Journal of Computer Science and Cybernetics, V.35, N.2 (2019), 135–145
DOI 10.15625/1813-9663/35/2/13315

REAL-TIME SMILE DETECTION USING DEEP LEARNING

CHI CUONG NGUYEN[1,*], GIANG SON TRAN[1,2], THI PHUONG NGHIEM[1],
JEAN CHRISTOPHE BURIE[3], CHI MAI LUONG[4]

[1]*ICTLab, University of Science and Technology of Hanoi, VAST*
[2]*Sorbonne University, IRD, UMMISCO, F-93143, Bondy, France*
[3]*L3i Laboratory, University of La Rochelle, France*
[4]*Institute of Information Technology, VAST*
[*]*cuongnc@st.usth.edu.vn*

Abstract. Real-time smile detection from facial images is useful in many real world applications such as automatic photo capturing in mobile phone cameras or interactive distance learning. In this paper, we study different architectures of object detection deep networks for solving real-time smile detection problem. We then propose a combination of a lightweight convolutional neural network architecture (BKNet) with an efficient object detection framework (RetinaNet). The evaluation on the two datasets (GENKI-4K, UCF Selfie) with a mid-range hardware device (GTX TITAN Black) show that our proposed method helps in improving both accuracy and inference time of the original RetinaNet to reach real-time performance. In comparison with the state-of-the-art object detection framework (YOLO), our method has higher inference time, but still reaches real-time performance and obtains higher accuracy of smile detection on both experimented datasets.

Keywords. Deep Learning; Convolutional Neural Network; Real-Time Smile Detection.

1. INTRODUCTION

Smile is one of the most important facial expressions to present human's feelings of happiness, satisfaction or pleasure [3]. Automatic smile detection from facial images is being widely used in many real world applications such as product rating, automatic photo capturing, distance learning, patient monitoring, etc. Although gaining much attention from the scientific community, real time smile detection from facial images captured in real world scenarios of unconstrained conditions such as lighting and background is still a difficult and challenging problem. This can be explained by the fact that to detect smile, the task of locating human faces in images and the task of classifying a face is smiling or not often require heavy computations which may take too long to reach real time performance (below 40ms, equivalent to 25 frames per seconds).

There exist many methods to detect smile from facial images [5, 7, 8, 11, 16, 23]. These methods are generally divided into two categories: (1) traditional handcrafted feature-based methods and (2) deep learning-based methods. In the first case, sliding windows are used to generate sub-regions of the image, handcrafted features are then extracted from these sub-regions. Finally, classifiers such as Support Vector Machine (SVM), Extreme Learning

Machine (ELM) are applied in each sub-region to classify it as smile or non-smile. In the second case, Convolutional Neural Networks (CNNs) are usually used to automatically propose sub-regions from input image, extract features and categorize them as smile or non-smile. Nowadays, CNN-based methods are much more popular than handcrafted feature-based methods due to its amazing performance in terms of accuracy.

In order to speed up performance of smile detection, several methods are proposed in the literature such as YOLO [17], Faster R-CNN [18], RetinaNet [14] and MobileNets [10]. Among these methods, YOLO and MobileNets are considered as the most state-of-the-art real-time object detection methods in terms of speed. For example, YOLO is capable of gaining up to 25 frames per second. In contrast, RetinaNet as claimed by the authors [14] obtains better accuracy than YOLO but having lower time performance.

In this paper, our work is inspired by the disadvantage of RetinaNet where the goal is to improve inference time performance of RetinaNet for real-time smile detection. To reach this goal, we exploit the use of RetinaNet with several CNN architectures namely BKNet [6], ResNet [9], VGG [20] and MobileNets [10]. We then propose a new method as a combination of RetinaNet and BKNet for real-time smile detection of facial images. We will show that our method reaches real-time smile detection performance while still having a good accuracy.

The rest of the paper is described as follows: We briefly review the state of the art methods in Section 2. In Section 3, we explain in detail the methods we use to experiment real-time smile detection. Our experimental results are shown in Section 4. In the final Section 5, we present our conclusions and perspectives.

2. RELATED WORKS

2.1. Handcrafted feature-based methods

Handcrafted feature-based methods detect smile based on the extraction of facial geometry information. The detected face is then fed to a classification solution to further categorize it as smile or not. Shinohara et al., 2004 [19] used Fisher Weight Map (FWM) and Higher-Order Local Auto-Correlation (HLAC) for face representation. The authors in [12] extracted the lips and cheeks from the human face by using a 6-dimensional feature vector.

Freire et al., 2009 [7] used Local Binary Patterns (LBP) as the main descriptors and SVM as classifier to detect smile. The method achieves 90% accuracy of smile detection. Other authors in [4, 15] introduced the use of Histogram of Oriented Gradients (HOG) to extract features from facial images for face recognition and smile detection. Recent works by Huang et al., 2018 [11] optimizes face detection method by adding a pre-processing step for skin color detection, edge detection and face estimation. The candidates are then passed to traditional classifier using HOG and SVM. Although this method can achieve 120fps for full HD images, its true positive rate is only 64.0%.

Gao et al., 2016 [8] proposed a semi-automated method which combines multiple features (Self-Similarity of Gradients (GSS), HOG, Raw pixel) and multiple classifiers (AdaBoost, Linear ELM). The evaluation showed that the proposed method obtains an improved performance up to 94.61% accuracy of smile detection. An et al., 2015 [1] proposed a smile detector by using ELM. The proposed ELM classifier was trained with different feature descriptors such as LBP, Local Phase Quantization (LPQ), HOG and compared with other benchmark classifiers such as Linear Discriminant Analysis (LDA), SVM. The evaluation on GENKI-4K dataset [24] showed that the ELM-based smile detector takes 90 ms for prediction time compared to 70 ms and 3600 ms of LDA and SVM classifiers respectively.

2.2. CNN-based methods

CNN-based methods use Convolutional Neural Networks (CNNs) to automatically learn high level features of images and classify them as smile or non-smile. Zhang et al., 2015 [26] used signal recognition and verification as supervision to learn expression features for smile detection. The results showed that their method is better in reducing 21% of the error rate in GENKI-4K dataset. A deep convolutional network called Smile-CNN [2] was released for smile detection with the accuracy of 92.4% and 91.8% for SVM and AdaBoost classifiers, respectively.

Zhang et al., 2018 [25] proposed acceleration methods using heatmap (for global face and facial parts) for a cascaded CNN model to speed up inference time. This method achieves 0.499s per face detected using a GeForce GTX TITAN Black. In another work, Nguyen et al., 2018 [16] introduced the use of Faster R-CNN to speed up inference time of smile detection process. Although the method gains up to 50% faster inference performance than the original Faster R-CNN with the acceptable accuracy of 84.5%, it has not yet reached the real time performance of smile detection.

Dinh et al., 2017 [6] proposed a lightweight CNN architecture named BKNet to detect smile. The evaluation showed that the proposed CNN architecture gains high smile detection accuracy (95.08%) compared to state of the art methods. Redmon and Farhadi, 2017 [17] introduced a high-speed real-time object detection method named YOLO. The method takes the entire image in a single instance and predicts the bounding box coordinates and class probabilities for these boxes. The biggest advantage of YOLO is its super speed which can process up to 25 frames per second. YOLO is currently the most state-of-art approach for real-time object detection.

3. METHODOLOGY

3.1. BKNet

Dinh et al., 2017 [6] proposed a deep CNN architecture based on VGG network [20] named BKNet (see Fig. 1). BKNet is constructed from four stacked convolutional blocks. The first three blocks consist of two convolutional layers with the number of filters being 32, 64 and 128, respectively. The last convolutional block includes three convolutional layers instead of two like previous blocks. BKNet has the window size of 3×3 and is followed by a ReLU (Rectified Linear Unit) activation function for each convolutional layer. Each convolutional block is followed by a max pooling layer with size of 2×2. At the end of network, 2 fully connected layers with 256 neurons are used and followed by a ReLU activation function. The output will be passed through the last fully connected layer including 2 neurons with softmax activation function.

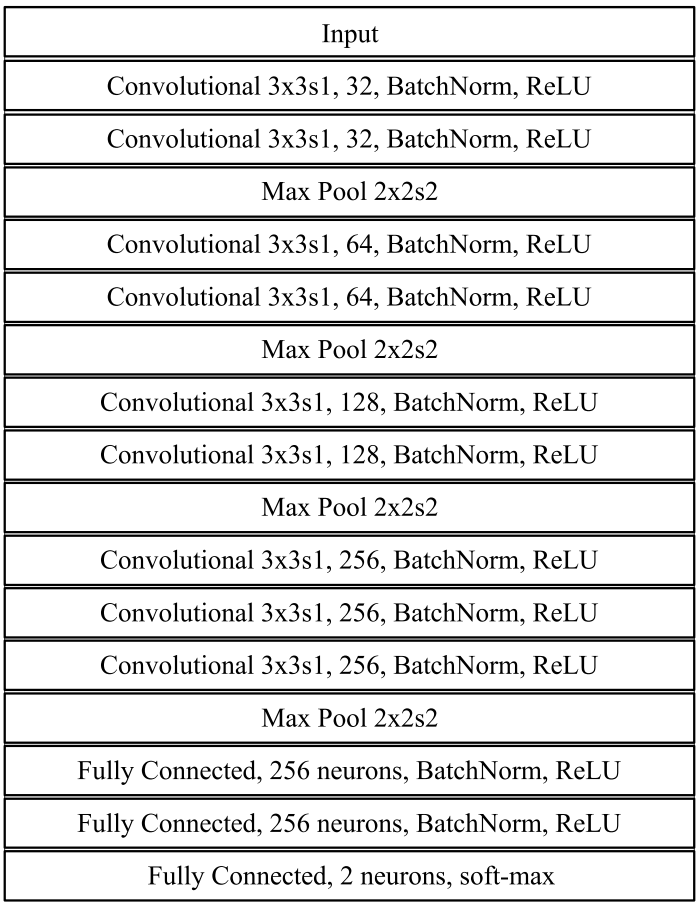
Input
Convolutional 3x3s1, 32, BatchNorm, ReLU
Convolutional 3x3s1, 32, BatchNorm, ReLU
Max Pool 2x2s2
Convolutional 3x3s1, 64, BatchNorm, ReLU
Convolutional 3x3s1, 64, BatchNorm, ReLU
Max Pool 2x2s2
Convolutional 3x3s1, 128, BatchNorm, ReLU
Convolutional 3x3s1, 128, BatchNorm, ReLU
Max Pool 2x2s2
Convolutional 3x3s1, 256, BatchNorm, ReLU
Convolutional 3x3s1, 256, BatchNorm, ReLU
Convolutional 3x3s1, 256, BatchNorm, ReLU
Max Pool 2x2s2
Fully Connected, 256 neurons, BatchNorm, ReLU
Fully Connected, 256 neurons, BatchNorm, ReLU
Fully Connected, 2 neurons, soft-max

Figure 1. BKNet architecture [6]

By using a half number of convolutional layers and less blocks than original VGG architecture [20], BKNet becomes a lightweight and powerful network structure for smile detection. Besides, batch normalization layer is added to each convolutional layer and fully connected layer to normalize the data. This allows using higher learning rate and minimizing the effects from large errors during the training process.

3.2. RetinaNet

Lin et al., 2018 [14] proposed a convolutional neural network namely RetinaNet for high speed object detection. Fig. 2 presents the architecture of RetinaNet. From the figure, RetinaNet contains a backbone network, namely Feature Pyramid Network (FPN) and two sub-networks, namely Classification Subnet and Box Regression Subnet.

Feature Pyramid Network (FPN). FPN acts as the backbone network of RetinaNet. It provides rich and multi-scale image features by combining low resolution with high resolution via a top-down pathway and lateral connection. Each level of FPN encodes a different kind of information at a different scale. At each FPN level, there are several anchors moving around the FPN feature maps. An anchor is a rectangle with different sizes and ratios

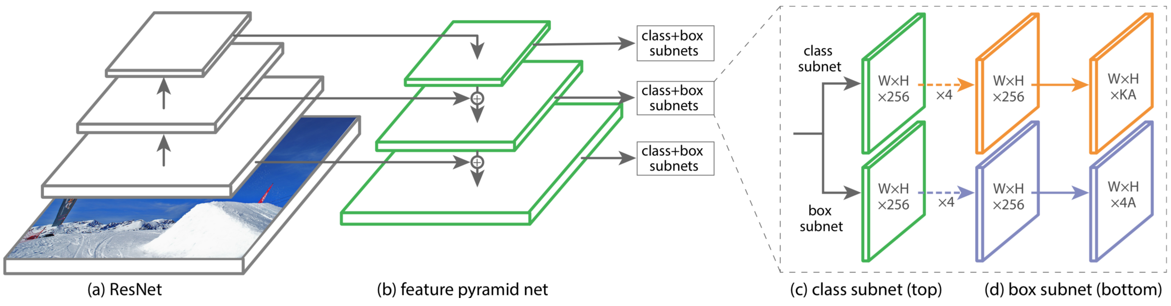

Figure 2. RetinaNet architecture [14]

presenting position of a potential object. These anchors are resized according to the scale of the FPN levels and can be duplicated on all the possible positions in the feature maps. Due to this, FPN is used for detecting objects at different scales.

Classification Subnet. Each FPN level is fed into two sub-networks in order to fully exploit every FPN level that holds a different kind of information. The first sub-network is classification subnet, it predicts the probability of object existing on each spatial position for each anchor and object classes. The sub-network is simple Fully Convolutional Network (FCN) followed to each FPN level and takes the feature map from a pyramid level as input.

Box Regression Subnet. In order to regress the offset for each anchor with the ground-truth object, each pyramid level is attached with another small FCN if an object exists. The architecture of the box regression subnet is similar to classification subnet except that it finishes with 4 linear outputs per spatial location.

3.3. Our proposed method

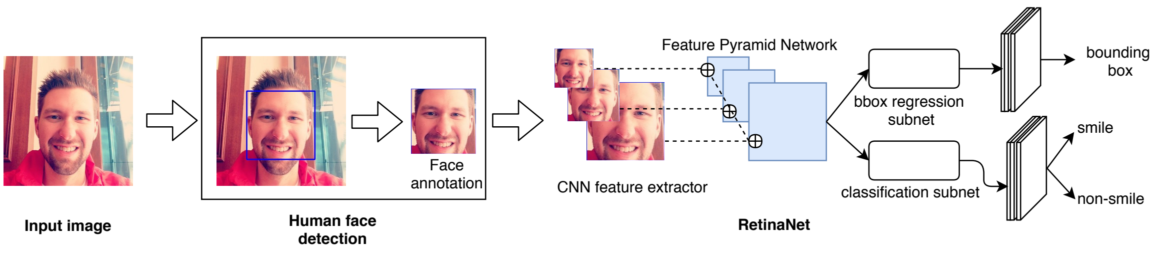

Figure 3. Pipeline of our smile detection method

Fig. 3 presents the pipeline of our proposed smile detection method. From the figure, the input image is annotated with each human face coordinate, which will be fed into RetinaNet with a CNN network as a feature extractor. Learned representations are then used as input for the subnetworks: (1) bounding box (bbox) regression subnet, and (2) classification subnet. For (1), the subnet performs a bounding box regression, presenting human face locations. Output of the bbox subnet is the anchor box to a nearby human face location if one exists. For (2), the subnet performs a classification of background and human face, predicting the probability of human face presence as smile or non-smile at each position for each of the anchors.

By analyzing the architecture of RetinaNet, we realize that its current backbone network (FPN) is built on top of ResNet architecture [9], which is a very complex CNN with many convolution modules and each module has many convolutional layers. From this, the idea behind our method is to find a simple network to replace ResNet for improving time performance of RetinaNet while still keeping a good accuracy of real-time smile detection. By doing the research study, we select the two networks (BKNet and VGG) to perform this idea. We chose VGG [20] since it is the state-of-the-art detection and classification method in terms of accuracy and BKNet since it is proved by the authors [6] to be a powerful and lightweight network structure for smile extraction and classification. By the experiments in the next section, we will show that BKNet should be the best choice to implement our idea.

4. EXPERIMENTS

4.1. Dataset

GENKI-4K [21]. This dataset contains facial images captured in various real life conditions, contexts, and backgrounds. All the images are manually labeled into 2 classes: smile and non-smile. There are in total 4,000 images in the dataset in which 2,162 images are labeled as smile and 1,838 images are labeled as non-smile. Since this dataset is gathered from various real life scenarios, there exist several unclear images which are very difficult to identify smile. Due to this, we pre-process the dataset by manually eliminating the unclear images. After the removal, we have in total of 3,699 images in which 2,008 images are labeled as smile and 1,691 images are labeled as none-smile.

UCF Selfie [13]. This dataset includes 46,836 selfie images classified in 36 different attributes such as facial gestures (smiling, frowning, mouth open), face shapes (oval, round, heart) and lighting conditions (harsh, dim). There are in total 12,207 smile and 34,629 non-smile images. Similarly to GENKI-4K dataset, we have to manually eliminate images having the confusion between smile and non-smile. After elimination process, we select 5,034 smile images and the same amount of non-smile images for training our model with a balanced dataset.

Human face annotation. As mentioned above, RetinaNet uses several anchors moving around the FPN feature maps. To feed FPN, coordinates of the anchors presenting position of potential objects need to be provided. For this, we pre-process data by creating human face annotations to the experimental images. There are many methods to generate these annotations. In this work, we apply the Haar-cascade based method for face detection proposed in [22]. In this method, the image features are extracted from input by using different rectangles. The AdaBoost classifier is then used to select only those features known to highest accuracy in face classifier.

4.2. Experimental setup

Hardware and software configuration. We perform our experiments on a HP Server with 6-core 2.40GHz Xeon E5-2620 v3 processor, 32GB RAM. In order to measure detection and inference time of our models, we conduct it on a mid-range hardware configuration, the NVIDIA GTX Titan Blacks (a graphics card with performance equivalent to GTX 1060, each has 2880 CUDA cores running at 889MHz and 6GB RAM at 7GHz). The experiment is performed with TensorFlow and Python 2.7 running on Debian 9.

Model configuration. With RetinaNet, we fine-tune the backbone network (FPN) on top of three networks: VGG16, ResNet50 and BKNet. For its classification subnet, we apply four 3×3 convolutional layers followed by ReLU activation function. Then, we apply sigmoid activation function to the outputs and focal loss is used as the loss function. For the bounding box regression subnet, *smooth_l1* with *sigma=3* is taken as the loss function.

Training and inference. In the training process, we do not use pre-trained weights of the backbone network, we evaluate accuracy and inference time of all models by training them from scratch. To train the models, we assume that the prior probability of all anchor boxes and the bias of convolutional layers of classification subnet is 0.01. The weight decay is 0.0001 and the initial learning rate is 10^{-5} with Adam optimizer for 10k iterations in 50 epochs. The inference step involves forwarding an image through the network. We decode box predictions from 1k top scoring predictions per FPN level with a thresholding detector equal to 0.05. The highest predictions from all FPN levels are merged with a threshold of 0.5 to yield the final detections.

Accuracy measurement. We define accuracy to measure the correctness of the classification process as follows

$$accuracy = \frac{\text{number of } \textbf{correctly} \text{ detected images}}{\text{total number of images}}.$$

In our experiment, we consider a "correctly detected image" based only on the ground-truth label (smile or not smile) because we found out that the network generally performs correct face boundary detection.

Inference time measurement. Our models' inference time measurement is started after the model is fully loaded. Test image is then fed into the network as input. We stop the timing measurement after the network produces a probability output for smile/non-smile classification. We use `time()` method from `time` module in Python standard library to perform this timing measurement. This function provides timing precision upto 1 microsecond on Linux platforms.

4.3. Results and discussions

Table 1 shows the accuracy and inference time of our studied smile detection methods with test data after training the models in 50 epochs on both GENKI-4K and UCF Selfie datasets with a mid-range hardware configuration (GTX TITAN Black). Bold values indicate our proposed method in this paper. We chose to experiment our methods on a mid-range device since this affect a larger proportion of users than high-end GPUs.

It can be seen from the table that the combination of RetinaNet with MobileNets and BKNet are able to reach real-time performance on both datasets, while with VGG16, although having a high accuracy, it has not yet reached real-time performance of smile detection. We do not compare our result with the method of Dinh et al., 2017 [6] (95.08% accuracy on GENKI-4K) since BKNet is proposed for classification problem of smile or non-smile images

Table 1. Accuracy and inference time of studied smile detection methods

Method	GENKI-4K		UCF Selfie	
	Accuracy (%)	Inference (ms)	Accuracy (%)	Inference (ms)
RetinaNet + ResNet50	93.2	44	91.34	53
RetinaNet + VGG16	95.14	43	96.15	67
RetinaNet + MobileNets	87.12	34	88.90	39
RetinaNet + BKNet	**94.04**	**36**	**95.19**	**38**
YOLO	93.17	31	94.23	35

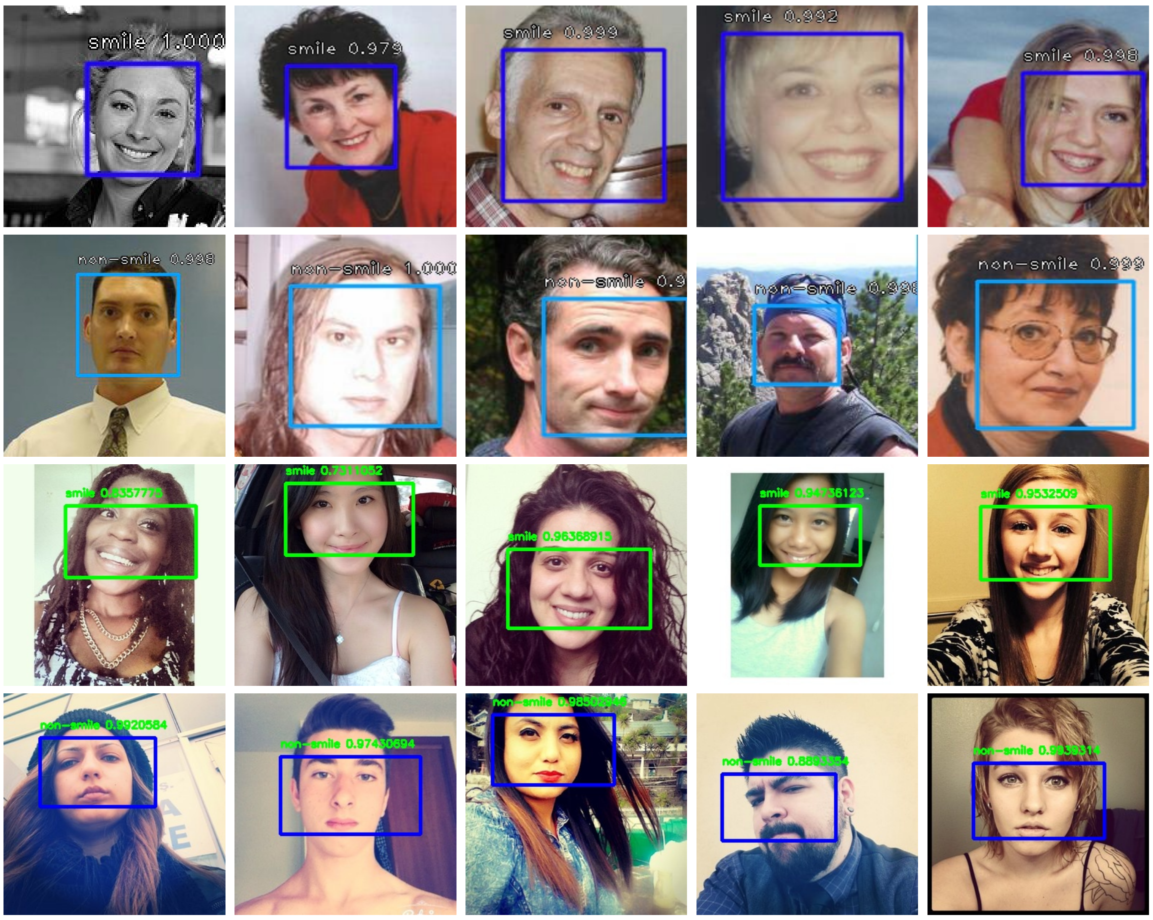

Figure 4. Sample detection results of our method on GENKI-4K (the first two rows) and UCF Selfie (the last two rows)

only, while our method is for detection problem which includes a boundary box of the smile or non-smile face.

It is noticable that although YOLO obtains fastest performance on both datasets (31ms on GENKI-4K and 35ms on UCF Selfie), it has lower accuracy than our method of RetinaNet and BKNet (93.17% vs. 94.04% on GENKI-4K and 94.23% vs. 95.19% on UCF Selfie). This highlights the advantage of our proposed method as a high quality real-time smile detector with better accuracy than YOLO on both tested datasets.

Fig. 4 shows some qualitative results of our proposed method on GENKI-4K dataset (the first two rows) and UCF Selfie dataset (the last two rows). The rectangle describes the detected region on each image with the corresponding score. From this figure, our method is able to detect smile or non-smile accurately from images of various facial shapes (oval, round, heart) with different lighting conditions (harsh, dim).

5. CONCLUSION AND PERSPECTIVES

In this paper, we studied several combinations of convolutional neural networks (VGG16, MobileNets, BKNet) as feature extractors with the high-speed object detection framework (RetinaNet) for real-time smile detection of facial images. The experiments on the two datasets (GENKI-4K, UCF Selfie) with a mid-range hardware device (GTX TITAN Black) show that our proposed method of RetinaNet with BKNet helps in enhancing both accuracy and inference time of the original RetinaNet to reach real-time performance of smile detection. Compared with the state-of-the-art object detection framework (YOLO), our method has lower inference time, but better accuracy of smile detection.

Several research directions can be taken into account to continue this work. Firstly, more data can be used to train the model for gaining better detection accuracy. Secondly, the number of blocks and layers of the networks (RetinaNet and BKNet) can be further fine-tuned to get better accuracy yet faster performance. Last but not least, more experiments could be performed on more devices to examine the scalability of our method.

REFERENCES

[1] L. An, S. Yang, and B. Bhanu, "Efficient smile detection by extreme learning machine," *Neurocomputing*, vol. 149, pp. 354–363, 2015.

[2] J. Chen, Q. Ou, Z. Chi, and H. Fu, "Smile detection in the wild with deep convolutional neural networks," *Machine Vision and Applications*, vol. 28, no. 1-2, pp. 173–183, 2017.

[3] F. De la Torre and J. F. Cohn, "Facial expression analysis," in *Visual Analysis of Humans.* Springer, 2011, pp. 377–409.

[4] O. Déniz, G. Bueno, J. Salido, and F. De la Torre, "Face recognition using histograms of oriented gradients," *Pattern Recognition Letters*, vol. 32, no. 12, pp. 1598–1603, 2011.

[5] O. Déniz, M. Castrillon, J. Lorenzo, L. Anton, and G. Bueno, "Smile detection for user interfaces," in *International Symposium on Visual Computing.* Springer, 2008, pp. 602–611.

[6] V. S. Dinh, T. B. C. Le, and P. T. Do, "Facial smile detection using convolutional neural networks," in *Knowledge and Systems Engineering (KSE), 2017 9th International Conference on.* Hue, Viet Nam, 2017, pp. 136–141.

[7] D. Freire, M. C. Santana, and O. Déniz-Suárez, "Smile detection using local binary patterns and support vector machines." in *VISAPP (1)*, 2009, pp. 398–401.

[8] Y. Gao, H. Liu, P. Wu, and C. Wang, "A new descriptor of gradients self-similarity for smile detection in unconstrained scenarios," *Neurocomputing*, vol. 174, pp. 1077–1086, 2016.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* Las Vegas, NV, USA, June 2016, pp. 770–778.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[11] D.-Y. Huang, C.-H. Chen, T.-Y. Chen, J.-H. Wu, and C.-C. Ko, "Real-time face detection using a moving camera," in *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA).* Krakow, Poland, May 16–18, 2018, pp. 609–614.

[12] A. Ito, X. Wang, M. Suzuki, and S. Makino, "Smile and laughter recognition using speech processing and face recognition from conversation video," in *2005 International Conference on Cyberworlds (CW'05).* Singapore, Singapore, Nov. 23-25, 2005, pp. 8–pp.

[13] M. M. Kalayeh, M. Seifu, W. LaLanne, and M. Shah, "How to take a good selfie?" in *Proceedings of the 23rd ACM International Conference on Multimedia*, ser. MM '15. New York, NY, USA: ACM, 2015, pp. 923–926. [Online]. Available: http://doi.acm.org/10.1145/2733373.2806365

[14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence, KIEM TRA LAI CAC THONG TIN TL NAY*, 2018.

[15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[16] C. C. Nguyen, G. S. Tran, T. P. Nghiem, N. Q. Doan, D. Gratadour, J. C. Burie, and C. M. Luong, "Towards real-time smile detection based on faster region convolutional neural network," in *Multimedia Analysis and Pattern Recognition (MAPR), 2018 1st International Conference on.* Ho Chi Minh City, Viet Nam, 2018, pp. 1–6.

[17] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, Honolulu, HI, USA, July 21–26, 2017, pp. 6517–6525. [Online]. Available: https://doi.org/10.1109/CVPR.2017.690

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[19] Y. Shinohara and N. Otsuf, "Facial expression recognition using fisher weight maps," in *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.* Seoul, South Korea, South Korea, May 19, 2004, pp. 499–504.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015. [Online]. Available: arXiv.org⟩cs⟩arXiv:1409.1556

[21] http://mplab.ucsd.edu, "The MPLab GENKI Database, GENKI-4K Subset."

[22] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001 IEEE Computer Society Conference on, CVPR*, vol. 1. IEEE, 2001, pp. I–I.

[23] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[24] J. Whitehill, G. Littlewort, I. Fasel, M. Bartlett, and J. Movellan, "Toward practical smile detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 11, pp. 2106–2111, 2009.

[25] H. Zhang, X. Wang, J. Zhu, and C.-C. J. Kuo, "Accelerating proposal generation network for fast face detection on mobile devices," in *2018 25th IEEE International Conference on Image Processing (ICIP).* Athens, Greece, Oct. 7–10, 2018, pp. 326–330.

[26] K. Zhang, Y. Huang, H. Wu, and L. Wang, "Facial smile detection based on deep learning features," in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR).* Kuala Lumpur, Malaysia, Nov. 3–6, 2015, pp. 534–538.

Received on November 22, 2018
Revised on January 24, 2019