multi-task learning architecture. Their architecture is as follows: First, they used pre-trained skip-gram word2vec embedding on a corpus of 250 million tweets to embed the input tweets. Second, they pass the embedding to the CNN layer and performed Max Pooling. Third, pass the feature vectors to the BiLSTM layer; all previous stages are considered shared layers. Finally, three dense layers are employed as task-specific layers, one for offensive speech detection, one for hate speech identification, and one for the sentiment. The reason for incorporating sentiment in offensive and hate speech detection is that sentiment may include additional information for the model since offensive language or hate speech are often sentimental and express negative emotion towards the target. Their model achieved 87.87% of the f1 score.

According to the findings of this survey, most studies did not make further research to address the problem of data imbalance; instead, they relied on the standard weighted cross-entropy. However, only two of them addressed the issue in data-level methods with oversampling techniques by augmenting the dataset to the positive class (offensive class). The purpose of this research is to overcome previous limitations by using and assessing various loss functions to better address the problem of data imbalance.

3. Dataset

The proposed dataset (Mubarak et al., 2022) for the OSACT-2022 Shared challenge comprises 13k Arabic tweets collected using a set of emojis with a high malicious effect independent of the tweet text. As a result, they chose tweets that had one or more emojis. The data is classified into three categories: offensive, hate speech, and fine-grained hate speech, and divided into 70% for training, 10% for development, and 20% for testing. However, the dataset's distribution is severely imbalanced and skewed, with 35% being offensive and 11% being hate speech. Offensive tweets and violent account for 1.5% and 0.7% of the total corpus, respectively. To address this challenge, we used the proposed approach of experimenting with different loss functions rather than simply adopting standard weighted loss cross-entropy.

4. Methodology

In this section, we will go over the key components of the proposed method, starting with the data preprocessing techniques used, then a discussion of the suggested loss functions, followed by an overview of the pre-trained models used, finally the ensemble learning approach of the three pre-trained models is presented.

4.1. Data Pre-Processing

We eliminated non-Arabic letters, punctuation marks, digits, and Arabic diacritics during the pre-processing. Following the removal of unnecessary characters, the text is normalized into its unified form. Because social media material is written in unconventional ways and is not a formal language, some users choose to repeat the same word characters to emphasize its meaning, such as ”جووووول” instead of ”جول” which means GOAL. We addressed this problem by eliminating duplicate letters from each word (elongation removal) (Hegazi et al., 2021). We did not remove emojis or emotions as they can significantly aid the tweet classification decision. The final step is to replace selected terms with meaningful tokens in order to unify them throughout the dataset, as seen below:

- Replace URL with ”رابط”
- Replace mentions @USER with ”مستخدم”
- Replace Email with ”بريد”

4.2. Loss Functions

In this subsection, we discuss five different loss functions with their variations as follows :

- Weighted Cross-Entropy loss(CE)
- Weighted CE combined with label smoothing
- Focal Loss
- Focal Loss combined with label smoothing
- Dice Loss
- Tversky Loss
- Focal Tversky Loss
- Vector Scaling(VS) Loss
- VSLoss combined with label smoothing
- Weighted VSLoss
- Weighted VSLoss combined with label smoothing

4.2.1. Weighted Cross-Entropy Loss

Standard Cross-Entropy loss: is calculated as follows:

$$\text{CE} = -\frac{1}{N}\sum_{i}\sum_{j\in\{0,1\}} y_{ij}\log p_{ij} \tag{1}$$

As shown in Eq.1, each x_i contributes equally to the overall objective. The standard technique for dealing with the case when we don't want all x_i to be regarded equally is to provide various weighting factors to distinct classes. Eq.1 is modified as follows for the former:

$$\text{Weighted CE} = -\frac{1}{N}\sum_{i}\alpha_i\sum_{j\in\{0,1\}} y_{ij}\log p_{ij} \tag{2}$$

where $\alpha_i \in [0,1]$ may be set by assigning sample weights inversely proportionately to the class frequency. Empirically, these methods are extensively used as the training objective for data-imbalanced NLP problems(Lample et al., 2016), (Meng et al., 2019), (Devlin et al., 2018), (Yu et al., 2018), (McCann et al., 2018), (Ma and Hovy, 2016), (Chen et al., 2017).

Weighted Cross-Entropy loss + Label Smoothing: The use of a smoothing parameter $\epsilon \in [0,1]$ is the only difference between standard weighted cross-entropy and weighted cross-entropy paired with label smoothing (Szegedy et al., 2016), (Müller et al., 2019). Label smoothing is a regularization technique that solves the overconfidence and overfitting issues. The cross-entropy with label smoothing is calculated as follows:

$$H(y_{i,j}, p_{i,j}) = (1-\epsilon)H(y,p) + \epsilon H(y,p) \tag{3}$$

4.2.2. Focal Loss

Standard Focal Loss: (Lin et al., 2017) it is a dynamically scaled cross-entropy loss created by adding a modulating term to the cross-entropy loss so that the scaling factor decays to zero as confidence in the correct class increases (easily classified examples) and increases on low confidence cases (hard misclassified, examples). This procedure is used to quickly focus the learning process on difficult examples. The modulating factor $(1-p_t)^\gamma$ is added to the cross-entropy loss. Configure $\gamma > 0$ lowers the relative loss for cases that have been correctly classified $p_t > .5$, emphasising challenging, misclassified cases. There is a focusing parameter that may be adjusted here $\gamma \geq 0$. The equation of Focal loss as follows:

$$\text{FL}(p_t) = -(1-p_t)^\gamma \log(p_t) \tag{4}$$

Focal loss + Label Smoothing: The Focal loss is edited to add label smoothing parameter, as stated before the label smoothing aids to tackle the problem of overconfidence.

Model	Weighted CE		Weighted CE + LS[‡]		Focal Loss		Focal Loss+LS[‡]		Dice Loss		Tversky Loss		Focal Tversky Loss		VS Loss	
	$\mathbf{F_1}$	PR - RR	$\mathbf{F_1}$	PR - RR	$\mathbf{F_1}$	PR - RR	$\mathbf{F_1}$	PR - RR	$\mathbf{F_1}$	PR - RR	$\mathbf{F_1}$	PR - RR	$\mathbf{F_1}$	PR - RR	$\mathbf{F_1}$	PR - RR
MARBERT	83.7	83.2 - 84.2	83.6	83.3 - 84.1	84.5	84.4 - 84.6	84.4	84.3 - 84.4	83.2	82.8 - 83.5	84.1	84.7 - 83.6	83.7	84.6 - 83.0	**85.6**	**85.8 - 85.4**
MARBERT(v2)	84.5	83.6 - 85.9	84.7	83.8 - 86.1	84.9	84.8 - 85.1	**85.2**	**85.1 - 85.2**	84.3	84.0 - 84.8	83.7	83.1 - 84.5	83.7	83.0 - 84.6	84.6	84.3 - 85.0
QARiB	85.4	84.7 - 86.3	85.2	84.6 - 86.1	85.4	85.2 - 85.6	85.4	85.2 - 85.6	**85.7**	**85.5 - 85.8**	84.9	84.8 - 85.1	85.0	85.7 - 84.3	83.6	84.8 - 82.6

Table 1: Comparsion Between Different Loss Functions. [‡] Refer to Label Smoothing, PR for Precision rate, and RR for Recall rate.

4.2.3. Dice Loss

The Sørensen–Dice coefficient (Dice, 1945), (Sorensen, 1948), often known as the dice coefficient (DSC), is a harmonic mean of precision and recall thus weighs false positives (FPs) and false negatives (FNs) equally. Furthermore, (Milletari et al., 2016) proposed to convert the denominator to the square form for faster convergence, which results in the dice loss (DL) shown below:

$$\mathrm{DL} = 1 - \frac{2\sum_i p_{i1} y_{i1} + \gamma}{\sum_i p_{i1}^2 + \sum_i y_{i1}^2 + \gamma} \tag{5}$$

In our context, p is the set of all positive cases predicted by a certain model, and y is the set of all golden positive examples in the dataset. When applied to boolean data, the definitions of true positive (TP), false positive (FP), and false-negative (FN) are used (FN). It is usual to apply a γ factor to both the nominator and the denominator for smoothing reasons.

4.2.4. Tversky Loss

The Tversky index (Tversky, 1977), (Hashemi et al., 2018) is a broadening of the Dice similarity coefficient and F_β scores. The Tversky index is defined as where and govern the level of penalties for FPs and FNs. One of the Dice loss function's shortcomings is that it equally weights false positive (FP) and false negative (FN) detections. To improve the recall rate, FN detections should be weighted higher than FPs. The following formulation is used to define the Tversky loss function:

$$T(\alpha, \beta) = \frac{\sum_{i=1}^{N} p_{0i} g_{0i}}{\sum_{i=1}^{N} p_{0i} g_{0i} + \alpha \sum_{i=1}^{N} p_{0i} g_{1i} + \beta \sum_{i=1}^{N} p_{1i} g_{0i}} \tag{6}$$

4.2.5. Focal Tversky Loss

The focal Tversky loss function (FTL) is a combination of the regular Tversky loss and focal loss (modulating term). FTL is parametrized by γ, for control between easy and hard training examples. In (Lin et al., 2017), the focal parameter exponentiates the cross-entropy loss to focus on hard classes detected with a lower probability. The focal Tversky Loss (FTL) function is defined as follows:

$$FTL = (T)^{1/\gamma} \tag{7}$$

4.2.6. Vector Scaling Loss

Vector-scaling (Vs) loss (Kini et al., 2021) is an improved form of cross-entropy with three additional parameters that integrate additive and multiplicative logit modifications, which had previously been proposed in the literature but in isolation. The following is the **binary VS-loss** for labels $y \in \{\pm 1\}$, weight parameters $\omega_\pm > 0$, additive logit parameters $\iota_\pm \in \mathbb{R}$, and multiplicative logit parameters $\Delta_\pm > 0$:

$$\ell_{\mathrm{VS}}(y, f_w(x)) = \omega_y \cdot \log\left(1 + e^{\iota_y} \cdot e^{-\Delta_y y f_w(x)}\right) \tag{8}$$

The VS-loss for imbalanced datasets with C $>$ 2 classes is as follows:

$$\ell_{\mathrm{VS}}(y, f_w(x)) = -\omega_y \log\left(e^{\Delta_y f_y(x) + \iota_y} / \textstyle\sum_{c \in [C]} e^{\Delta_c f_c(x) + \iota_c}\right) \tag{9}$$

Here $f_w : R^d \to R^C$ and $f_w(x) = [f_1(x), \ldots, f_C(x)]$ is the vector of logits. The various modifications include adding a label smoothing parameter, applying sample weights inversely related to the class frequency, and combining the previous settings together.

4.3. Pre-Trained Models

Because the dataset is a collection of tweets, selecting pre-trained models that have been trained on Twitter data with diverse dialects was critical. Following the literature (Abdul-Mageed et al., 2021), utilising models pre-trained on social media data (e.g., Twitter data) improves finetuning performance over training on standard data (e.g., Wikipedia) if the finetuning procedure is done on a dataset that is mostly composed of tweets. The details about the employed models are described below.

QARiB: (Abdelali et al., 2021b) QCRI Arabic and Dialectal BERT model was trained on 420 million tweets and 180 million sentences of text comprised of 14B tokens. The data for the tweets was gathered using the Twitter API. The text data was derived from a mix of Arabic GigaWord, Abulkhair Arabic Corpus (El-Khair, 2016), and OPUS (Lison and Tiedemann, 2016). The model is a bidirectional transformer encoder model (BERT) (Devlin et al., 2018) with 110M parameters that contains 12 encoder layers, 12 attention heads, and 768 hidden sizes. The QARiB model trained with Dice-Loss (Dice, 1945), (Li et al., 2019) achieved 85.717% on F1-score.

MARBERT & MARBERTv2: MARBERT (Abdul-Mageed et al., 2021) was trained on 1 billion Arabic tweets by randomly picking tweets from a huge in-house dataset of around 6 billion tweets made up of 15.6 billion tokens and with a sequence length of just 128. The model was trained using the same network architecture as BERT Base (masked language model) but without the next sentence prediction (NSP) component. Because the model has been pre-trained on a variety of tweets, it can recognize a variety of dialects, not only Modern Standard Arabic (MSA). The model incorporates 163M parameters, including 12 encoder layers, 12 attention heads, and 768 hidden sizes. Because the model was trained with a sequence length of just 128, it is inadequate for Question Answering. As a result, they re-train the model using a different collection of MSA resources, including Books (Hindawi), El-Khair (El-Khair, 2016), Gigaword, OSCAR (Suárez et al., 2019), OSIAN (Zeroual et al., 2019), and AraNews dataset (Nagoudi et al., 2020) with a longer sequence length of 512 tokens totaling 29B tokens. The MARBERT model trained using Focal loss + Label smoothing achieved an f1 score of 85.66%, while the MARBERTV2 model trained with VSLoss obtained an f1 score of 85.21%.

4.4. Ensemble Learning Model

We employed the ensemble learning approach to enhance and improve model performance. We noticed that the three models generate different mistakes on different samples; so, we used Ensemble learning approaches since the ensemble's ability to correct the errors of some of its members is entirely dependent on the diversity of the classifiers that comprise the ensemble. Our final ensemble model is based on a majority voting technique between the following models: 1). QARiB trained with Dice loss. 2). MARBERT trained with VS loss. 3). MARBERTV2 trained with Focal loss + label smoothing.

5. Results and Discussion

5.1. Performance Metrics

We employed different metrics to assess the model performance and to understand/analyse its efficiency and errors. We calculated Precision, recall, and F1-score in the macro setting. We used Macro F1-score instead of Accuracy to assess the model's performance since the proposed dataset is highly imbalanced, making the Accuracy unsuitable for this task.

5.2. Experimental Results

In this section, we present the results of experimenting with various models and architectures trained with different loss functions and the impact on performance.

Different Models: We evaluated many pre-trained models to determine the most effective one for achieving the best results individually or as part of an ensemble group. The majority of the pre-trained models that were fine-tuned on the proposed data generated outcomes that were comparable to each other. Among all models tested, Light Gradient Boosting Machine (LGBM) trained on QARiB embeddings fine-tuned on the proposed data yielded the best results individually. In addition, we eliminated the emojis and emotions from the proposed dataset and trained the MARBERT model, however, the results were not competitive. Furthermore, we tested two versions of AraBERT, a base and a large version trained on Twitter data, and they achieved 85.54% and 85.15% on the F1-score measure, respectively. Finally, we tried a different model combination in an ensemble approach; the first experiment consisted of Arabert-base-Twitter, MARBERT, and QARiB and obtained 86.73% on the F1-score. The second was a combination of MARBERTV2, MARBERT, and QARiB that resulted in an f1-score of 87.04%. The final experiment obtained an f1-score of 86.43% by combining LightGBM trained on QARiB embeddings, MARBERT, and MARBERTV2. The experiments are shown in Table 2.

Different Loss Functions: According to the no-free lunch, theory (Wolpert and Macready, 1997), there is no optimum solution for all problems. Furthermore, after experimenting with various loss functions on the selected models, we observed that some loss functions perform better for some models but not others. However, under the f1-score metric, most of the suggested loss functions exceeded the standard weighted cross-entropy. The comparison between different loss functions and models is presented in Table 1.

Models	Macro-F1(%)
MARBERT(Without emojis)	85.077
AraBERT-Large-Twitter	85.158
QARiB	85.424
AraBERT-Base-Twitter	85.548
MARBERT	85.574
MARBERTV2	85.723
LightGBM(QARiB Embeddings)	**85.798**
Ensemble(LightGBM+ MARBERT+MARBERTV2)	86.432
Ensemble(AraBERT-B-T+ MARBERT+QARiB)	86.733
Ensemble(MARBERTV2+ MARBERT+QARiB)	**87.044**

Table 2: Different Models With an F1-score On Development Set.

5.3. Discussion

Results show in Table 1 that Weighted CE was not the best performer compared to the rest of the loss functions. Its best result was on the QARiB model, yielding an F1 score of 84.4%, higher than MARBERT-v2's result of 84.5% and MARBERT's of 83.7%. Even with the addition of Label Smoothing, Weighted CE still failed to outperform the rest of the loss functions while not making any significant difference from the standard Weighted CE. For MARBERT, results were quite similar between the loss functions, namely Weighted CE, Weighted CE with Label Smoothing and Focal Tversky

True Label	Predicted Label	Attribution Score	Word Importance
OFF	+1 (0.79)	0.22	[CLS] يلعنهم الله [UNK] [SEP]
OFF	+1 (0.71)	0.32	[CLS] بعينك شيطان 😡 [SEP]
OFF	+1 (0.81)	0.87	[CLS] انبح النار كلب يا 😂 من اسمك لقاح اتراك 🟩 [UNK] [SEP]
NOT OFF	-1 (0.20)	-1.90	[CLS] 💪 اصرار بحياتك لازم يكون عندك ناجح عشان تكون [SEP]
NOT OFF	-1 (0.20)	-0.51	[CLS] زين اعلانات كانه من زينه ▪ 😷 [SEP]
NOT OFF	-1 (0.21)	-2.22	[CLS] ياجماعه الله ماشاء قولوا [SEP]

Table 3: Word Attributions In Dataset's Tweets (🟥 Not Offensvie -1 , 🟩 Offensive +1)

loss, yielding an F1 score of 83.7%, 83.6%, and 83.7%, respectively, while Focal loss and Focal loss with Label Smoothing had close results with an F1 score of 84.5% and 84.4%, respectively. MARBERT's best performer was VS loss, yielding an F1 score of 85.6. For MARBERT-v2, results were also quite similar between the loss functions, namely Tversky loss and Focal Tversky loss, both yielding an F1 score of 83.7%, while Weighted CE, Weighted CE with label Smoothing, Focal loss, Dice loss, and VS loss had close results with an F1 score of 84.5%, 84.7%, 84.9%, 84.3%, and 84.6%, respectively. MARBERT-v2's best performer was Focal loss with Label Smoothing, yielding an F1 score of 85.2%. For QARiB, results were close between the loss functions, but with significant improvements compared to the other models. Weighted CE, Weighted CE with Label Smoothing, Focal loss, and Focal loss with Label Smoothing yielded an F1 score of 85.4, 85.2%, 85.4%, and 85.4%, respectively. MARBERT's best performer was Dice loss, yielding an F1 score of 85.7%.

5.4. Model Interpretability

We used Captum (Kokhlikyan et al., 2020), a model interpretability and understanding library for PyTorch (Paszke et al., 2019), to interpret the final model decision or predicted class. It allows researchers and developers to efficiently understand which features are contributing to the model's outputs using tools such as integrated gradients, smooth-grad, and others. Furthermore, Captum solves the lack of transparency in deep learning models, or as their called, Black Boxes. This term refers to how difficult it is to understand and explain the behaviour of a model. Captum looks at a single prediction and identifies features leading to that prediction through Integrated Gradients. In our case, the features are the words or emojis in the tweet that led the model to a specific predicted outcome. Green indicates that the tokens are pulling towards offensiveness, while red indicates that they are pulling toward inoffensiveness. The colour intensity represents the magnitude of the signal. Table 3 illustrates the word attributions for selected examples of the proposed dataset using Captum.

6. Conclusion and Future Works

In this work, we proposed a method for dealing with Arabic Offensive text detection. Our final model is a Deep Learning ensemble learning system consisting of three different Deep Learning models. Furthermore, because the dataset distribution is highly skewed, testing with alternative loss functions to observe how they affect model performance revealed that simply replacing the standard weighted cross-entropy with different loss functions enhanced the model's Macro F1-score by 0.5-2%. On the development set, the proposed pipeline achieved 87.04%, while on the test set, it obtained 85.17%. In future work, we aim to test the effectiveness of those loss functions on a wide range of tasks in the Arabic language. This result would also support our findings that standard cross-entropy loss is ineffective for long-tailed data distribution. Because most real-world datasets in various tasks are highly imbalanced, such a study would assist the researcher in better addressing the problem of highly imbalanced datasets.

7. References

Abdelali, A., Hassan, S., Mubarak, H., Darwish, K., and Samih, Y. (2021a). Pre-training bert on arabic tweets: Practical considerations.

Abdelali, A., Hassan, S., Mubarak, H., Darwish, K., and Samih, Y. (2021b). Pre-training bert on arabic tweets: Practical considerations. *arXiv preprint arXiv:2102.10684*.

Abdellatif, M. and Elgammal, A. (2020). Offensive language detection in arabic using ulmfit. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 82–85.

Abdul-Mageed, M., Elmadany, A., and Nagoudi, E. M. B. (2021). 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)*, pages 7088–7105, Online, August. Association for Computational Linguistics.

Agrawal, S. and Awekar, A. (2018). Deep learning for detecting cyberbullying across multiple social media platforms. In *European conference on information retrieval*, pages 141–153. Springer.

Alakrot, A., Murray, L., and Nikolov, N. S. (2018). Dataset construction for the detection of anti-social behaviour in online communication in arabic. *Procedia Computer Science*, 142:174–181.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Cao, K., Wei, C., Gaidon, A., Arechiga, N., and Ma, T. (2019). Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32.

Chen, D., Fisch, A., Weston, J., and Bordes, A. (2017). Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11, pages 512–515.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

Djandji, M., Baly, F., Antoun, W., and Hajj, H. (2020). Multi-task learning using arabert for offensive language detection. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 97–101.

El-Khair, I. A. (2016). 1.5 billion words arabic corpus. *arXiv preprint arXiv:1611.04033*.

Elmadany, A., Zhang, C., Abdul-Mageed, M., and Hashemi, A. (2020). Leveraging affective bidirectional transformers for offensive language detection. *arXiv preprint arXiv:2006.01266*.

Farha, I. A. and Magdy, W. (2020). Multitask learning for arabic offensive language and hate-speech detection. In *Proceedings of the 4th workshop on open-source Arabic corpora and processing tools, with a shared task on offensive language detection*, pages 86–90.

Gupta, A., Pal, A., Khurana, B., Tyagi, L., and Modi, A. (2021). Humor@IITK at SemEval-2021 task 7: Large language models for quantifying humor and offensiveness. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 290–296, Online, August. Association for Computational Linguistics.

Hada, R., Sudhir, S., Mishra, P., Yannakoudakis, H., Mohammad, S. M., and Shutova, E. (2021). Ruddit: Norms of offensiveness for English Reddit comments. 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)*, pages 2700–2717, Online, August. Association for Computational Linguistics.

Haddad, B., Orabe, Z., Al-Abood, A., and Ghneim, N. (2020). Arabic offensive language detection with attention-based deep neural networks. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 76–81.

Hashemi, S. R., Salehi, S. S. M., Erdogmus, D., Prabhu, S. P., Warfield, S. K., and Gholipour, A. (2018). Tversky as a loss function for highly unbalanced image segmentation using 3d fully convolutional deep networks. *arXiv preprint arXiv:1803.11078*.

Hassan, S., Samih, Y., Mubarak, H., Abdelali, A., Rashed, A., and Chowdhury, S. A. (2020). Alt submission for osact shared task on offensive language detection. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 61–65.

Hegazi, M. O., Al-Dossari, Y., Al-Yahy, A., Al-Sumari, A., and Hilal, A. (2021). Preprocessing arabic text on social media. *Heliyon*, 7(2):e06191.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Huang, C., Li, Y., Loy, C. C., and Tang, X. (2016). Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5375–5384.

Husain, F. (2020). Osact4 shared task on offensive language detection: Intensive preprocessing-based approach. *arXiv preprint arXiv:2005.07297*.

Keleg, A., El-Beltagy, S. R., and Khalil, M. (2020). Asu_opto at osact4-offensive language detection for arabic text. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 66–70.

Kini, G. R., Paraskevas, O., Oymak, S., and Thrampoulidis, C. (2021). Label-imbalanced and group-sensitive classification under overparameterization. *Advances in Neural Information Processing Systems*, 34.

Kokhlikyan, N., Miglani, V., Martin, M., Wang, E., Alsallakh, B., Reynolds, J., Melnikov, A., Kliushkina, N., Araya, C., Yan, S., et al. (2020). Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Li, X., Sun, X., Meng, Y., Liang, J., Wu, F., and Li, J. (2019). Dice loss for data-imbalanced nlp tasks. *arXiv preprint arXiv:1911.02855*.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Lison, P. and Tiedemann, J. (2016). OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia, May. European Language Resources Association (ELRA).

Ma, X. and Hovy, E. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., and Van Der Maaten, L. (2018). Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196.

McCann, B., Keskar, N. S., Xiong, C., and Socher, R. (2018). The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Meng, Y., Wu, W., Wang, F., Li, X., Nie, P., Yin, F., Li, M., Han, Q., Sun, X., and Li, J. (2019). Glyce: Glyph-vectors for chinese character representations. *Advances in Neural Information Processing Systems*, 32.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Milletari, F., Navab, N., and Ahmadi, S.-A. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE.

Mubarak, H. and Darwish, K. (2019). Arabic offensive language classification on twitter. In *International Conference on Social Informatics*, pages 269–276. Springer.

Mubarak, H., Darwish, K., and Magdy, W. (2017). Abusive language detection on Arabic social media. In *Proceedings of the First Workshop on Abusive Language Online*, pages 52–56, Vancouver, BC, Canada, August. Association for Computational Linguistics.

Mubarak, H., Darwish, K., Magdy, W., Elsayed, T., and Al-Khalifa, H. (2020). Overview of osact4 arabic offensive language detection shared task. In *Proceedings of the 4th Workshop on open-source arabic corpora and processing tools, with a shared task on offensive language detection*, pages 48–52.

Mubarak, H., Hassan, S., and Chowdhury, S. A. (2022). Emojis as anchors to detect arabic offensive language and hate speech. *arXiv preprint arXiv:2201.06723*.

Müller, R., Kornblith, S., and Hinton, G. E. (2019). When does label smoothing help? *Advances in neural information processing systems*, 32.

Nagoudi, E. M. B., Elmadany, A., Abdul-Mageed, M., and Alhindi, T. (2020). Machine generation and detection of Arabic manipulated and fake news. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 69–84, Barcelona, Spain (Online), December. Association for Computational Linguistics.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, et al., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Saeed, H. H., Calders, T., and Kamiran, F. (2020). Osact4 shared tasks: Ensembled stacked classification for offensive and hate speech in arabic tweets. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 71–75.

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

Sorensen, T. A. (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons. *Biol. Skar.*, 5:1–34.

Suárez, P. J. O., Sagot, B., and Romary, L. (2019). Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*. Leibniz-Institut für Deutsche Sprache.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Tversky, A. (1977). Features of similarity. *Psychological review*, 84(4):327.

Wang, Y.-X., Ramanan, D., and Hebert, M. (2017). Learning to model the tail. *Advances in Neural Information Processing Systems*, 30.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Yu, A. W., Dohan, D., Luong, M.-T., Zhao, R., Chen, K., Norouzi, M., and Le, Q. V. (2018). Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*.

Zeroual, I., Goldhahn, D., Eckart, T., and Lakhouaja, A. (2019). OSIAN: Open source international Arabic news corpus - preparation and integration into the CLARIN-infrastructure. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 175–182, Florence, Italy, August. Association for Computational Linguistics.