A Lightweight Facial Emotion Recognition System Using Network Optimization Methods

Erick C. Valverde, Angela C. Caliwag, Wansu Lim
Kumoh National Institute of Technology

Abstract

Facial emotion recognition system has been recently developed for more advanced applications of face detection. The FER system classifies the human expression in various categories such as angry, disgust, fear, happy, sad, surprise, and neutral. Conventional FER systems have issues with low accuracy and high resource requirements. In order to increase the accuracy, an extreme version of Inception V3, known as Xception, is leveraged in this paper. To enable low resource requirements, a 68-landmark face detector from Dlib is used. Moreover, to develop a lightweight FER model, different network optimization methods are applied to Xception. The optimization methods used are pruning and quantization to support lower computational costs and reduce memory usage, respectively. Furthermore, to increase the inference speed of the FER system, a deep learning (DL) compiler is used to implement advanced optimization techniques to the model. The objectives of these optimization methods are experimentally demonstrated by the proposed lightweight FER system as compared to VGG-Net and ResNet50. Hence, the proposed system can be used to realize an efficient FER system in real-time inference.

Ⅰ. Introduction

Emotion recognition based on artificial intelligence (AI) has been recently developed to realize a better human-computer interaction. It is the process of identifying human emotions by analyzing facial expressions [1], decoding voice patterns [2], monitoring eye movements [3], or examining brain signals [4]. Since humans have cultural differences or distinct ways of expressing emotions, AI-based emotion recognition has long been a broad area of study. That is, the challenges of human diversity make it harder for computers to draw accurate conclusions. Nonetheless, researchers have been exerting efforts to alleviate the issues with emotion recognition by utilizing different supervised learning task algorithms. This could increase the potential of creating a more accurate emotion recognition algorithm based on human interactions with computers through visual, sounds, or neurological signals.

One of the most prominent fields of study nowadays regarding emotion recognition is facial emotion recognition (FER). FER system refers to an emotion recognition process that analyzes human face expressions to identify a specific emotion. As shown in Fig. 1, this system involves two tasks: 1) face detection task and 2) facial emotion classification task. First, the human face is detected from the image acquired. Second, the detected face is analyzed by an FER algorithm to classify which emotion it displays. The human emotions are commonly categorized as follows: 1) angry, 2) disgust, 3) fear, 4) happy, 5) sad, 6) surprise, and 7) neutral. These seven emotions are accountable for the complex implementation of FER which typically provide a various range of accuracy. For instance, [5] obtained the highest accuracy in happiness, while the lowest accuracy is observed in fear, which results to a low overall accuracy of 76%. [6] explored different

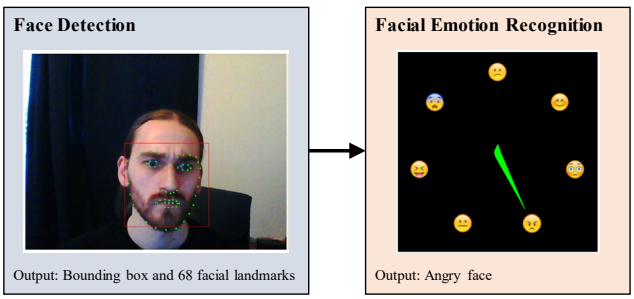


Fig. 1. Demonstration of an FER system [9].

supervised classification task algorithms for FER using K-nearest neighbor (KNN) and artificial neural network (ANN), separately. But both KNN-based and ANN-based FER methods lead also to low accuracy results (i.e., 54.16% and 66.66%, respectively) due to the corresponding low accuracies from different emotion categories. To resolve this issue with the accuracy, deep learning (DL)-based methods are proposed to improve the overall accuracy of FER algorithms up to 90% [7]. By this, researchers focus more on developing deeper networks or known as convolutional neural networks (CNNs) to find more complex features and increase the accuracy of FER system. The complexity of a CNN model refers to millions of connections within an architecture. It aims to search for significant facial patterns that can bring even up to more than 95% accuracy to FER system [8]. However, the inference speed and computational costs of FER system are often disregarded in exchange for higher accuracy results.

To cope with the issues of CNN models, some researchers

TABLE I
Comparison of Face Detection Methods [15]

Metrics / Methods	Haar	HOG	LBP
Detected Frames	653484	772954	503516
Detected Faces	652451	772954	503350
TPR (%)	78.23	**92.68**	60.37
FNR (%)	21.76	7.31	39.64

proposed to develop lightweight FER models. A lightweight FER model refers to an FER model with reduced resource requirements and faster inference speed. Several studies create transfer learning-based lightweight FER models by using state-of-the-art pre-trained DL models like VGG-Face [9], VGG16 [10], and Deep Face Convolutional Neural Network (CNN) [10]. However, their studies benchmark only which pre-trained models can provide both high accuracy and less complex architecture as compared to other existing models. That is, the original structure of the pre-trained models is not reduced, and so, no relative improvement can be observed in the results. To realize a novel lightweight FER model, several optimization methods can be used. An optimization method removes unimportant and redundant connections within a CNN model's architecture, albeit at the risk of a significant reduction in accuracy. The commonly used types of optimization methods are: 1) pruning, 2) quantization, and 3) DL compiler. Pruning refers to effective search and elimination of unimportant and redundant connections within the CNN model to reduce hardware requirements [11]. On the other hand, quantization refers to the conversion of the model from higher to lower precision format to support faster computation [12]. Meanwhile, DL compiler targeted the hardware architecture of the model to maximize the performance of the available hardware, at less computational costs, during DL inference [13]. These three optimization methods can be used to build a lightweight FER model that can provide better inference performance over existing pre-trained models.

In this paper, we proposed a lightweight FER model by network optimizations. The proposed method aims to have an efficient FER system by reducing the complexity of the DL-based FER model. The application of several optimization methods can optimize both the software and hardware components of the FER system. The optimization method will be carefully implemented to prevent any potential model accuracy loss. The contributions in this paper are summarized as follows:

1) To reduce computational costs, pruning is applied to remove unimportant and redundant connections within the architecture of the FER model.
2) To reduce the memory usage and increase inference speed of FER system, quantization is used to save the lightweight FER model at lower precision format.
3) To further increase inference speed while maintaining the reduced computational costs, DL compiler will be used to redesign the FER model to leverage with the available hardware of the device.

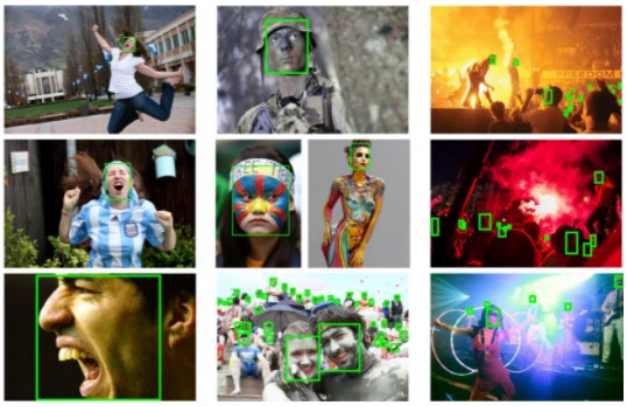


Fig. 2. Samples of WIDER FACE dataset.

The remainder of this paper presents in detail the development of FER system, different network optimization methods, the proposed methodology, and conclusion.

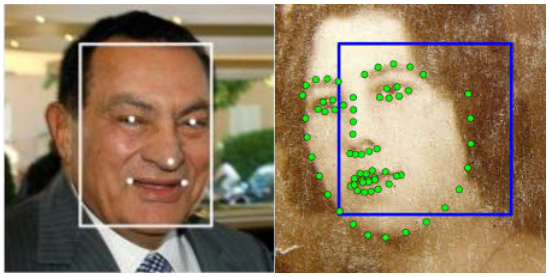
Ⅱ. Related Works

In this section, the development of facial emotion recognition (FER) system is discussed in detail. This is followed by the discussion of different network optimization methods that can be used to realize better inference performance in the FER system.

A. Development of Facial Emotion Recognition System

The current FER systems are divided into two tasks: 1) face detection task and 2) facial emotion classification task. In the FER system, the human face is detected and fed to an algorithm to analyze patterns and classify facial expressions. The emergence of state-of-the-art face detection algorithms led to the development of FER systems.

The traditional face detection algorithms like Viola-Jones detector [14] achieves real-time face detection of human using different feature descriptors [15], such as Haar-like features, histogram of oriented gradients (HOG), and linear binary pattern (LBP), to extract features and output the coordinates of the bounding box of a face. On one hand, Haar-like features method uses different rectangular filters to check the presence of a face. It has an integral image method and cascading classifiers to quickly calculate the sum of pixel values, and ignore background and non-face objects, respectively. On the other hand, HOG method calculates the histograms of gradients to check edges and corners from the images for extracting useful features. Lastly, LBP method checks textures by comparing pixels and generating binary patterns. It is used for various computer vision tasks due to its ability to check illumination invariance from an image. As shown in Table I, HOG obtained the highest true positive rate (TPR) while the highest false negative rate (FNR) is observed in LBP. Although the traditional methods are widely used because of its low computational costs, it is limited in practical applications due to its relatively low accuracy results. To address this issue, DL is used to create more advanced methods and extract meaningful features that can be used in various applications like face and object recognition. One of the most popular DL-based face detection models is multi-



(a) (b)

Fig. 3. Comparison of (a) MTCNN and (b) DAN.

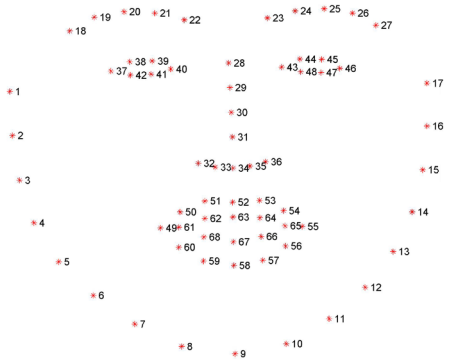


Fig. 4. Representation of 68 facial landmarks based on iBUG 300-W dataset annotations.

task cascaded convolutional neural network (MTCNN) [16]. MTCNN is composed of three cascaded CNNs, namely P-Net, R-Net, and O-Net, to provide the coordinates of bounding boxes and facial landmarks (i.e., eyes, nose, and two corners of the mouth) of the human face. As shown in Fig. 2, it solves issues with multiple posture variations, various human expressions, and lighting conditions, as tested on WIDER FACE dataset [17]. Another multi-stage CNNs for face alignment and detection is deep alignment network (DAN) [18]. DAN is a robust method that uses landmark heatmaps and feature images that transfers the landmark information between stages. As shown in Fig. 3, DAN generates a total of 68 facial landmarks, while MTCNN only generates 5. Thus, DAN can be used more effectively in complex application like in FER systems. However, since both MTCNN and DAN are built with multiple stages of CNNs, it requires high computational costs, which yield to low inference speed during real-time implementation. In this paper, a pre-trained 68-facial landmark detector trained in iBUG 300-W dataset (see Fig. 4) by Dlib [19] is used. The 68-facial landmark detector provides similar output like DAN but with reduced computational costs. It can be used to enable faster inference speed of FER system.

In line with the advancement of DL-based models, there are several facial expression or FER models that use CNN architecture to realize higher accuracy results. The traditional FER models [6] use K-nearest neighbor (KNN) and shallower network like artificial neural network (ANN). Both are used to classify human expression or emotion based on seven categories: 1) angry, 2) disgust, 3) fear, 4) happy, 5) sad, 6) surprise, and 7) neutral. These categories contribute to the complexity of building an FER model in addition to human diversity. Mostly, each emotion has huge similarity to other categories, which make it difficult to distinguish from other types of emotions. As a result, a non-parametric algorithm, such as a KNN-based model, cannot be utilized to effectively categorize the wide range of human emotions because it simply calculates and analyzes the distance between a sample and seven different emotional categories. Similarly, a shallower network, like an ANN-based model, is not efficient to extract significant features of several human emotions. The accuracy of ANN-based model reached up to 60% only, which is 10% higher than KNN-based model. For this reason, researchers focus more on developing FER models with deeper networks to learn more useful features. Deeper networks are generally leveraged using CNN architectures. It consists of millions of connections to extract and learn useful features that can be used to classify certain emotion based on the human face. [7] proves that increasing the depth of a network can improve the accuracy results of FER models up to 90%. The number of parameters within a CNN architecture is increased by applying more layers and using smaller kernels, which can learn more complex patterns. However, DL-based FER models suffer from low inference speed issue due to high computational cost requirement. To solve this issue, several researchers exerted efforts on developing lightweight FER models. A lightweight FER model is an optimized algorithm to support low computational cost and high inference speed. Some studies benchmark commonly used pre-trained models for FER task by transfer learning. They compare VGG-Face [9], VGG-16 [10], and Deep Face CNN [10] in terms of computational and memory costs. Although the existing pre-trained models can provide high generalizations, their requirements remained high. In addition, deeper networks can easily have problem of overfitting due to considerable number of parameters. Hence, key to lightweight FER models is to significantly reduce the parameters without affecting the accuracy. Most of the parameters are observed in the last fully connected layers of the CNN models. For instance, in the VGG-Net [20], the last fully connected layers include 90% of the total parameters. To reduce the parameters, Inception V3 [21] replaces the last fully connected layers with Global Average Pooling operation, which takes the average of the elements in feature image. To further reduce the parameters, an extreme version of Inception V3, known as Xception [22], is developed. Xception utilizes deep residual learning and depth-wise separable convolutions to separate the feature extraction and composition processes. Most of the efficient FER models developed nowadays are based on Xception architecture [23]. Although the computational cost of FER system is reduced by leveraging with Xception architecture, further optimizing this is still a challenge. In this paper, we proposed a lightweight FER model by applying different network optimization methods.

B. Network Optimization Methods

There are several network optimization methods that can effectively reduce CNN models, realize faster inference speed, and lower computational costs. In this paper, we used different optimization methods, such as pruning, quantization, and DL compiler, to reduce the complexity and requirements of Xception model for the FER task.

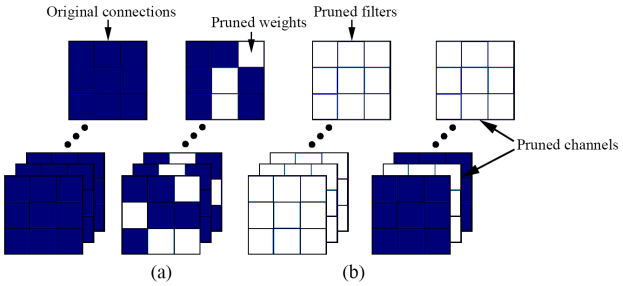


Fig. 5. Comparison of (a) unstructured pruning and (b) structured pruning. Note that unstructured pruning prunes weights while structured pruning prunes channels and/or filters.

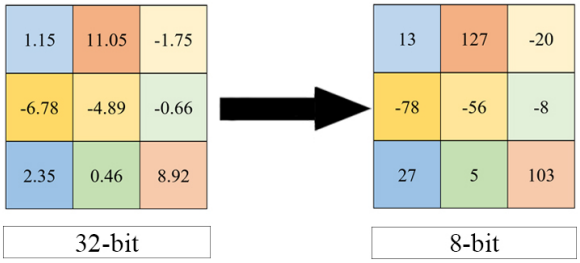


Fig. 6. General process of quantization. Sample of converting 32-bit floating-point number to 8-bit integer.

Pruning refers to an effective search and elimination of unimportant and redundant connections within the CNN model [11]. The weights, filters, and channels are the connections in the CNN model that build up its enormous complexity. By applying pruning method, the complexity of the model can be reduced. As a result, the model's memory and computational costs will be reduced during implementation. Pruning can be categorized as unstructured and structured. Unstructured pruning prunes redundant weights of a CNN model in fine-grained nature, as shown in Fig. 5(a). It effectively reduces the computational costs of the model. On the other hand, structured pruning refers to pruning of insignificant channels and/or filters of a CNN model in coarse-grained nature, as shown in Fig. 5(b). It can leverage the hardware parallelism of the device in expense of the substantial reduction in the storage footprint of the model. For comparison, unstructured pruning is more flexible than structured pruning in searching for optimized pruning structure. So, the former is easier to implement and typically achieves higher compression rate without affecting the accuracy results. In this paper, we used unstructured weight pruning to reduce the computational costs of Xception model while securing zero or negligible accuracy loss.

Meanwhile, **quantization** refers to the conversion of the bit representation of each weight from the CNN architecture into lower precision format [12]. Since CNN models are stored in 32-bit floating-point (FP32), it can be converted to either FP16 or even lower integer (INT) like INT8 and INT4 by using quantization. As a result, the memory usage is reduced, and the inference speed is increased. Quantization can be categorized as uniform and non-uniform. A uniform quantization has an equal step size or quantization level, which provides significant acceleration to the model. On the other hand, a non-uniform quantization does not have an equal

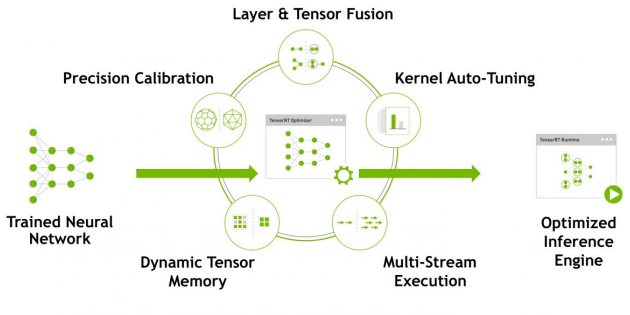


Fig. 7. Advanced performance optimizations using NVIDIA TensorRT.

quantization level, but usually achieves higher compression rate. The general process of quantization is shown in Fig. 6. In this paper, we used a uniform quantization to enable high inference speed to the FER system.

Lastly, a **DL compiler** is used to leverage with the hardware architecture of the model [13]. It can optimize models by redesigning and compiling its architecture to easily access hardware optimizations for faster inference. It includes advanced performance optimization, varied computation graph optimization, tensor optimization, and half precision format support. There are various DL compilers that can be used to assist inference acceleration, such as TensorFlow-Lite (TFLite), Alibaba Mobile Neural Network (MNN), Open Neural Network Exchange (ONNX), and NVIDIA TensorRT. These DL compilers have similar processes of optimizing CNN models. An example is shown in Fig. 7, which represents the advanced performance optimizations using NVIDIA TensorRT. In this paper, a DL compiler is utilized to compile the Xception model into an efficient end-to-end framework in order to improve the inference performance of the FER system.

Ⅲ. Proposed Lightweight FER System

The schematic diagram of the proposed lightweight FER system is shown in Fig. 8. As shown in the figure, there are three main processes in the proposed lightweight FER system: 1) face detection of the input image, 2) application of network optimization to the FER model, and 3) facial emotion classification.

First, for the face detection of the input image, we used a pre-trained landmark detector from Dlib library to identify 68 key points or facial landmarks marked at certain x and y coordinates in the human face. The key points localize the region around the human face, such as the eyebrows, eyes, nose, mouth, chin, and jaw. The Dlib 68-landmark face detector also provide the coordinates of a bounding box enclosing the human face. It is trained on the iBUG 300-W dataset, which is built by the Intelligent Behavior Understanding Group (iBUG) at Imperial College London. As shown in Fig. 9, the iBUG 300-W dataset contains "in-the-wild" images collected from the internet and corresponding 68 facial landmarks and bounding box annotations. It has more than 4000 static images, each having single face of various poses, expressions, and illuminations. The trained model is tested on 300-W test dataset collected also "in-the-wild" images, 300 indoor and 300 outdoor, each image has multiple

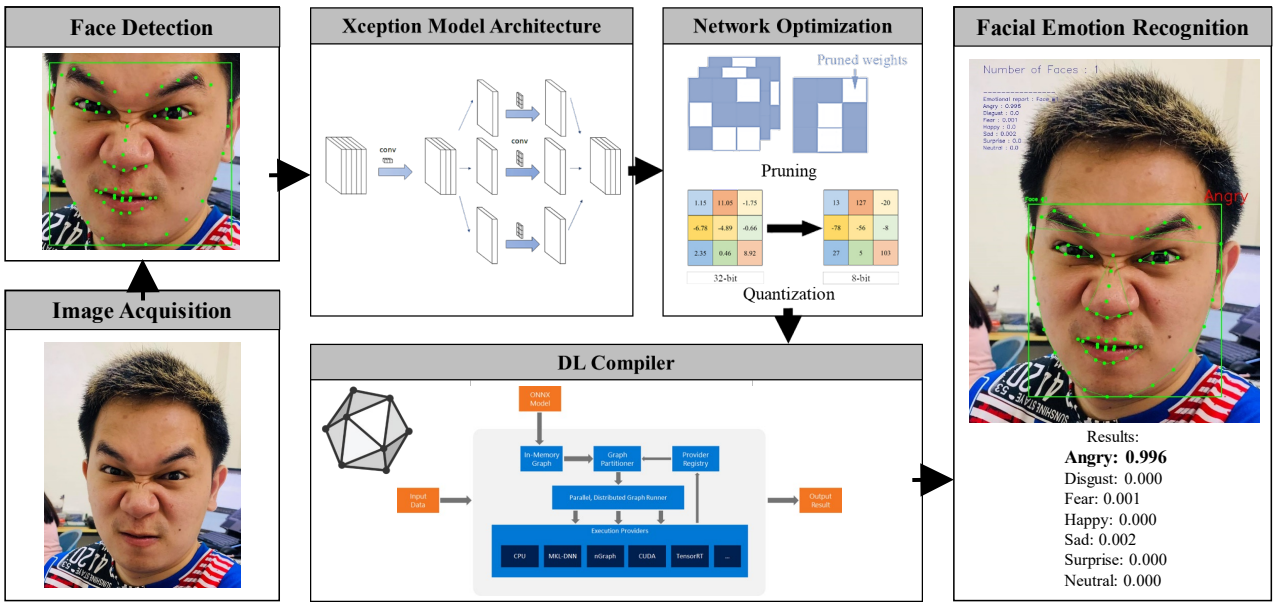


Fig. 8. Schematic diagram of the proposed lightweight FER system.

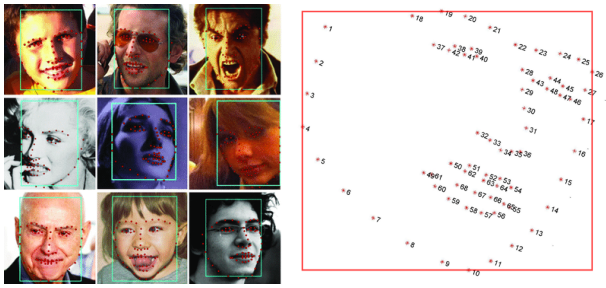


Fig. 9. Samples of iBUG 300-W dataset with annotations.

Fig. 10. Samples of 300-W test dataset (outdoor – left, indoor – right).

faces and huge facial variations, see samples in Fig. 10. The pre-trained face detection model has an efficient inference performance in detecting faces based on 300-W test dataset. For this reason, the Dlib 68-landmark face detector is used in this paper to extract useful patterns from several types of emotions in human faces. It can provide an efficient calculation to distinguish one specific emotion to another for an accurate FER system.

Second, for the application of network optimization to the FER model, we used pruning, quantization, and DL compiler to the Xception model to develop lightweight FER system. An Xception model has better accuracy and relatively smaller parameters than the existing CNN models (e.g., VGG-Net and ResNe50) as trained and tested on FER2013 dataset (see

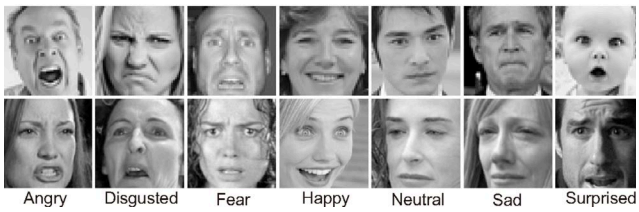


Fig. 11. Samples of FER2013 dataset.

samples in Fig. 11). However, it still has issues with high computational costs, high memory usage, and low inference speed during FER implementation. In this paper, we utilized several optimization methods that targeted both software and hardware components of the FER system to solve the issues.

Initially, we used unstructured pruning method to reduce the total number of non-zero parameters of Xception model by removing redundant and insignificant parameters. This optimization method will significantly reduce the computational costs of the model. Typically, the compression ratio of the pruned CNN architecture relative to the uncompressed is given by

$$R = \frac{\sum_{i=1}^{L} U_{w,f,c}{}^{(i)}}{\sum_{i=1}^{L} P_{w,f,c}{}^{(i)}}, \quad (1)$$

where R is called the compression ratio between uncompressed (U) and pruned models (P); $U_{w,f,c}{}^{(i)}$ and $P_{w,f,c}{}^{(i)}$ represents the total number of weights (w), filters (f), and/or channels (c) up to layer L of the uncompressed and pruned CNN architecture, respectively.

Then, we used a uniform quantization method to reduce the memory usage and increase the inference speed of the Xception model by converting its high precision format from FP32 to FP16 or INT8 low precision format. General weights quantization is computed by

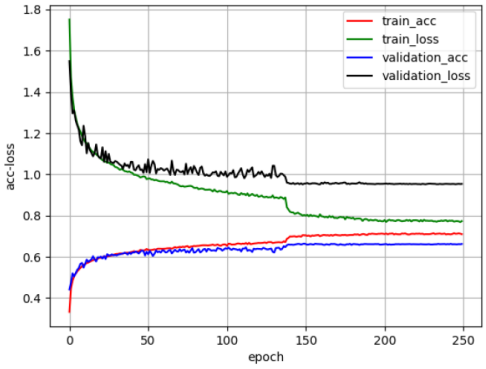


Fig. 12. Accuracy and loss curve of [23] based on FER2013 dataset.

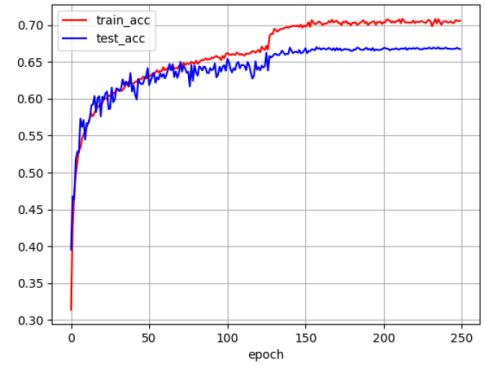


Fig. 13. Train and test accuracy of [23] based on actual test set.

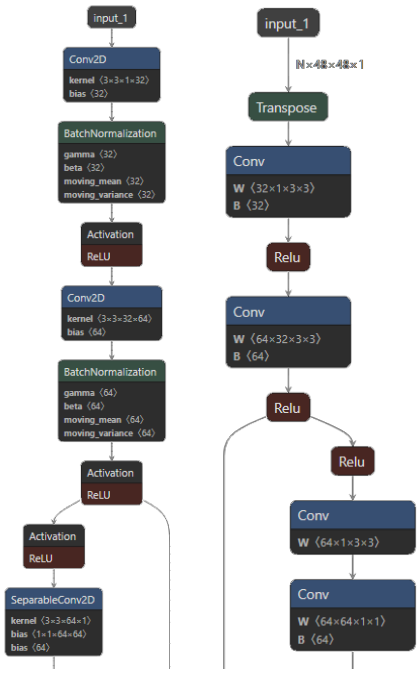


Fig. 14. Representation of the proposed lightweight FER model in (a) Keras format and (b) ONNX format.

$$W_b = round(S_b W_{fp}) \in \left[-\left(\frac{2^b-1}{2}\right), \left(\frac{2^b-1}{2}\right)\right] \quad (2)$$

$$s_b = \frac{2^b-1}{2M_W} \quad (3)$$

$$M_W = max\left(abs(W_{fp})\right), \quad (4)$$

where W_b represents the quantized weight tensor, b is the desired lower-precision format, s_b is the quantization scale factor, W_{fp} is the original weight tensor in higher-precision floating-point format, and M_W is the absolute maximum weight from W_{fp}. The quantized weight tensor is obtained by the product of the quantization scale factor and each weight from the original weight tensor rounded to the nearest integer. The quantized weights are bounded by the symmetrical dynamic range of the desired lower-precision format. This will reduce the memory usage of Xception model by more than 50% and slightly increase its inference speed.

The general problem of applying pruning and/or quantization is due to the difficult search of the compression hyperparameters in each layer of the CNN model. There are several approaches that can be used to effectively optimize the hyperparameters. One of these approaches is known as a heuristic method or manual tuning of the hyperparameter per layer of the CNN architecture. Another one is a black box optimization or Bayesian optimization method, which is an automatic hyperparameter-search approach. However, these approaches are inefficient because they rely only on repeated trial-and-error procedures to determine the ideal hyperparameters in a layerwise manner, which frequently results in a significant accuracy loss. To solve this problem, we implemented a constrained approach of pruning and quantization methods to minimize any possible accuracy degradation after optimization. This approach will not consider any compression hyperparameters to eliminate the risk of having low accuracy results.

Finally, we employed a DL compiler to further optimize and increase the inference speed of Xception model. Each framework (e.g., PyTorch, Keras, and TensorFlow) has its own distinct representation of a computation graph which often leads to a restriction in using a model from one framework to another. We solved this issue by using an Open Neural Network Exchange Format (ONNX), which is designed to standardize layer definitions of a network and support most of deep learning model formats. In this paper, a Keras framework is employed to train and optimize Xception model. Then an existing Keras to ONNX converter is utilized to reconstruct the network graph with the equivalent operators on ONNX format. This will allow a performance-focused inference optimizer, called ONNX Runtime, to automatically utilize the available hardware accelerators and runtime on the host device. Hence, improving the performance of the model. This inference engine partitions the execution graph into subgraphs and run each subgraph on the most efficient execution provider such as CUDA and TensorRT. By applying pruning, quantization, and DL compiler, a lightweight FER model is developed, as shown in Fig. 8.

Lastly, for the facial emotion classification, the proposed lightweight FER model shows better performance when implemented in real-time inference, as shown in the last diagram in Fig. 8.

TABLE II
Actual Test Results of The Proposed Lightweight FER Model

Lightweight FER Model	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Angry	**0.996**	0.000	0.001	0.000	0.002	0.000	0.000
Disgust	0.002	**0.997**	0.000	0.000	0.000	0.000	0.000
Fear	0.003	0.000	**0.839**	0.000	0.128	0.027	0.003
Happy	0.001	0.000	0.001	**0.992**	0.000	0.005	0.002
Sad	0.160	0.001	0.029	0.003	**0.414**	0.001	0.391
Surprise	0.004	0.000	0.035	0.003	0.000	**0.957**	0.000
Neutral	0.156	0.000	0.050	0.008	0.271	0.004	**0.511**

TABLE III
Comparison of Different FER Models

Model	Train set accuracy (%)	Test set accuracy (%)
VGG-Net	98.98	59.32
ResNet-50	98.87	57.48
CNN	99.7	58.9
HOG+ CNN [41]	-	61.86
Improved Inception [42]	-	66.41
Network from [43]	-	66.4
CNN-based+ Softmax [45]	-	65.03
ShallowNet [47]	-	63.49
Lightweight FER Model	**71.00**	**67.00**

TABLE IV
Hardware and Inference Performance of Different FER Models

Model	Parameters	CPU (%)	MEM (%)	FPS
VGG-Net	87566680	21.50	3.2	-
ResNet-50	25500000	49.78	2.9	-
CNN	95263	10.83	12.0	-
Lightweight FER Model	**58423**	**17.50**	**3.1**	**21**

IV. Experiment

In this section, we evaluated the performance of the proposed lightweight FER model. First, the loss and accuracy results of training the lightweight FER model is demonstrated. Then, the performance of the lightweight FER model is compared to other existing FER systems in terms of the computational costs, memory usage, and inference speed.

The Xception model is trained on FER2013 dataset for a total of 250 epochs. The model is being optimized, using the pruning and quantization methods, and trained simultaneously. Fig. 12 shows the potential change curves of the loss and accuracy values after 250 epochs of training and optimizing the model. As can be seen in the figure, the train accuracy reached 71% while the validation accuracy achieved 67% as tested on FER2013 dataset. Fig. 13 shows the actual test accuracy of the model which reached up to 67% as validated on "in-the-wild" images from the internet. The results of the proposed lightweight FER model are comparable with these figures.

Furthermore, the model trained in Keras framework is compiled into ONNX format. Fig. 14 shows the difference in architectures of the proposed model. The ONNX format of the Xception model has fewer blocks as compared to Keras. This means that some of the computation graph within the architecture is efficiently optimized. Table II shows the actual test result of the proposed lightweight FER model across different emotion categories. The proposed model accurately classified the test images based on their specific emotion. The highest accuracy is observed to angry, disgust, and happy, while the lowest accuracy is observed to sad and neutral. The comparison of the accuracy of the proposed lightweight FER model to other existing FER models is shown in Table III. As can be seen in the table, the proposed model has the highest test accuracy result reaching 67%. Other models, such as VGG-Net, ResNet-50, and CNN, have overfitting problems since their accuracy on train set is close to 100% but their resulting test set accuracy only reached about 60%. Moreover, Table IV shows the hardware and inference performance of the proposed lightweight FER model. As shown in the table, the proposed model has the least number of parameters. In addition to that, the computational cost of the model based on CPU is second only to CNN, however, the significant difference in their memory usage makes the proposed model (3.1%) superior to CNN (12%). The inference speed measured in frame per second (fps) of the proposed model is relatively high (21 fps). Hence, the proposed lightweight FER model outperforms other existing models.

V. Conclusion

In this paper, we proposed a lightweight FER system that realizes an efficient inference performance during real-time implementation. We experimentally demonstrated that the application of network optimization methods, like pruning and quantization, can effectively reduce the computational costs and memory usage of the Xception model without affecting its accuracy results. Also, employing DL compiler, like ONNX, can further increase the inference speed of the model. This is by utilizing the half precision support and TensorRT execution of the compiler. Our obtained experimental results show that the proposed lightweight FER system, with the used of Dlib 68-landmark face detector, outperforms other existing FER models specially in terms of real-time inference. Specifically, our proposed system achieved smaller number of parameters (58423 only), low CPU and memory usage (17.50% and 3.1%), and relatively high inference speed reaching 21 fps, as compared to other existing FER systems.

ACKNOWLEDGMENT

This work was supported by the Ministry of SMEs and Start-ups, S. Korea (S2829065, S3010704), and by the National Research Foundation of Korea (2020R1A4A101777511, 2021R1I1A3056900).

References

[1] H. Liu, J. Zeng and S. Shan, "Facial Expression Recognition for In-the-wild Videos," 2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020), 2020, pp. 615-618, doi: 10.1109/FG47880.2020.00102.

[2] K. Tarunika, R. B. Pradeeba and P. Aruna, "Applying Machine Learning Techniques for Speech Emotion Recognition," 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2018, pp. 1-5, doi: 10.1109/ICCCNT.2018.8494104.

[3] R. S. Soundariya and R. Renuga, "Eye movement-based emotion recognition using electrooculography," 2017 Innovations in Power and Advanced Computing Technologies (i-PACT), 2017, pp. 1-5, doi: 10.1109/IPACT.2017.8245212.

[4] B. Xue, Z. Lv and J. Xue, "Feature Transfer Learning in EEG-based Emotion Recognition," 2020 Chinese Automation Congress (CAC), 2020, pp. 3608-3611, doi: 10.1109/CAC51589.2020.9327161.

[5] Chung, K.-M., Kim, S., Jung, W. H., & Kim, Y. (2019). Development and validation of the yonsei face database (yface db). Frontiers in Psychology, 10. https://doi.org/10.3389/fpsyg.2019.02626

[6] R. Ranjan and B. C. Sahana, "An Efficient Facial Feature Extraction Method Based Supervised Classification Model for Human Facial Emotion Identification," 2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2019, pp. 1-6, doi: 10.1109/ISSPIT47144.2019.9001839.

[7] E. Pranav, S. Kamal, C. Satheesh Chandran and M. H. Supriya, "Facial Emotion Recognition Using Deep Convolutional Neural Network," 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), 2020, pp. 317-320, doi: 10.1109/ICACCS48705.2020.9074302.

[8] M. A. Ozdemir, B. Elagoz, A. Alaybeyoglu Soy and A. Akan, "Deep Learning Based Facial Emotion Recognition System," 2020 Medical Technologies Congress (TIPTEKNO), 2020, pp. 1-4, doi: 10.1109/TIPTEKNO50054.2020.9299256.

[9] J. Schwan, E. Ghaleb, E. Hortal and S. Asteriadis, "High-performance and lightweight real-time deep face emotion recognition," 2017 12th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP), 2017, pp. 76-79, doi: 10.1109/SMAP.2017.8022671.

[10] A. Atanassov and D. Pilev, "Pre-trained Deep Learning Models for Facial Emotions Recognition," 2020 International Conference Automatics and Informatics (ICAI), 2020, pp. 1-6, doi: 10.1109/ICAI50593.2020.9311334.

[11] H. Yang, S. Gui, Y. Zhu and J. Liu, "Automatic Neural Network Compression by Sparsity-Quantization Joint Learning: A Constrained Optimization-Based Approach," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 2175-2185, doi: 10.1109/CVPR42600.2020.00225.

[12] F. Tung and G. Mori, "Deep Neural Network Compression by In-Parallel Pruning-Quantization," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 3, pp. 568-579, 1 March 2020, doi: 10.1109/TPAMI.2018.2886192.

[13] Y. Cai, H. Li, G. Yuan, W. Niu, X. Tang, B. Ren, and Y. Wang, "YOLObile: Real-Time Object Detection on Mobile Devices via Compression-Compilation Co-Design," 2020 arXiv:2009.05697. [Online]. Available: https://arxiv.org/abs/2009.05697

[14] Y. Zhou, D. Liu and T. Huang, "Survey of Face Detection on Low-Quality Images," 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), 2018, pp. 769-773, doi: 10.1109/FG.2018.00121.

[15] A. Adouani, W. M. Ben Henia and Z. Lachiri, "Comparison of Haar-like, HOG and LBP approaches for face detection in video sequences," 2019 16th International Multi-Conference on Systems, Signals & Devices (SSD), 2019, pp. 266-271, doi: 10.1109/SSD.2019.8893214.

[16] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1499-1503, 2016.

[17] S. Yang, P. Luo, C. C. Loy and X. Tang, "WIDER FACE: A Face Detection Benchmark," In Proc. of IEEE Conf. CVPR, 2016.

[18] M. Kowalski, J. Naruniec and T. Trzcinski, "Deep Alignment Network: A Convolutional Neural Network for Robust Face Alignment," 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 2034-2043, doi: 10.1109/CVPRW.2017.254.

[19] Y. Liu, Z. Xu, L. Ding, J. Jia and X. Wu, "Automatic Assessment of Facial Paralysis Based on Facial Landmarks," 2021 IEEE 2nd International Conference on Pattern Recognition and Machine Learning (PRML), 2021, pp. 162-167, doi: 10.1109/PRML52754.2021.9520746.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, arXiv:1409.1556. [Online]. Available: https://arxiv.org/abs/1409.1556

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2818-2826, doi: 10.1109/CVPR.2016.308.

[22] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1800-1807, doi: 10.1109/CVPR.2017.195.

[23] N. Zhou, R. Liang and W. Shi, "A Lightweight Convolutional Neural Network for Real-Time Facial Expression Detection," in IEEE Access, vol. 9, pp. 5573-5584, 2021, doi: 10.1109/ACCESS.2020.3046715.