IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 28 September 2023, accepted 24 October 2023, date of publication 6 November 2023,
date of current version 9 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3330081

RESEARCH ARTICLE

A Deep Learning-Based Framework for Offensive Text Detection in Unstructured Data for Heterogeneous Social Media

JAMSHID BACHA[1,2], FARMAN ULLAH[3], JEBRAN KHAN[4], ABDUL WASAY SARDAR[2,5], AND SUNGCHANG LEE[2,6], (Member, IEEE)

[1]School of Electrical Engineering and Computer Science, Technische Universität Berlin, 10623 Berlin, Germany
[2]School of Electronics and Information Engineering, Korea Aerospace University, Goyang-si 10540, South Korea
[3]College of Information Technology, United Arab Emirates University (UAEU), Abu Dhabi, United Arab Emirates
[4]Department of Artificial Intelligence, Ajou University, Suwon, Gyeonggi 16499, South Korea
[5]Natural Computing Research and Applications Group, Smurfit School of Business, University College of Dublin, Dublin 4, D04 V1W8 Ireland
[6]Thingswell Inc., Dongan-gu, Anyang-si, Gyeonggi-do 14056, South Korea

Corresponding authors: Sungchang Lee (sclee712@gmail.com) and Farman Ullah (farman@uaeu.ac.ae)

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government through the Ministry of Science and ICT under Grant NRF-2022R1F1A1074652.

ABSTRACT Social media such as Facebook, Instagram, and Twitter are powerful and essential platforms where people express and share their ideas, knowledge, talents, and abilities with others. Users on social media also share harmful content, such as targeting gender, religion, race, and trolling. These posts may be in the form of tweets, videos, images, and memes. A meme is one of the mediums on social media which has an image and embedded text in it. These memes convey various views, including fun or offensiveness, that may be a personal attack, hate speech, or racial abuse. Such posts need to be filtered out immediately from social media. This paper presents a framework that detects offensive text in memes and prevents such nuisance from being posted on social media, using the collected KAU-Memes dataset 2582. The latter combines the ''2016 U.S. Election'' dataset with the newly generated memes from a series of offensive and non-offensive tweets datasets. In fact, this model uses the KAU-Memes dataset containing symbolic images and the corresponding text to validate the proposed model. We compare the performance of three proposed deep-learning algorithms to train and detect offensive text in memes. To the best of the authors knowledge and literature review, this is the first approach based on You Only Look Once (YOLO) for offensive text detection in memes. This framework uses YOLOv4, YOLOv5, and SSD MobileNetV2 to compare the model's performance on the newly labeled KAU-Memes dataset. The results show that the proposed model achieved 81.74%, 84.1%, mAP, and F1-score, respectively, for SSD-MobileNet V2, and 85.20%, 84.0%, mAP, and F1-score, respectively for YOLOv4. YOLOv5 had the best performance and achieved the highest possible mAP, F1-score, precision, and recall which are 88.50%, 88.8%, 90.2%, and 87.5%, respectively, for YOLOv5.

INDEX TERMS Cyberbully, unstructured data, deep learning, YOLO, social media, offensive, MobileNet-SSD, image processing.

I. INTRODUCTION

The meme spreads via different social media platforms and shows some fun or targets something humorously or offensively. Memes on social media can be in any form, posted via images, videos, or tweets, likely to have a significant impact on communication [1], [2]. However, the main form of memes on social media are images that include some text and a background image having multi-model nature and causing confusion in understanding the contents of the

The associate editor coordinating the review of this manuscript and approving it for publication was Maria Chiara Caschera.

image [3]. On social media, hate speech is one of the common content [4]. This is one of the important reasons to understand the meaning and intention of memes and identify whether they can be offensive or non-offensive. Memes can spread hatred in society via social media: a legitimate concern justifying the need to filter such content automatically and immediately.

A meme can be a racial, religious, personal attack, or maybe an attack on the community. personal attack, or maybe an attack on the community. The literature revealed several interesting works on memes: emotion analysis in [5], sarcastic meme detection in [6], and hateful meme detection in [7]. Where they discussed the multi-model nature of memes which makes them very difficult to understand and classify. This is also difficult for a machine learning model to classify whether a meme can be offensive or non-offensive. The reason is that memes depend on the context and focus on the image and text. Without relevant knowledge of the context in which the meme was created, it is rather risky to speculate on whether the meme is offensive or not. Similarly, it is hard for a standard OCR to extract and detach texts from the meme, because memes can be noisy. Another critical factor is that since the text in the meme is overlaid on top of the image, the text needs to be extracted using OCR, which can result in errors that require additional manual post-editing [8].

The deeper meaning of memes can be funny for one; but can be offensive for another. These memes are usually spread on social media such as Facebook, Instagram, Twitter, and Pinterest. However, some people use it to target a person, a specific religion, or an entire community. These memes can elicit depressive behaviors and should be filtered out from social media. Even some political campaign managers have already turned to memes on social media in their quest to directly or indirectly influence election results: because people can see those memes and accept the idea they promote. Many researchers are trying to solve this problem by identifying offensive memes, but millions of memes on social media are hard to remove manually. According to,[1] an average of 95 million images are uploaded daily. On Twitter, for instance, there is nearly 40% post that has visual contents.[2] Also, the tweets with images can get 150% higher retweets than the tweet which don't have images.[3]

There are multiple approaches for memes classification, like OCR technique [9] extracts the text from the images. However, using the OCR text extraction can extract all the text from the images, like watermarks, implicit and explicit entities which can lead us to the incorrect classification of the memes. The meme's typo graphic text extraction using optical character recognizer OCR is explained in [6] for sarcasm detection in memes.

Offensive memes can be dangerous and insult people [10]. A meme can be aggressive [11], troll [12], and cyberbullying [13]. Figure 1 shows examples of offensive and non-offensive memes. Where Figure 1 (a) and (b) are the memes, there is no offensive text that makes the meme to be offensive. While on the other hand, Figure 1 (c), (d), and (e) are images where some offensive text is used and makes the memes offensive. There are a lot of images that include offensive text in images. The text associated with images can make clear that the meme is offensive or non-offensive. That's why this framework focuses on the text and detecting offensive content in unstructured data.

Therefore to address such problems and overcome the error rate, this proposed approach is based on YOLO to detect the offensive text inside the memes on social media. Accordingly, this paper proposed a new dataset with the addition of an existing dataset on the 2016 U.S. Election and the offensive and non-offensive tweets dataset from [14].

The contributions to this paper are as follows:

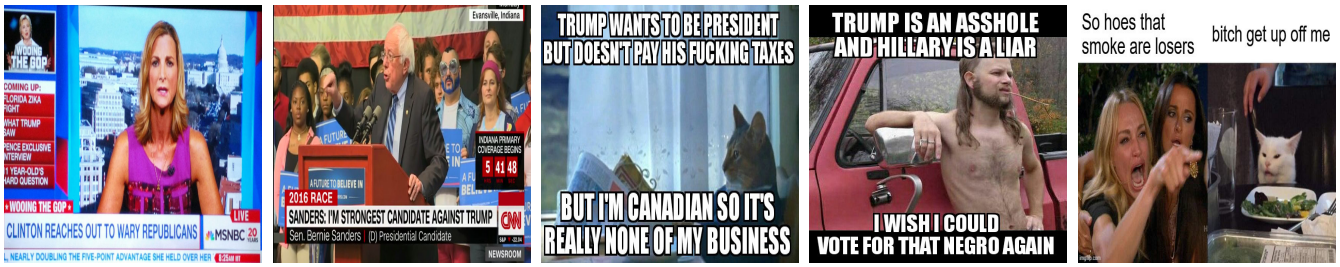
1) A new framework based on the computer vision model is presented in this study for the detection of offensive content in unstructured data.
2) This paper studies text detection in unstructured data and formulated two kinds of text detection, i.e., offensive and non-offensive.
3) Generated a new KAU-Memes dataset consists of 2582 memes and is labeled for YOLO and SSD-MobileNet algorithms individually.
4) This paper presented a performance comparison of YOLOv4, YOLOv5, and SSD MobileNet-V2 algorithms based on training, detection time, mAP, F1-score, precision vs recall curve, and confusion matrix.
5) Extensive experiments on 2016.US.Election and the KAU-Memes dataset prove that algorithms performance improves with a high number of memes.

The paper is organized as follows: Related work to offensive and hateful memes is discussed in Section II. The proposed model is described in Section III. Results and discussion of the model are explained in Section IV. Section V discussed the conclusion of the proposed model and future work plan.

II. LITERATURE REVIEW

There are different approaches used for offensive, cyberbullying, toxic comments, and hateful speech classification and detection. Bad behavior became a big issue on social media platforms [15]. On social media, there are different rumors [16], hateful content [17], and cyberbullying [18] contents people share. However, memes perform a big role in such kind of situations on social media. Some approaches have been proposed to overcome these problems of hate speech and offensive content. Such as the troll memes classification has been developed based on pre-trained models i.e. EffNet, VGG16, and Resnet [19]. Two models are proposed by [20] among them one works as a text features extraction and the second is on image-based features extraction while sending the memes to the transformer

[1]https://www.wired.co.uk/article/instagram-doubles-to-half-billion-users
[2]https://unionmetrics.com/blog/2017/11/include-image-video-tweets/
[3]https://blog.hubspot.com/marketing/visual-content-marketing-strategy



(a) (b) (c) (d) (e)

FIGURE 1. Offensive memes examples on social media.

TABLE 1. Brief summary of offensive memes classification models and performance results. Where A = Accuracy, F = F1-Score, WF = Weighted F1-score, and R = Recall.

Paper	Dataset Structure	Models	Results %	Target Classes
[38]	Image Classification	ResNet50	48.0 WF	Troll or Non-Troll memes
[39]	Greeks Tweets Multimodal	Combine of ResNet and BERT	94.7 F	Racist and xenophobic speech detection.
[40]	Multimodal Posts	Combine CNN with binary particle swarm optimization and VGG-16	74 WF	non-aggressive, medium aggressive and high-aggressive
[41]	Memes	CNN, VGG16, Inception, Multilingual-BERT, XLMRoberta, XLNet models	58 WF	Troll, Not-Troll
[42]	Memes	BiLSTM and CNN were used for image and text features	30 WF	Troll, Not-Troll
[43]	Memes	Multi layer dense network structure with NLP, RNN, GloVe and FastText, and LSTM	98.0 A	Offensive, non-offensive, slightly offensive, very offensive
[44]	Memes	Transformer-based image encoder, BiLSTM for text encoder Feed-Forward Network as a classification	63.1 R	Offensive or Non-Offensive

model, however, VGG16 has been used for feature extraction from memes. A framework by [21] is based on deep learning to automatically detect the harmful speech in memes based on the fusion of visual and linguistic contents of the memes. To simultaneously classify memes into five different categories like offensiveness, sarcasm, sentiment, motivational, and humor, a multi-task framework via BERT and ResNet is proposed by [22].

A model based on the visual-linguistic transformer is integrated with the pre-trained visual and linguistic features to detect the abusiveness in memes is explained in [23]. To enhance the performance of hateful memes, [24] developed an ensemble learning approach by including classification results from multiple classifiers.

DisMultiHate model is proposed in [25] for the classification of multimodal hateful content. For the improvement of hateful content classification and explainability, they target the entities in memes. A combination of a Feature Concatenation Model (FCM), a Textual Kernels Model (TKM), and a Spatial Concatenation Model (SCM) can be used to boost the multimodel memes classification [26]. A framework named deep learning-based Analogy-aware Offensive Meme Detection (AOMD) by [27] is proposed

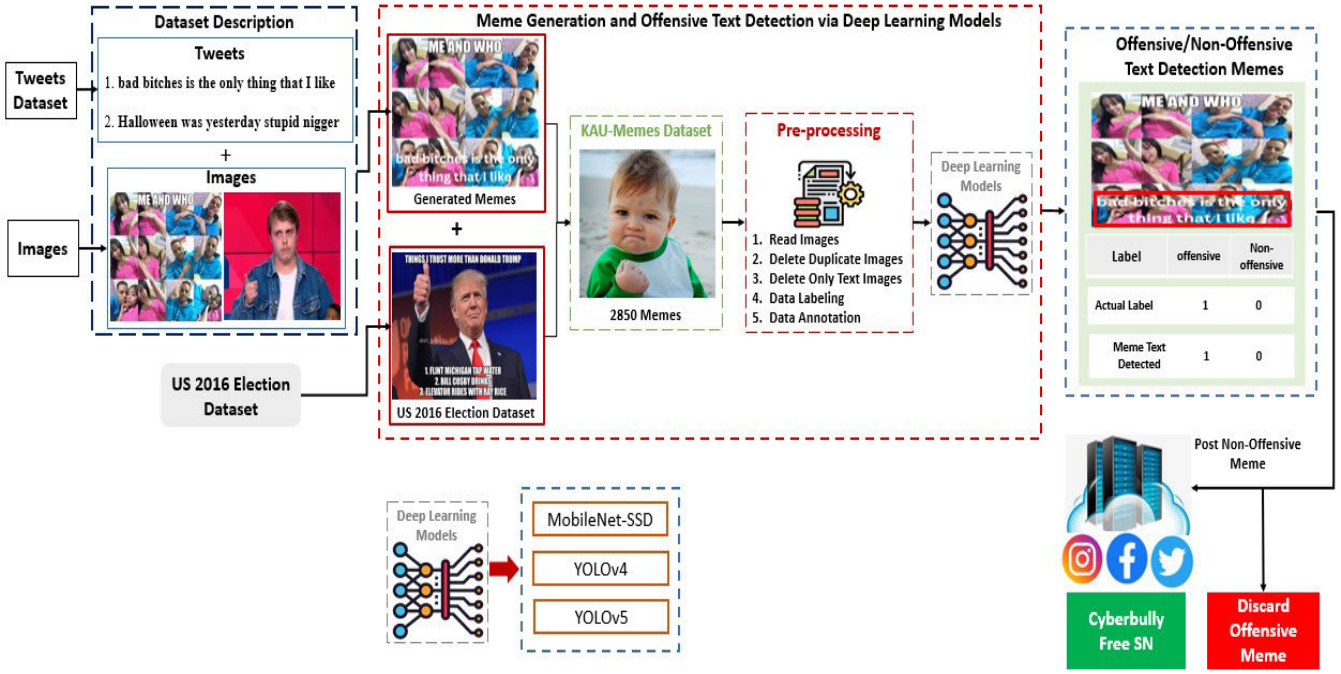


FIGURE 2. Proposed framework for offensive and non-offensive text detection in Memes.

which learns the implicit analogy from the memes to detect offensive analogy memes. KnowMeme model, which is based on a knowledge-enriched graph neural network that uses the information facts from human commonsense can accurately detect offensive memes [28]. Reference [29] proposed that convolutional neural networks (CNN), VGG16, and bidirectional long short-term memory (BiLSTM) can be used for the offensive and non-offensive classification in multimodal memes. Reference [30] proposed a joint model to classify undesired memes based on counteractive unimodal features and multimodal features. For making the constituent module of the framework they employed multilingual-BERT, multilingual-DistilBERT, XLM-R for textual and VGG19, VGG16, and ResNet50 for visual. A textual, visual, and info-graphic cyberbully is detected based on a deep neural architecture which includes Capsule network deep neural network with dynamic routing for textual bullying content detection and CNN for visual bullying content prediction and discretizing the info-graphic content by separating image and text from each other by Google Lens [31]. A deep learning-based framework for a bully or non-bully identification based on residual BiLSTM and RCNN architecture is discussed in [32]. Reference [33] explained that hate speech detection can be improved by augmenting text with image-embedding information.

A new approach by [34] named WELFake is suggested. They used 20 linguistic features and then combined these features with word embeddings and implemented voting classification. This model is based on a count vectorizer and Tf-idf word embedding and uses a machine learning classifier. For unbiased dataset creation, they merged four existing datasets named Kaggle, McIntire, Reuters, and BuzzFeed;

A dataset of images with their comments is collected from Instagram and labeled with the help of Crawdflower workers. Where the criteria for labeling were to i) does the example create cyber aggression which means the image intentionally harms someone, or ii) does it create cyberbullying which mean is there any aggressiveness that contains against a person who can not defend herself or himself [35]. This dataset is also used by [36] for the detection of cyberbullying detection. Another dataset from [37] is collected from Instagram posts and their comments which consist of 3000 examples. They asked two questions i) do the comments contain any bullying ii) If yes, is the bullying due to the contents of the image to label the dataset?

Some state-of-the-art papers' summary has explained in Table 1. This shows us how each of the models performs toward offensive memes classification.

III. PROPOSED FRAMEWORK FOR OFFENSIVE MEMES FILTERING ON SOCIAL MEDIA

The proposed model for offensive and non-offensive text detection in memes is depicted in Figure 2. The goal is to train the model with the training dataset and then test the model with the test dataset to check the performance comparison of the YOLOv4, YOLOv5, and SSD MobileNet-V2 models. This platform can be used as a plug-in for heterogeneous social media to filter out offensive memes. As millions of memes on social media can not be filtered out manually. This approach can help us to overcome the spread of offensive memes that are already posted and will be posted on social

media. After the data preprocessing, YOLOv4, YOLOv5, and SSD MobileNet-V2 models are used to detect the offensive and non-offensive text in memes. The model trains over the dataset and generates weights and checkpoints. When the YOLO model is trained over the labeled image dataset it generates weights files. These files are usually named *yolov-final.weights* with the extension of weights. The weights file can be used as a plug-in for any social media in the future. Plug-ins also known as extensions, add-ons, or computer software can be added to a host program to add new functions without making any changes in the host program. In our case, it can be added to Facebook, Twitter, Instagram, etc. It enables programmers to update a main program while keeping the user within the program's environment. So, the model will discard memes to upload on social media when there is offensive content.

Let's consider an image that contains blood, a gun, private parts of the body, or something else in the image. If someone uploads such kinds of images, the Facebook algorithm discards that image or just shows us that ''this photo may show violent or graphic content'' as everyone has experienced this while using social media. Let's consider an image that contains offensive text. What if someone uploads any of the images from Figure 1 1 as we can see in these images there is offensive text targeting politicians. No one has experienced that Facebook or any other social media can do the same for those images or videos that contain offensive text. These models have trained with the training labeled (bounding boxes) images KAU-Memes dataset. When the training process is finished YOLO or SSD models generate a final weight or a checkpoint. Consider these weights or checkpoints as a trained AI model. Now, let's assume an image with some text and when we pass it from the trained AI model (weight or checkpoint). The trained AI model gives us a bounding box on the text inside the image and decides whether the text is offensive or not. As there are thousands of images in any social media database so this can be used by just executing a for loop over that database and passing images one by one from this trained AI model (weight or checkpoints) and the trained AI model makes a decision in the image as a bounding box and if the bounding box labeled is offensive then delete that image and if the bounding box is not offensive then keep the image in the database. Social media and their databases are filled with such kinds of images. So, this AI-trained model helps us to delete those images that have offensive text from the database of any social media, rather than checking images one by one manually because there are millions of images uploaded This is explained in Algorithm 1 1. This plugin can be installed on any social media and every future image should be passed over it if there is offensive text discard it and do not allow it to upload on social media.

YOLOv4, YOLOv5, and SSD MobileNet are famous for their robustness, accuracy, and real-time object detection. Here these models are used for the first time to detect offensive text inside the images. YOLO for COVID-19 automatic detection from X-ray images is explained in [45]. YOLO is used for electrical component recognition in real-time [46]. Pedestrian detection in real time but at night is explained in [47]. By using these models the images having offensive text can be detected immediately and accurately before it goes viral. Even this trained AI model can be installed in a camera and the camera can fit in a two-tire vehicle. There are many offensive texts in the streets as can be seen on this website [48]. This can be used as a smart city and when the camera detects offensive text on the street walls there should be some action to clean that wall from those offensive words.

Algorithm 1 Algorithm for Detecting Offensive Text in Image

```
Images ← ImagesInDatabase
for image in Images do
    Ofensive ← Checkpoints(image)
    if Ofensive == "offensive" then
        Delete image
    else
        Keep image
    end if
end for=0
```

A. DATA GENERATION

This section explains the KAU-Memes dataset which contains the images having text embedded in these images. The text in the images is offensive and non-offensive memes. Before the data generation, the algorithms were tested in 2016.U.S.Election 738 memes dataset got higher performance, but the dataset consisted of fewer memes, so the model performance was poor. To improve the performance, this approach generates memes by a third-party website.[4] However, for meme generation, there is a need for a text dataset that can be embedded in images. So, this approach used the offensive tweets dataset from [14] embedded it on famous images, and generated a new KAU-Memes dataset. This dataset consists of 24802 labeled tweets; however, only a few of them were used to generate 2850 memes. While generating the memes, the text on images was embedded in different colors, fonts, and angles. The model can filter every kind of offensive meme on social media.

B. TEXT VARIATION IN MEMES

There are hundreds of text fonts, colors, and orientations in memes on social media. Memes can have any form of text and background image. This section explains different types of variations of text in memes. Figure 3 (a) shows the most challenging and common variation of text which is found in the dataset. While generating the data, text in different orientations was embedded over images to make the model more robust and accurate. This model also tried to take care

[4]https://imgflip.com/memegenerator

of the image background clutter because every time, there can be a different image in the memes which is shown in Figure 3 (b). The text position in the image can be seen in Figure 3 (c). Sometimes the text can be in the center of the meme, below the image, or maybe to the left or right side of the image. The size of the text also varies in memes. So KAU-Memes dataset also exists in such kind of variation in the text as shown in Figure 3 (d). Last but not least, the dataset also consists of different formats and colors of text and also blur text, which can be seen in Figure 3 (e), (f) respectively. There are yellow, black, white, etc., color formats for offensive and non-offensive text.

(a) Text orientation (b) Background clutter

(c) Text Position (d) Text Scale

(e) Text Colors (f) Blur Text

FIGURE 3. Common variations KAU-Memes data showing different background clutters, scales, colors, and orientations.

C. PREPROCESSING AND CLEANING OF THE DATASET

This section highlights the importance of data preprocessing and cleaning. Usually, in a huge dataset, it is nearly common that the data may have repetition. There are two or more than two time repetitive images in the 2016.US.Election dataset. As the data repetition can make the model overfit. To remove such kind of duplicate images from the dataset, the duplication tool[5] [49] is used. This is upto date repository based on CNN, Perceptual hashing (PHash), Difference hashing (DHash), Wavelet hashing (WHash), and Average hashing (AHash). Also, the memes were removed manually from the dataset, which only consisted of text, and there was no image in the background.

D. DATA ANNOTATION AND LABELING

For the annotation procedure, the dataset in [4], and for new memes which were generated, they are labeled according to the tweets dataset of [14]. For the manual data annotation, the labeling tool Roboflow[6] is used. The bounding boxes around the text in the memes are made in a manner allowing users to decide whether that text is offensive or non-offensive. This bounding box helps the model because it localizes the area for the YOLO and SSD MobileNet. Roboflow tool generates a text file for each meme with the same file name as the image. Roboflow generates the coordinate in the form of (x1,y1) and (x2, y2) with the label 0 if offensive and 1 if non-offensive, in a text file.

E. THE PRINCIPLE OF YOU ONLY LOOK ONCE (YOLO) MODEL

You Only Look Once (YOLO) is usually used for object detection. It detects the object in the image as a regression problem. Unlike other models, YOLO doesn't do the sliding window, and YOLO looks at the entire image when it is training and testing and implicitly encodes the class information. Many deep learning algorithms are available; however, they cannot detect the object in a single run. YOLO also makes the detection in a single forward propagation through a neural network which makes it suitable for real-time applications. YOLO outperforms the top detection models like DPM and R-CNN [50].

The YOLO detector analyzes the image at once, so the detection obtained by YOLO is based on all the information in the image. Using the input image features, the algorithm splits an image into an SxS grid. The rectangle box is then produced using the confidence score of the detected object extracted from each grid in the introduced image, as shown in Figure 4. Each cell predicts the bounding box and confidence score. The bounding box contains five prediction parameters, which are determined by (x, y, w, h) and the confidence value, where (x, y) coordinates represent the center of the bounding box, and (h, w) reflects the height and the width of the entire image. The confidence scores represent the measurement of how confident the architecture is that the box contains the object (text) to be predicted.

1) YOU ONLY LOOK ONCE VERSION 4 (YOLOV4)

YOLOv4 architecture has some improvements to the older versions. The backbone for YOLOv4 is CSPDarkent53.

[5]https://github.com/idealo/imagededup
[6]https://roboflow.com/

(a) Input Image (b) Division into SxS grid (c) Class probability (d) Bounding boxes (e) Bounding box and successful detection of the text

FIGURE 4. A generalize illustration of YOLO pipeline for offensive and non-offensive text detection in memes.

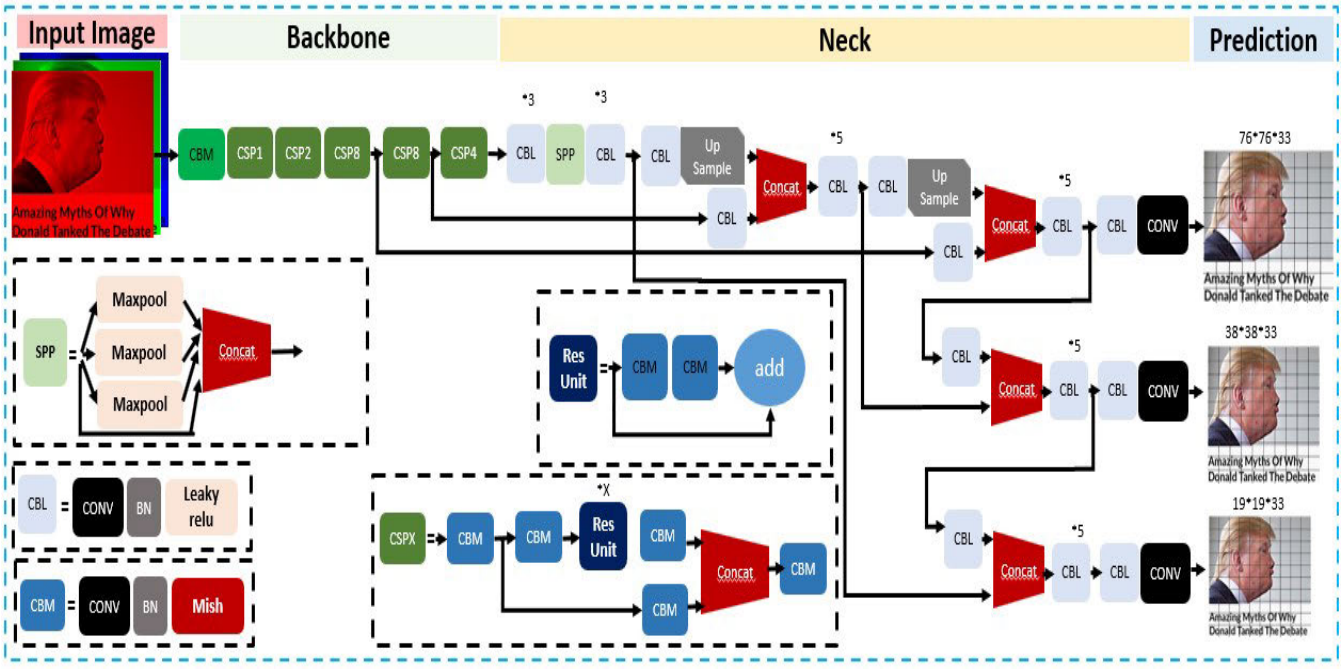


FIGURE 5. YOLOv4 architecture.

Because of this new network, the model can keep the accuracy and reduce the computation. Also, the path aggregation network (PANet) is used in YOLOv4, which can help the model to boost the information flow networks [51].

2) YOU ONLY LOOK ONCE VERSION 5 (YOLOV5)

On the other side, the YOLOv5 is compiled by PyTorch. Due to the application features of PyTorch, the model has high productivity and flexibility. YOLOv5 uses the same CSPDarknet and PANet, as can be seen in Table 2. For the activation function, YOLOv5 uses a sigmoid function rather than the Mish function for YOLOv4 [52].

YOLO algorithms are robust in real-time object detection and represented by Redmon in 2016 [50]. The 4th version of YOLO was released in 2020 [53], compare to the old version of YOLO, the mAP and FPS were improved to 10% and 12%, respectively. They made many changes in the architecture of YOLO models, but the major changes are the adjustment of network structure and an increasing number of applied tricks. YOLOv4 changed the backbone to CSPDarknet53 from the old Darknet53. Some data augmentation techniques were also adopted, i.e., Cutout, Grid Mask, Random Erase, Hide and Seek, Class label smoothing, MixUp, Self-Adversarial Training, Cutmix, and Mosaic data augmentation.

After a few months, another company named Ultralytics released a new version of YOLO named YOLOv5. Instead of publishing research and comparison with other models of YOLO, the company just released YOLOv5's source code on GitHub [54]. However, the main changes in architecture between YOLOv4 and v5 and the advancement in YOLOv5 are presented in Figure 5 and 6, respectively. In YOLOv5 leaky Relu is adopted as an activation function (CBL module)

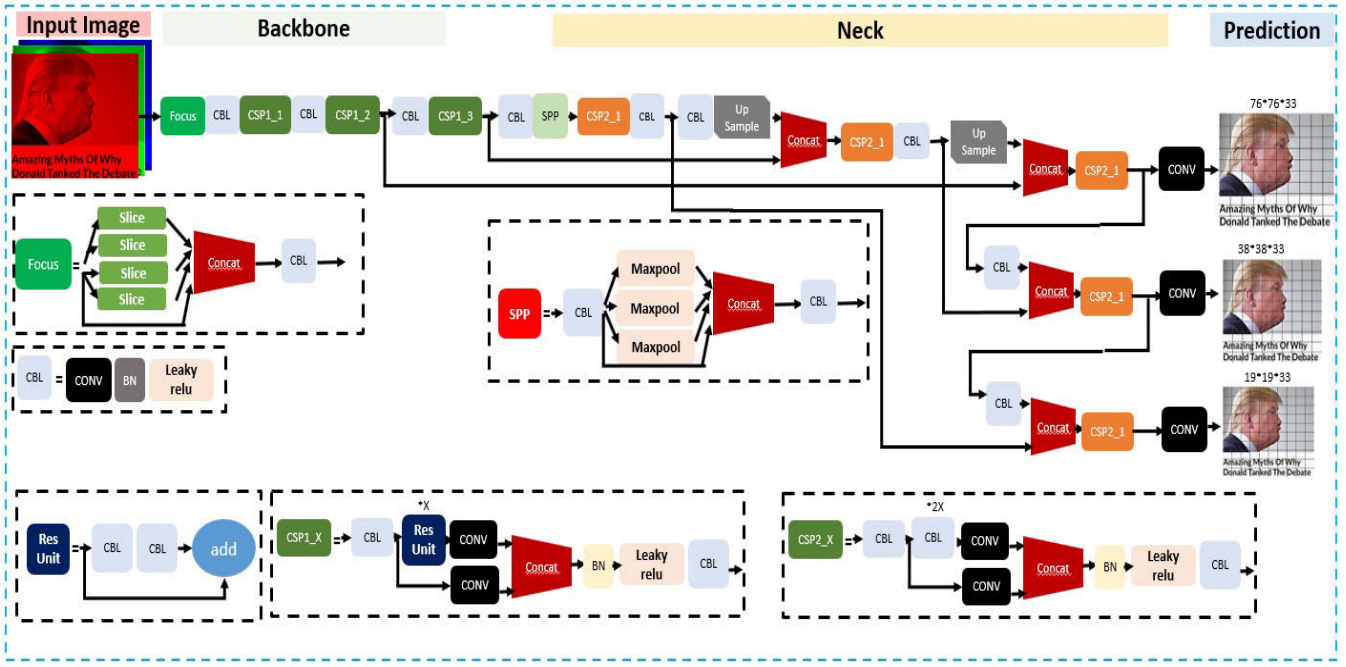


FIGURE 6. YOLOv5 architecture.

in the hidden layers, while in YOLOv4, there are two modules with leaky Relu and mish activation functions (CBL and CBM). Secondly, in the backbone, YOLOv5 adopted a new module at the beginning that is named Focus. Focus makes four slices of an input image and concatenates all of them for convolution operation. For example, an image of 608 × 608 × 3 is divided into four small images with 304 × 304 × 3, concatenated into a 304 × 304 × 12 image. Third, for the backbone and neck, YOLOv5 designed two CSPNet modules. To maintain processing accuracy and reduce computation power, CSPNet combines feature maps from the start and at the end of a network stage [55]. Compared to the standard convolution module in YOLOv4, YOLOv5 adopted the CSPNet module, i.e., CSP2_x in the neck, to strengthen the network feature fusion. Besides the structure adjustment, YOLOv5 adopted an algorithm to automatically learn bounding box anchors in the input stage, which could help calculate the anchor box size for other image sizes and improve the detection quality. Except this, YOLOv5 uses Generalize Intersection Over Union (GIoU) as a loss as shown in Equation 1 [56] for the regression loss function in the bounding box instead of Complete Intersection Over Union (CIoU) loss in YOLOv4 as shown in Equation 2. GIoU can solve the imperfect calculation of non-overlapping bounding boxes that remain in the previous Intersection Over Union (IoU) loss function. CIoU incorporates all three geometric factors: including distance, aspect ratio, and overlapping area. To better determine difficult regression cases, CIoU enhances the accuracy and speed. YOLOv5 is constructed under a new environment at PyTorch [56], which makes the training procedure more friendly than Darknet.

$$\mathcal{L}_{GIoU} = 1 - IoU + \frac{|C - B \cup B^{gt}|}{|C|} \tag{1}$$

where B^{gt} represents the ground truth box, B is the predicted box, C is equal to the smallest box which covers B and B^{gt}, and $IoU = (B \cap B^{gt})/B \cup B^{gt}$ is the intersection over the union.

$$\mathcal{L}_{CIoU} = 1 - IoU + \frac{\rho^2(p, p^{gt})}{c^2} \tag{2}$$

where p^{gt} and p are the central points of boxes B and B^{gt}, is represented Euclidean distance, c is the diagonal length of the smallest box C, V and are the consistency of the aspect ratio.

TABLE 2. Architecture comparison of YOLOv4 and YOLOv5.

	YOLOv4	YOLtherefore to addressOv5
Backbone	CSPDarknet53	CSPDarknet53
Neck	SSP & PANet	PANet
Neural Network Type	Fully Convolution	Fully Convolution
Head	YOLO Layer	YOLO Layer
Framework	Darknet	PyTorch

F. SINGLE SHOT DETECTOR (SSD)

In the field of computer vision, the models become more complex and deeper for more accurate results and performance. However, the advancement makes the model latency

and size bigger, which cannot be used in a system that has computational challenges. SSD-MobileNet can help in such kinds of challenges. This model is basically designed for those situations that require high speed. The MobileNetV2 provides an inverted residual structure for better modularity. MobileNet eliminates the non-linearities in tight layers and results in higher performance for previous applications. The MobileNet-SSD detector inherits the design of VGG16-SSD, and the front-end MobileNet-v2 network provides six feature maps with different dimensions for the back-end detection network to perform multi-scale object detection. Since the backbone network model is changed from VGG-16 to MobileNet-v2, the MobileNet-SSD detector can achieve real-time performance and is faster than other existing object detection networks.

G. EVALUATION MATRIX

Usually, a basic matrix intersection over union (IoU) is used to evaluate the performance of object detection models, which can be seen in Figure 7. IoU is the overlap of the detection box (D) and the ground truth box (G), which can be calculated by using Equation 3 [57]. When we obtain the IoU, then we use the confusion matrix, i.e., False Positive (FP), True Positive (TP), False Negative (FN), and True Negative (TN) for accuracy measurement. For TP, a specific class ground truth must be the class of detection, also the IoU must be greater than 50%. As TP is the correct detection of the class. In case the detection owns the same class as the ground truth, and the IoU is less than 50%, then it is considere FP. Which means the detection is not corrected. If the model does not make detection and there is a ground truth, then it is considere FN, which means that the instance is not detected. In many cases, the background does not have any ground truth and also no detection, so that is classified as TN.

$$IoU = \frac{Intersection}{Union} = \frac{G \cap P}{G \cup P} \tag{3}$$

FIGURE 7. Examples of intersection over union on text in a meme.

For the performance comparison of YOLOv4, YOLOv5, and SSD MobileNet-V2 algorithms mAP, F1-Score, Precision, and Recall can be used as criteria. Where mAP [58] is the mean average precision is the mean of average precision (AP) as shown in Equation 4. Where n is the number of classes while the AP is the average precision for that given class n. mAP returns a score after comparing the ground truth bounding box with the detected box. After taking the mean of AP, we can get the mAP which can be used to calculate the accuracy of machine learning algorithms. F1-score [59] measures a model's accuracy over the dataset and can use to evaluate binary classification problems. Equation 5 can be used for the F1-score calculation using precision and recall. The highest possible value for the F1-score is 1 and the lowest is 0. The precision is the ratio of true prediction with the total number of predictions, while the recall is the ratio of true prediction to the total number of objects in the image [60], which are shown in Equation 6 and Equation 7 respectively.

$$mAP = \frac{1}{n}\sum_{k=1}^{n} AP_k \tag{4}$$

$$F1Score = 2 * \frac{precision * recall}{precision + recall} \tag{5}$$

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{6}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{7}$$

IV. RESULTS AND DISCUSSION

A. EXPERIMENTAL SETUPS

All the models were trained on Colab Pro+, and the resources used for the training are shown in Table 3. To train the models properly, YOLOv4, YOLOv5, and SSD MobileNet with different parameters were trained to achieve the highest possible mAP. The parameters for each of the models are shown in Table 4.

TABLE 3. Colab specification used for the training of YOLOv4, YOLOv5 and SSD MobileNet.

CPU	GPU	RAM	Storage	OS	Platform
12	SXM4-40GB0	83.5 GB	166.8 GB	Posix	Linux

TABLE 4. Training hyperparameters for each of the models to achieve the highest possible performance.

Model	α	Batch	Momentum	Decay	Image
MobileNet	0.006	32	0.9	0.95	320x320
YOLOv4	0.001	64	0.949	0.0005	416x416
YOLOv5	0.01	64	0.949	0.0005	604x604

B. PERFORMANCE BASED ON EVALUATION MATRIX

The ML models achieved the highest possible results for the public online dataset in [4] consisting of 743 memes, and

the models performed poorly because of the small number of memes. The results for the best possible parameters for each of the models are shown in Table 5.

TABLE 5. Performance of ML models on 743 memes.

Model	mAP %	Precision %	Recall %	F1-Score
MobileNet	35.52	9.23	8.12	8.63
YOLOv4	42.31	13.1	10.31	11.53
YOLOv5	**45.12**	**13.5**	**11.66**	**12.51**

After the models were appropriately trained, YOLOv5 achieved the lowest 17 MB weight, the highest mAP of 88.50%, and consumed 11.60 milliseconds on average for 152 offensive and non-offensive text detection in memes. Similarly, for YOLOv4, the weight was 244 MB and the text detection time was 42.68 milliseconds. SSD MobileNet text detection time is lower than YOLOv4 and higher than YOLOv5 with an IoU of 0.5. Also, YOLOv5 consumed the lowest time while training. However, among all models, YOLOv5 performance was found to be the best based on training and detection time and smallest weight/checkpoint size as shown in Table 6. In the case of YOLOv4 and MobileNet, the training time consumed by YOLOv4 was recorded less than MobileNet but the size of checkpoints of YOLOv4 was higher. Also, the detection time on GPU by MobileNet was 31.28 milliseconds which is less than the YOLOv4 detection time.

TABLE 6. Weight and checkpoint size of YOLOv4, YOLOv5 and SSD-MobileNet, training and detection time.

Model	Training	Size MB	Image	mAP %	Time GPU
MobileNet	541	24.8	320x320	81.74	31.28
YOLOv4	267	244	416x416	85.20	42.68
YOLOv5	**32**	**17**	**608x608**	**88.50**	**11.60**

Using the KAU-Memes dataset, this approach performs three different experiments. In the first experiment, the data is split into 90% training and 10% validation sets, in the second experiment 80% and 20%, and in the third experiment 70% and 30% training and validation.

The results of both models can be seen in Table 7 using the KAU-Memes dataset for offensive text detection in memes. It is clear from the table that YOLOv5 shows the highest mAP of 91.40%, the precision of 86.2%, recall of 91.9%, and F1-score of 88.4% than YOLOv4 while splitting the dataset in 90% training and 10% validation. Also, the YOLOv5 has

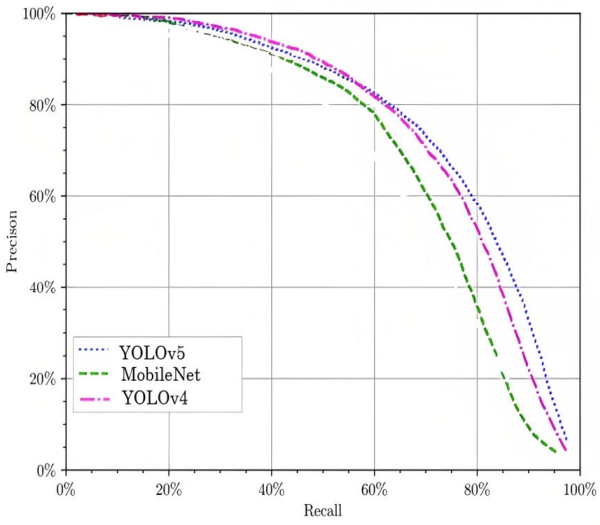


FIGURE 8. Precision × recall plot on the validation set considering all the predictions.

good performance than YOLOv4 and SSD MobileNet, even for the 80% and 70% training of the dataset. Because of the high mAP and F1-score, the offensive and non-offensive text detection in memes by YOLOv5 are more accurate compared to YOLOv4 and SSD-MobileNet.

C. PRECISION VS RECALL

Using all the predictions for the detection of offensive text in memes, a curve for precision vs. recall is plotted as shown in Figure 8. The curve established the settlement between the precision and recall rate. Higher confidence means higher precision in their predictions but a lower recall. YOLOv4 and YOLOv5 had nearly 90% recall rates. The best model is the one whose Area Under the Curve (AUC) is the highest. Therefore, it can be seen that YOLOv5 AUC is the highest compared to YOLOv4 and SSD MobileNet.

D. DETECTION RESULTS

Some offensive and non-offensive text detection were performed to find the model's performance. In Figure 9 (a), the YOLOv5 predicts the offensive text with a high confidence of 0.96, and also, for non-offensive, the confidence value is almost 0.93. Similarly, Figure 9 (b) shows the prediction of offensive text and non-offensive using the YOLOv4 model with the confidence of 0.86 and 0.80, respectively.

Other than YOLOv4 and v5, the performance of SSD MobileNet-V2 is also good for the detection of offensive memes. The SSD-MobileNet V2 detects the offensive text with a confidence of 0.83 for the offensive meme and 0.78 confidence for non-offensive, which can be seen in Figure 9 (c).

E. MODELS PERFORMANCE LOSS

To explore the performance of the algorithm in more detail, it is necessary to find their incorrect detection. Which

TABLE 7. Results of YOLOv4, YOLOv5, and SSD MobileNet V2 algorithms for offensive and non-offensive memes text detection with a train-validation split of 90%-10%, 80%-20%, and 70%-30%.

Algorithms	Data Splits (%)											
	90-10				80-20				70-30			
	mAP (%)	F1 (%)	P (%)	R (%)	mAP (%)	F1 (%)	P (%)	R (%)	mAP (%)	F1 (%)	P (%)	R (%)
YOLOv4	87.69	86.0	83.0	90.0	85.20	84.0	84.0	84.0	83.47	82.0	80.0	80.0
YOLOv5	**91.40**	**88.4**	**86.2**	**90.9**	**88.50**	**88.8**	**90.2**	**87.5**	**88.10**	**88.4**	**89.4**	**87.5**
MobileNet V2	83.82	85.3	83.8	86.9	81.74	84.1	81.7	86.8	80.15	73.7	80.2	68.2

(a) Offensive Text Detection by YOLOv5

(b) Offensive Text Detection by YOLOv4

(c) Offensive Text Detection by MobileNet

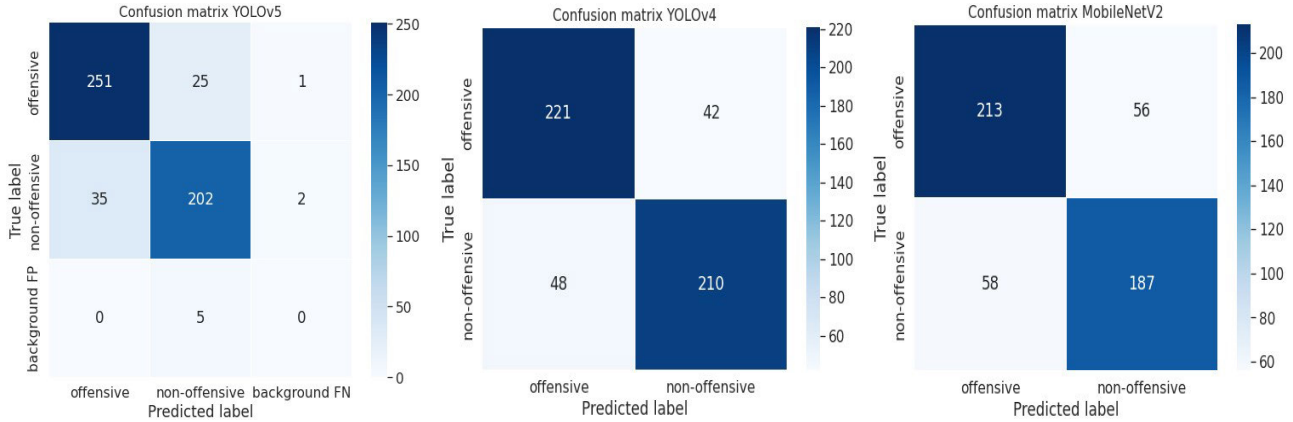
FIGURE 9. Offensive text detection by YOLOv5, YOLOv4, and SSD-MobileNeta.

(a) Small Offensive text detection by YOLOv5

(b) Small Offensive text detection by YOLOv4

(c) Small Offensive text detection by SSD MobileNet

FIGURE 10. Offensive small text detection by YOLOv5, YOLOv4, and SSD-MobileNet.



(a) Confusion matrix by YOLOv5

(b) Confusion matrix by YOLOv4

(c) Confusion matrix by MobileNetV2

FIGURE 11. Confusion matrix from YOLOv5, YOLOv4, and SSD-MobileNet.

can help in future research for improvement. YOLOv5s is the best detection among other models. However, when it comes to detecting the offensive text in a meme that has small size text, the performance goes down for each of the models. In Figure 10, all the models detect the offensive text with a different confidence score. Among all the models, YOLOv5s still performs better for small text detection than YOLOv4 and SSD-MobileNet. YOLOv5 detects small text with a confidence score of 0.88, YOLOv4 achieves 0.81, and SSD MobileNet achieves a 0.75 confidence score. The performance can be improved by adding more images having offensive text in small sizes.

F. ANALYSIS BASED ON CONFUSION MATRIX

A confusion matrix can be used for the performance of different models. The confusion matrix also provides information on the type and source of errors. Where the elements on the diagonal represent all the correct classes. The confusion matrix for YOLOv5s is shown in Figure 11 (a) where the offensive text is detected 251 times correctly, but the model confused 35 times with non-offensive text. Similarly, the non-offensive text is detected 202 times correctly, but it is confused with offensive text around 25 times. In the YOLOv5 confusion matrix, the False Positive (FP) is divided into two parts based on the value of IOU. If IOU $= 0$, the false positive prediction is far from the ground truth. Also, if IOU is between 0 and 0.5 then the overlap between the ground truth and prediction is not enough to decide it as a true positive. For YOLOv4, the offensive text is detected 221 times and non-offensive text 210 times correctly, but 48 times the offensive text is confused with non-offensive text, and 42 times the non-offensive text is confused with offensive text detection as shown in Figure 11 (b). Similarly, for SSD-MobileNet V2, the model detected offensive text 213 times but confused 58 times with non-offensive and non-offensive text confused with offensive text 56 times, as shown in Figure 11 (c).

V. CONCLUSION

In this approach, a new framework for better detection of offensive content in unstructured data for heterogeneous social media is proposed. In terms of accuracy and speed, the newly tested framework was systematically applied to two versions of YOLO and SSD MobileNet with different parameters. For the SSD MobileNet using different parameters. Hence, it was observed: (1) for the SSD MobileNet model, the increased number of training images size could not contribute to better performance; (2)as shown in Table 5, a big gap in the mAP and F1 scores between SSD MobileNet and YOLO versions.

In addition, YOLOv5s achieved the highest mAP of 88.50%, a faster training time of 32 minutes for 80% and 20% of training and validation data, and a faster processing speed for multiple meme detection of 11.6 milliseconds. The YOLOv5 model still had the best performance in comparison with YOLOv4 and MobileNet.

In this approach mAP, F1-score, precision, and recall were used to evaluate the feasibility of the proposed framework. As explained in the results section, trained the models on publicly available datasets to verify all the corresponding outputs for accuracy. In the final analysis, it was determined that the results were not good enough to use the model for future and unknown datasets.

In parallel, a new KAU-Meme dataset was generated to detect the desired two classes of offensive and non-offensive text in unstructured data. This dataset contained 2582 high, average, and low-quality memes from the combination of 2016 US Election memes and tweets datasets. These selected images contain some of the most popular memes used on social media and are labeled based on strict criteria. To encourage future novel research, the dataset is available on GitHub as well as by email to the corresponding author and primary author of this paper.

Compared the performance of all models based on training and detection times, evaluation matrix, detection of text in the memes, precision and recall curve, small text detection, and confusion matrices. After evaluating all the results, YOLOv5 performance was the best based on training, detection time, mAP, precision vs. recall curve, the detection confidence score of normal and small text in memes, and confusion matrix.

However, there are some limitations to this approach in that the model performed poorly when the meme contained small text. Expanding the dataset with more small text memes can help to improve the performance. Secondly, the number of classes can improve because this approach is only limited to two classes however, there are other classes, such as harassment, propaganda, sexual aggression, violence, and racism which spread via Facebook, Twitter, WhatsApp, and Reddit. Thirdly, the model is limited to detecting offensive English text, and future, it can be improved for other languages. Fourthly, It is also possible to detect offensive text and also to detect the image inside the meme because it is possible that the photo may be some famous personality or belongs to some religion, so it will be stronger to detect the target of the offensive meme.

REFERENCES

[1] J. H. French, "Image-based memes as sentiment predictors," in *Proc. Int. Conf. Inf. Soc. (i-Soc.)*, Jul. 2017, pp. 80–85.
[2] M. R. Mirsaleh and M. R. Meybodi, "A Michigan memetic algorithm for solving the community detection problem in complex network," *Neurocomputing*, vol. 214, pp. 535–545, Nov. 2016.
[3] S. He, X. Zheng, J. Wang, Z. Chang, Y. Luo, and D. Zeng, "Meme extraction and tracing in crisis events," in *Proc. IEEE Conf. Intell. Secur. Informat. (ISI)*, Sep. 2016, pp. 61–66.
[4] S. Suryawanshi, B. R. Chakravarthi, M. Arcan, and P. Buitelaar, "Multimodal meme dataset (MultiOFF) for identifying offensive content in image and text," in *Proc. 2nd Workshop Trolling, Aggression Cyberbullying*, 2020, pp. 32–41.
[5] A. Sengupta, S. K. Bhattacharjee, M. S. Akhtar, and T. Chakraborty, "Does aggression lead to hate? Detecting and reasoning offensive traits in hinglish code-mixed texts," *Neurocomputing*, vol. 488, pp. 598–617, Jun. 2022.
[6] A. Kumar and G. Garg, "Sarc-M: Sarcasm detection in typo-graphic memes," in *Proc. Int. Conf. Adv. Eng. Sci. Manag. Technol. (ICAESMT)*, Dehradun, India: Uttaranchal Univ., 2019.
[7] Y. Zhou, Z. Chen, and H. Yang, "Multimodal learning for hateful memes detection," in *Proc. IEEE Int. Conf. Multimedia Expo. Workshops (ICMEW)*, Jul. 2021, pp. 1–6.
[8] R. N. Nandi, F. Alam, and P. Nakov, "Detecting the role of an entity in harmful memes: Techniques and their limitations," in *Proc. Workshop Combating Online Hostile Posts Regional Lang. During Emergency Situations*, 2022, pp. 43–54.
[9] N. Islam, Z. Islam, and N. Noor, "A survey on optical character recognition system," *J. Inf. Commun. Technol.*, vol. 10, no. 2, pp. 1–4, Dec. 2016.
[10] J. Drakett, B. Rickett, K. Day, and K. Milnes, "Old jokes, new media—Online sexism and constructions of gender in Internet memes," *Feminism Psychol.*, vol. 28, no. 1, pp. 109–127, Feb. 2018.
[11] S. T. Aroyehun and A. Gelbukh, "Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling," in *Proc. 1st Workshop Trolling, Aggression Cyberbullying (TRAC)*, 2018, pp. 90–97.
[12] L. G. M. de la Vega and V. Ng, "Modeling trolling in social media conversations," in *Proc. 7th Int. Conf. Lang. Resour. Eval. (LREC)*, 2018. [Online]. Available: https://aclanthology.org/L18-1585/
[13] I. Arroyo-Fernndez, D. Forest, J.-M. Torres-Moreno, M. Carrasco-Ruiz, T. Legeleux, and K. Joannette, "Cyberbullying Detection Task: The EBSI-LIA-UNAM System (ELU) at COLING'18 TRAC-1," in *Proc. 1st Workshop Trolling, Aggression Cyberbullying (TRAC)*, 2018, pp. 140–149.
[14] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," in *Proc. Int. AAAI Conf. Web Social Media*, vol. 11, no. 1, 2017, pp. 512–515.
[15] D. Wang, B. K. Szymanski, T. Abdelzaher, H. Ji, and L. Kaplan, "The age of social sensing," *Computer*, vol. 52, no. 1, pp. 36–45, Jan. 2019.
[16] D. Choi, S. Chun, H. Oh, J. Han, and T. Kwon, "Rumor propagation is amplified by echo chambers in social media," *Sci. Rep.*, vol. 10, no. 1, pp. 1–10, Jan. 2020.
[17] M. H. Ribeiro, P. H. Calais, Y. A. Santos, V. A. F. Almeida, and W. Meira, "'Like sheep among wolves': Characterizing hateful users on Twitter," in *Proc. WSDM Workshop Misinformation Misbehavior Mining Web (MIS)*, 2018, Paper e0203794.
[18] C. Van Hee, G. Jacobs, C. Emmery, B. Desmet, E. Lefever, B. Verhoeven, G. De Pauw, W. Daelemans, and V. Hoste, "Automatic detection of cyberbullying in social media text," *PloS One*, vol. 13, no. 10, 2018, Art. no. e0203794.
[19] D. Kiela, H. Firooz, A. Mohan, V. Goswami, A. Singh, C. A. Fitzpatrick, and P. Bull, "The hateful memes challenge: competition report," in *Proc. NeurIPS*, 2021, pp. 344–360.
[20] M. Das, S. Banerjee, and A. Mukherjee, "Hate-alert@DravidianLangTech-ACL2022: Ensembling multi-modalities for Tamil TrollMeme classification," in *Proc. 2nd Workshop Speech Lang. Technol. Dravidian Lang.*, 2022, pp. 51–57.
[21] B. O. Sabat, C. C. Ferrer, and X. Giro-I-Nieto, "Hate speech in pixels: Detection of offensive memes towards automatic moderation," in *Proc. NeurIPS*, 2019, pp. 281–290.

[22] D. S. Chauhan, S. R. Dhanush, A. Ekbal, and P. Bhattacharyya, "All-in-one: A deep attentive multi-task learning framework for humour, sarcasm, offensive, motivation, and sentiment on memes," in *Proc. 1st Conf. Asia–Pacific chapter Assoc. Comput. Linguistics 10th Int. Joint Conf. Natural Lang. Process.*, 2020, pp. 281–290.

[23] R. Zhu, "Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution," 2020, *arXiv:2012.08290*.

[24] D. Kiela, "The hateful memes challenge: Competition report," in *Proc. NeurIPS*, 2021, pp. 5138–5147.

[25] R. K.-W. Lee, R. Cao, Z. Fan, J. Jiang, and W.-H. Chong, "Disentangling hate in online memes," in *Proc. 29th ACM Int. Conf. Multimedia*, Oct. 2021, pp. 5138–5147.

[26] R. Gomez, J. Gibert, L. Gomez, and D. Karatzas, "Exploring hate speech detection in multimodal publications," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 1459–1467.

[27] L. Shang, Y. Zhang, Y. Zha, Y. Chen, C. Youn, and D. Wang, "AOMD: An analogy-aware approach to offensive meme detection on social media," *Inf. Process. Manage.*, vol. 58, no. 5, Sep. 2021, Art. no. 102664.

[28] L. Shang, C. Youn, Y. Zha, Y. Zhang, and D. Wang, "KnowMeme: A knowledge-enriched graph neural network solution to offensive meme detection," in *Proc. IEEE 17th Int. Conf. eScience (eScience)*, Sep. 2021, pp. 186–195.

[29] S. Khedkar, P. Karsi, D. Ahuja, and A. Bahrani, "Hateful memes, offensive or non-offensive," in *Proc. Int. Conf. Innov. Comput. Commun.* Singapore: Springer, 2022, pp. 609–621.

[30] E. Hossain, O. Sharif, M. M. Hoque, M. A. A. Dewan, N. Siddique, and M. A. Hossain, "Identification of multilingual offense and troll from social media memes using weighted ensemble of multimodal features," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 9, pp. 6605–6623, Oct. 2022.

[31] A. Kumar and N. Sachdeva, "Multimodal cyberbullying detection using capsule network with dynamic routing and deep convolutional neural network," *Multimedia Syst.*, vol. 28, no. 6, pp. 2043–2052, Dec. 2022.

[32] S. Paul, S. Saha, and M. Hasanuzzaman, "Identification of cyberbullying: A deep learning based multimodal approach," *Multimedia Tools Appl.*, vol. 81, no. 19, pp. 26989–27008, Aug. 2022.

[33] F. Yang, X. Peng, G. Ghosh, R. Shilon, H. Ma, E. Moore, and G. Predovic, "Exploring deep multimodal fusion of text and photo for hate speech classification," in *Proc. 3rd Workshop Abusive Lang. Online*, 2019, pp. 11–18.

[34] P. K. Verma, P. Agrawal, I. Amorim, and R. Prodan, "WELFake: Word embedding over linguistic features for fake news detection," *IEEE Trans. Computat. Social Syst.*, vol. 8, no. 4, pp. 881–893, Aug. 2021.

[35] H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra, "Detection of cyberbullying incidents on the Instagram social network," 2015, *arXiv:1503.03909*.

[36] V. K. Singh, S. Ghosh, and C. Jose, "Toward multimodal cyberbullying detection," in *Proc. CHI Conf. Extended Abstr. Hum. Factors Comput. Syst.*, May 2017, pp. 2090–2099.

[37] H. Zhong, H. Li, A. C. Squicciarini, S. M. Rajtmajer, C. Griffin, D. J. Miller, and C. Caragea, "Content-driven detection of cyberbullying on the Instagram social network," in *Proc. IJCAI*, vol. 16, 2016, pp. 3952–3958.

[38] M. Balaji J and C. Hs, "TrollMeta@DravidianLangTech-EACL2021: Meme classification using deep learning," in *Proc. 1st Workshop Speech Lang. Technol. Dravidian Lang.*, 2021, pp. 277–280.

[39] K. Perifanos and D. Goutsos, "Multimodal hate speech detection in Greek social media," *Multimodal Technol. Interact.*, vol. 5, no. 7, p. 34, Jun. 2021.

[40] K. Kumari, J. P. Singh, Y. K. Dwivedi, and N. P. Rana, "Multi-modal aggression identification using convolutional neural network and binary particle swarm optimization," *Future Gener. Comput. Syst.*, vol. 118, pp. 187–197, May 2021.

[41] E. Hossain, O. Sharif, and M. M. Hoque, "NLP-CUET@DravidianLangTech-EACL 2021: Investigating Visual and Textual Features to Identify Trolls from Multimodal Social Media Memes," in *Proc. 1st Workshop Speech Lang. Technol. Dravidian Lang.*, 2021, pp. 300–306.

[42] A. K. Mishra and S. Saumya, "Identifying troll meme in Tamil using a hybrid deep learning approach," in *Proc. 1st Workshop Speech Lang. Technol. Dravidian Lang.*, 2021, pp. 243–248.

[43] R. K. Giri, S. C. Gupta, and U. K. Gupta, "An approach to detect offence in memes using natural language processing(NLP) and deep learning," in *Proc. Int. Conf. Comput. Commun. Informat. (ICCCI)*, Jan. 2021, pp. 1–5.

[44] R. Nayak, B. S. U. Kannantha, K. S, and C. Gururaj, "Multimodal offensive meme classification u sing transformers and BiLSTM," *Int. J. Eng. Adv. Technol.*, vol. 11, no. 3, pp. 96–102, Feb. 2022.

[45] A. Karacı, "VGGCOV19-NET: Automatic detection of COVID-19 cases from X-ray images using modified VGG19 CNN architecture and YOLO algorithm," *Neural Comput. Appl.*, vol. 34, no. 10, pp. 8253–8274, May 2022.

[46] H. Chen, Z. He, B. Shi, and T. Zhong, "Research on recognition method of electrical components based on YOLO V3," *IEEE Access*, vol. 7, pp. 157818–157829, 2019.

[47] Y. Xue, Z. Ju, Y. Li, and W. Zhang, "MAF-YOLO: Multi-modal attention fusion based YOLO for pedestrian detection," *Infr. Phys. Technol.*, vol. 118, Nov. 2021, Art. no. 103906.

[48] *Age Foto Stock*. Accessed: Mar. 2, 2023. [Online]. Available: https://www.agefotostock.com/age/en/details-photo/offensive-graffiti-on-shop-shutter-in-rome-italy/Y5G-1951508

[49] T. Jain, C. Lennan, Z. John, and D. Tran, "Imagededup," 2019. [Online]. Available: https://github.com/idealo/imagededup

[50] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[51] S. Li, X. Gu, X. Xu, D. Xu, T. Zhang, Z. Liu, and Q. Dong, "Detection of concealed cracks from ground penetrating radar images based on deep learning algorithm," *Construct. Building Mater.*, vol. 273, Mar. 2021, Art. no. 121949.

[52] D. Thuan, "Evolution of YOLO algorithm and YOLOv5: The state-of-the-art object detention algorithm," Tech. Rep., 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2004.10934

[53] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[54] Ultralytics. *GitHub*. Accessed: Nov. 22, 2022. [Online]. Available: https://github.com/ultralytics/yolov5

[55] C.-Y. Wang, H.-Y. Mark Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 1571–1580.

[56] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, and T. Killeen, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.

[57] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," in *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, Jul. 2020, pp. 237–242.

[58] L. Liu and M. T. Zsu, Eds., *Encyclopedia of Database Systems*, vol. 6. New York, NY, USA: Springer, 2009.

[59] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 233–240.

[60] I. D. Melamed, R. Green, and J. P. Turian, "Precision and recall of machine translation," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol. Companion (HLT-NAACL) Short Papers (NAACL)*, 2003, pp. 61–63.

JAMSHID BACHA received the B.Sc. degree in computer systems engineering from UET Peshawar, Pakistan, in 2020, and the master's degree in computer information systems and networks from Korea Aerospace University, South Korea. He is currently pursuing the Ph.D. degree with Technische Universität Berlin. His current research interests include machine learning, deep learning, computer vision, and wireless communication.

FARMAN ULLAH received the M.S. degree in computer engineering from CASE, Islamabad, Pakistan, in 2010, and the Ph.D. degree from Korea Aerospace University, South Korea, in 2016. He worked and collaborated on various projects funded by the Ministry of Economy, the Korea Research Foundation, and ETRI, South Korea. In 2007, he joined AERO, Pakistan, as an Assistant Manager of telemetry. He is currently an Assistant Professor with the College of IT, United Arab Emirates University (UAEU), Abu Dhabi, Al Ain, United Arab Emirates. Before Joining UAEU, he was an Assistant Professor with the Department of Electrical and Computer Engineering, COMSATS University Islamabad, Attock Campus, Pakistan, and a Postdoctoral Researcher with the High Processing Computing Laboratory, Jeonbuk National University, South Korea. He has authored/coauthored more than 40 peer-reviewed publications. His current research interests include embedded, wearable, the IoT applications, intelligent resource management for high-performance computing, and artificial intelligence and machine learning.

JEBRAN KHAN received the B.Sc. and M.Sc. degrees in computer systems engineering from the University of Engineering and Technology at Peshawar, Peshawar, Pakistan, and the Ph.D. degree in electronics and information engineering from Korea Aerospace University, Goyang, South Korea. He is currently a Postdoctoral Researcher with Ajou University, South Korea. His current research interests include social network analysis, modeling, frameworks, and their applications.

ABDUL WASAY SARDAR received the B.Sc. degree in computer engineering from COMSATS University Islamabad, Pakistan, in 2020, and the master's degree in computer information systems and networks from Korea Aerospace University, Goyang, South Korea. His current research interests include artificial intelligence, machine learning, deep learning, and computer vision.

SUNGCHANG LEE (Member, IEEE) received the B.S. degree from Kyungpook National University, in 1983, the M.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), in 1985, and the Ph.D. degree in electrical engineering from Texas A&M University, in 1991. From 1985 to 1987, he was with KAIST, as a Researcher, where he worked on image processing and pattern recognition projects. From 1992 to 1993, he was a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), South Korea, and the Director of the Government Project on Intelligent Smart Home Security and Automation Service Technology, from 2004 to 2009. In 2009, he was the Vice President of the Institute of Electronics and Information Engineers (IEIE), South Korea, and also the Director of the Telecommunications Society, South Korea. Since 1993, he has been a Faculty of Korea Aerospace University, Goyang, South Korea, where he is currently a Professor with the School of Electronics, Telecommunication and Computer Engineering.