Micro Expression Recognition: Multi-scale Approach to Automatic Emotion Recognition by using Spatial Pyramid Pooling Module

Lim Jun Sian, Marzuraikah Mohd Stofa, Koo Sie Min, Mohd Asyraf Zulkifley
Department of Electrical, Electronic and Systems Engineering, Universiti Kebangsaan Malaysia, Bangi, Malaysia

Abstract—**Facial expression is one of the obvious cues that humans used to express their emotions. It is a necessary aspect of social communication between humans in their daily lives. However, humans do hide their real emotions in certain circumstances. Therefore, facial micro-expression has been observed and analyzed to reveal the true human emotions. However, micro-expression is a complicated type of signal that manifests only briefly. Hence, machine learning techniques have been used to perform micro-expression recognition. This paper introduces a compact deep learning architecture to classify and recognize human emotions of three categories, which are positive, negative, and surprise. This study utilizes the deep learning approach so that optimal features of interest can be extracted even with a limited number of training samples. To further improve the recognition performance, a multi-scale module through the spatial pyramid pooling network is embedded into the compact network to capture facial expressions of various sizes. The base model is derived from the VGG-M model, which is then validated by using combined datasets of CASMEII, SMIC, and SAMM. Moreover, various configurations of the spatial pyramid pooling layer were analyzed to find out the most optimal network setting for the micro-expression recognition task. The experimental results show that the addition of a multi-scale module has managed to increase the recognition performance. The best network configuration from the experiment is composed of five parallel network branches that are placed after the second layer of the base model with pooling kernel sizes of two, three, four, five, and six.**

Keywords*—*Micro expression recognition; facial expression; spatial pyramid pooling module; multi-scale approach; deep learning

I. Introduction

According to the research from [1], [2], faces are the main human "tools" to express information in terms of emotion. Facial expression is an important means that enable humans to undergo social interaction with each other. This is because 55% of human feelings are manifested by their facial expression. For example, an observer can deduce that someone is feeling disgusting if his/her upper lip is rising upward.

Facial expression can be broken down into two categories, which are macro-expression and micro-expression. A macro-expression is an intentional facial expression, while a micro-expression is an unintentional facial expression. Benjamin et al. [3] investigated that the major differences between them are the intensity and time taken to manifest the expression. Deng et al. [4] reported both expressions are widely used as an input to various applications and the most obvious application is to estimate the hidden emotions.

On the other hand, Micro-expression (ME) is an unintentional, quick facial movement that is primarily used to express the emotions of happiness, sadness, and surprise [5]. A ME happened in a short time, usually happened in the range of 0.04s until 0.2s. Hence, it is a hard task for a human to use their bare eyes to detect the occurrence of ME. Even if a human is undergoing training to detect an ME, their average performance is only slightly better than other people who do not undergo the training process. Hence, Zhao and Li [6] showed that machine learning is proposed to aid humans in analyzing the ME to understand human's true emotions.

Machine learning (ML) can be broadly classified into traditional machine learning and deep learning. Researchers in pattern recognition tasks have frequently applied both techniques to the applications of facial expression recognition [7], human activity recognition [8], recycling system [9], and image recognition [10]. Traditional machine learning relies on a set of handcrafted features, which is then passed to a decision-making module algorithm such as decision tree, neural network, and Support Vector Machine (SVM) [11], [12]. However, it is a time-consuming task for a computer vision engineer to judge which features are the best to describe the emotions.

The deep learning methodology is different compared to the traditional machine learning approach, whereby the features of interest are obtained through iterative optimal training such as through the convolution process [10], [13]. Usually, after the feature maps have passed through a convolution process, they will undergo a pooling process. Fig. 1 shows the generalized framework of traditional machine learning and deep learning algorithms for human emotion recognition tasks.

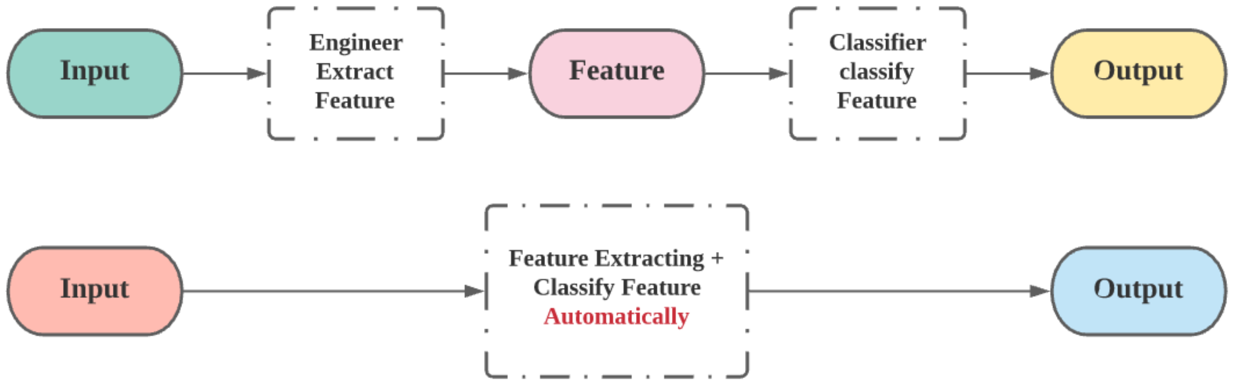
Input
Engineer Extract Feature
Feature
Classifier classify Feature
Output
Input
Feature Extracting + Classify Feature Automatically
Output

Fig. 1. Generalized Framework of Traditional Machine Learning (above) and Deep Learning (below).

Spatial Pyramid Pooling (SPP) originates from spatial pyramid matching that utilizes spatial statistical properties to represent global information for recognition purposes. Ke et al. [14] discussed the main benefit of the SPP module is it can produce constantly desired outputs without considering the input size requirement to the deep learning model. The training process of a deep learning model with multiple sizes of the image can also prevent over-fitting problems [15]. In this work, several configurations of SPP have been explored that include different numbers of layers and kernel sizes, as well as placement of the module. Besides that, the base model that has been used in this work is also compact in nature, whereby it is commonly used in tracking application, which requires fast computational model [16]. In Oh et al. [17] have surveyed different algorithms from different researchers, and their respective accuracy is summarized in Table I.

By referring to Table I, the previous works' accuracy using CASME II dataset is within the range of 40% to 60%, while for the SMIC dataset, the accuracy range is between 50% and 70%. In general, these accuracies are not satisfactory enough for real-life application. By referring to Table I, the previous works' accuracy using CASME II dataset is within the range of 40% to 60%, while for the SMIC dataset, the accuracy range is between 50% and 70%. In general, these accuracies are not satisfactory enough for real-life application. Micro-expression is a crucial set of facial cues that are extensively employed in all parts of human society. However, simple facial macro expression cues are not enough to effectively relay the real emotions. In order to resolve the issues, this study presents several variants of SPP to improve the recognition accuracy of the automated micro-expression recognition applications.

TABLE I. ACCURACY OF MICRO-EXPRESSION RECOGNITION FROM DIFFERENT PAPERS

Papers	**Features**	**Classifier**	**Accuracy (%)**	
			CASME II	***SMIC***
Huang et al. [18]	SpatioTemporal Completed Local Quantiza-tion Patterns (STCLQP)	SVM	58.39	64.02
He et al. [19]	Multi-task mid-level feature learning (MMFL)	SVM	59.81	63.15
Huang et al. [20]	Discriminative Spatiotemporal Local Radon-based Binary Pattern (STLPB-IP)	SVM	64.37	60.98
Li et al. [21]	Histograms of Image Gradient Orientation (HIGO)	SVM	67.21	68.29
Liong et al. [22]	Local Binary Pattern histograms from Three Orthogonal Planes (LBP-TOP)	SVM	46.00	54.00
Happy et al.[23]	Fuzzy Histogram of Optical Flow Orientations (HFOFO)	SVM	56.64	51.83
Le Ngo et al. [24]	LBP-TOP	SVM	49.00	58.00
Xu et al. [25]	Facial Dynamics Map	SVM	45.93	54.88
Ping et al. [26]	LBP-TOP	Group Sparse Spation-Temporal Reature Learning (GSLSR)	67.89	70.12
Zong et al. [27]	Hierarchical STLBP-IP	Kernelized Group Sparse Learning (KGSL)	63.83	60.78

Continuing from this introduction section will be Section II that presents a comprehensive overview of the basic architecture, hyperparameter function, and related layers used in modeling the CNN model. In Section III, the new improved CNN model is introduced by embedding spatial pyramid pooling into the basic model. Then, the experimental results were discussed in Section IV. Finally, the last section concludes the paper with some suggestions for future works.

II. Related Work

A. Basic Architecture of CNN

The structure of a neural network is very much similar to a human being's brain neuron. When the neuron is excited, it will deliver a chemical substance to its neighboring neuron, which will alter the state potential. If the next neuron's potential is higher than the threshold, the state will be activated and vice versa [28]. A compact deep learning structure of a convolutional neural network (CNN) mainly comprises three convolutional layers, pooling layers, and full-connected layers. The convolution process happened in convolutional layers to extract features from the input image through a sliding window operation. Then, the resultant feature maps will be passed to the pooling layers to reduce the map dimension. Fully-connected layers will be used to segregate the data into different classes [29], [30]. Fig. 2 shows the basic architecture of a CNN.

B. Convolutional Layer

A convolution operator performs a linear operation to a spatial map through a sliding window process. The process starts by applying a small number array (kernel) on the input data (tensor) to compute the product of each element of kernel and tensor for all tensor data. After that, each of the computed outputs will be summed up to form a new value in the respective position of the tensor (feature map). The whole steps will be repeated by applying multiple kernels into the tensor [31]. Based on Ma et al. [32], one kernel can extract one pattern characteristic of the input image. Fig. 3 shows the summary of a convolution process.

C. Padding

According to Rikiya et al. [31], the overlapping between the center element of the kernel with the outermost element of the input tensor should be avoided. Hence, a padding operator was introduced to enlarge the feature map dimension. There are two popular types of padding operations which are zero padding (or called the same padding) and valid padding. The process of zero padding is to make the convolution image larger by adding the zeros to its borders. The size of the output from the zero padding will be the same as the size before undergoing the convolution process [33], [34]. Fig. 4 shows the summary of a zero-padding process.

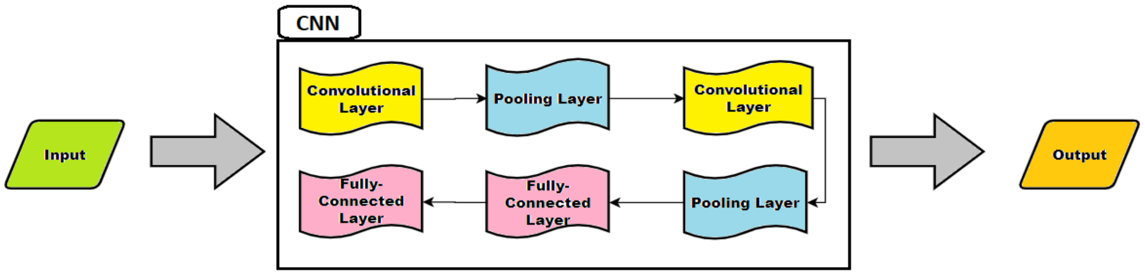

Fig. 2. Basic Architecture of CNN.

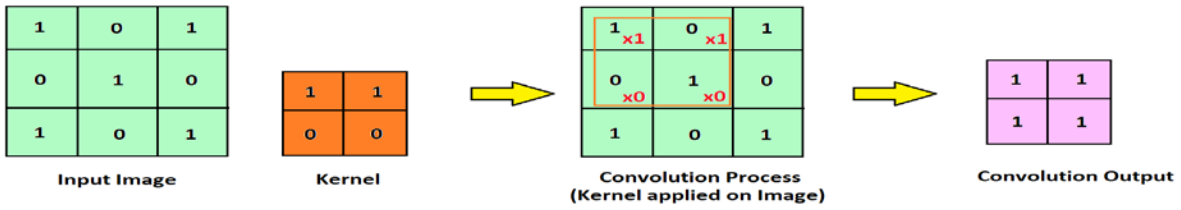

Fig. 3. Summary of Convolution Process.

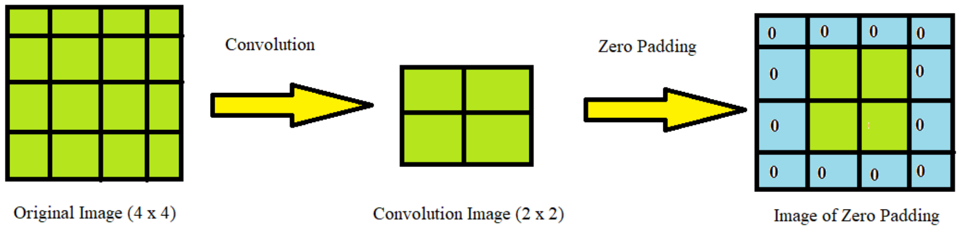

Fig. 4. Summary of Zero Padding Process.

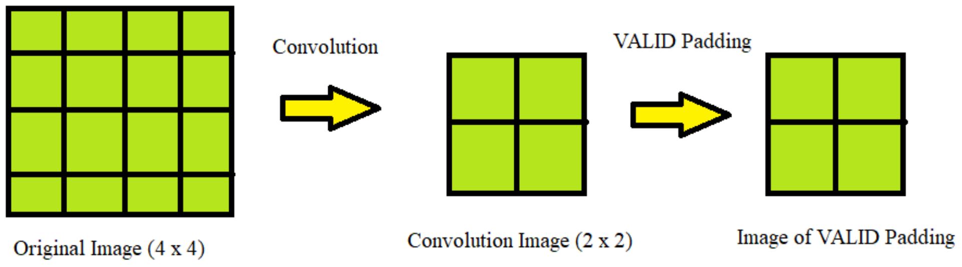

Fig. 5. Summary of VALID Padding Process.

If a Keras-Tensorflow library is used, a valid padding option means that no padding will be applied. The size of the image going through a valid padding option will be the same as the size of the convolution image as shown in Fig. 5.

D. Pooling Layer

There are two advantages of applying a pooling layer to the deep network. Firstly, it helps to decrease the size of the feature map and hence reduces the complexity of the network. Secondly, Guo et al. [35] shows it also helps to extract the important feature optimally. Based on Victor and Isabel [36], there are three types of pooling operators, which are maximum pooling, average pooling, and attentive pooling. For the maximum pooling operator, the maximum element in each overlapping area between the kernel and the feature map will be chosen as the resultant output. Shallu and Rajesh [37] identify the only disadvantage of maximum pooling operation is if most of the values on the feature map are high values, the significant features may be discarded. Fig. 6 shows the operational flow of the maximum pooling process with a 4 x 4 feature map, 2 x 2 kernel size with a step size of two pixels.

According to Sharma et al. [37], the average pooling operation process differs from the maximum pooling process. The new value in the respective position of the feature map is obtained by calculating the average pixels of each overlapping area between the kernel and the feature map. Fig. 7 shows the summary of an average pooling process with a 4 x 4 feature map, 2 x 2 kernel size with a step size of two pixels.

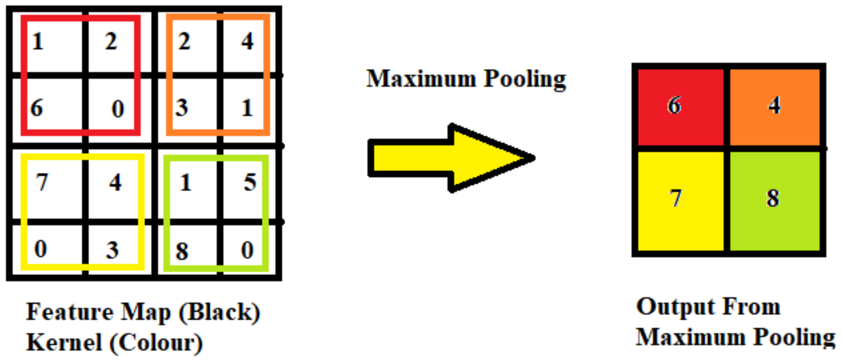

Fig. 6. Operational Flow of a Maximum Pooling Process (4 x 4 Feature Map, 2 x 2 Kernel Size with Step Size of Two Pixels).

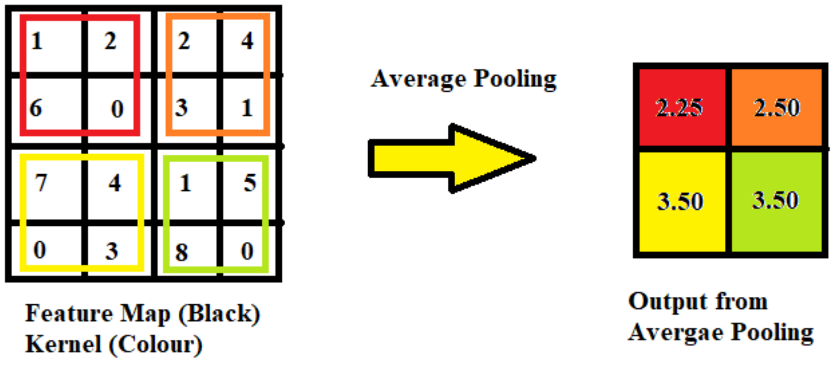

Fig. 7. Operational Flow of an Average Pooling Process (4 x 4 Feature Map, 2 x 2 Kernel Size with Step Size of Two Pixels).

E. *Activation Function*

The activation function, which is also known as the transfer function, is used to determine the neurons that will be excited and passed as the high state to the next neurons. According to Chigozie et al. [38] and Feng et al. [39], an ordinary neural network without an activation function, the output of each layer of the network will consist of a linear combination of its last layer, which is shown in (1). Let y = output, x = input, n = number of nth layers, b = bias.

$$y = w_1x_1 + w_2x_2 + \dots + w_nx_n + b \tag{1}$$

According to the formula above, the range of the output will start from the negative infinity until positive infinity. This shows that the neurons in the network are not limited to a certain finite range. Conversely, with the presence of an activation function, the linear output will be converted to a non-linear result and the output range will fall within a finite value. According to Chigozie et al. [38] and Feng et al. [39], the non-linear result is shown in (2).

$$y = a(w_1x_1 + w_2x_2 + \dots + w_nx_n + b) \tag{2}$$

In this study, two types of activation functions, which are ReLu and Softmax function will be utilized. Wang et al. [40] stated that ReLu is an activation function that is based on a piece-wise function. ReLu function is known to be good in handling gradient-vanishing problems. This is the main advantage of a ReLu function compared to other activation functions such as Sigmoid and Tanh functions [41]. Another advantage of a ReLu function is it can be computed at a faster speed compared to the other functions. A positive gradient with a value equal to one will be produced for positive input, while a negative gradient is produced when the input is negative. Fig. 8 shows the graphical representation of the ReLu function.

According to Martin et al. [42], the Softmax function is a useful function that converts the weight vector to the probability distribution. Such a function will make sure that the output is in the range between zero until one and the sum of the outputs will be unity [43]. The softmax function is commonly used in models with multiple classes. Chigozie et al. [38] showed that the probability of each class will be provided and the class with the highest probability is considered as the target class. The only disadvantage of the Softmax function is the output value of zero cannot be produced and hence, a sparse probability distribution cannot be produced through this function. This is because any small output value in the sparse probability distribution will be treated as a negligible value which is zero.

F. *Fully Connected Layer*

A fully connected layer means each neuron is connected to every neuron to its next layer. The major function of the fully connected layer is to classify the input image into a variety of classes. The Softmax function is used in its output layer [44].

G. *Local Response Normalisation*

Local Response Normalisation (LNR) is a non-trainable layer based on the lateral inhibition process. It decreases the neighboring pixels activation state, which deems to be too huge in order to form a big contrast in a feature map. This normalization process involves a decreasing operator by squaring and normalizing the pixel values of the feature map in a local neighborhood [45]. From Alex et al. [46], the (3) representation for LNR operation is shown as below, where $b^i_{x,y}$ = output neuron, $a^i_{x,y}$ = input neuron, N = total kernel in the layer, x, y = position of it kernel, others = constant value.

$$b^i_{x,y} = a^i_{x,y} / \left(k + a \sum_{j=\max(0,i-\frac{n}{2})}^{\min(N-1,i+\frac{n}{2})} \left(a^j_{x,y}\right)^2\right)^B \tag{3}$$

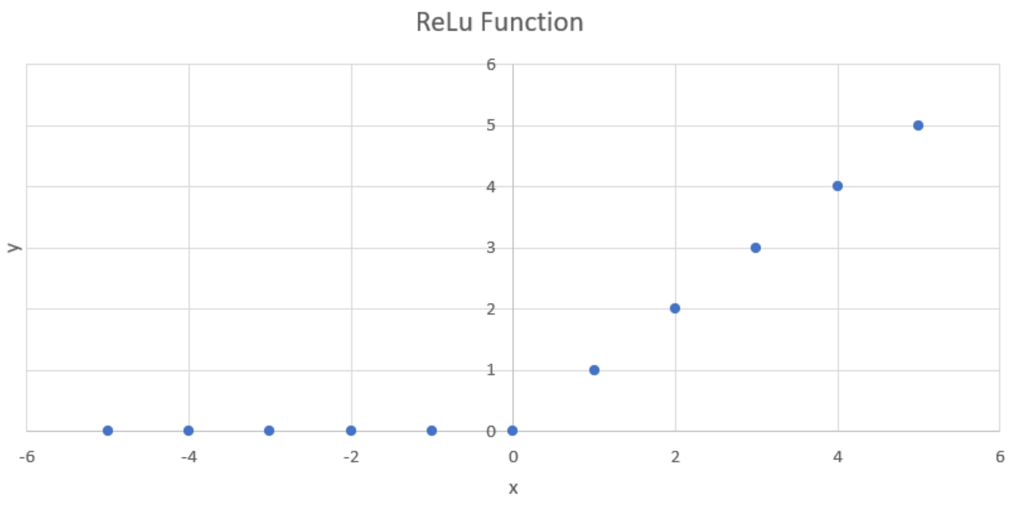

Fig. 8. Graphical Representation of ReLu Function.

III. Methodology

A. *CNN Model*

In this study, a compact CNN model [47] derived from the VGG-M model, which was introduced by Chatfield et al. [48]. A compact model is utilized because of the small number of available training data will lead to the problem of over-fitting according to the study by Nicholas et al. [49]. By having a compact property, a deep learning model can be optimized well even with a small number of data. In this study, the combined datasets have only 441 micro-expression videos, which is considered as a small training set. This is the primary reason deeper and newer models, such as Resnet [50] and DenseNet-SPP [51] are not used as the base model.

VGG-M has a good balance between computational speed and accuracy. Hence, it has excellent average performance and has been widely used in tasks involving the fields of vision such as multi-biometric recognition. The original VGG-M consists of nine layers network with five convolutional layers, three fully-connected layers and one flatten layer. The five convolutional layers have the kernel number of 96 for the first convolutional layer, 256 for the second convolutional layer, and 512 for the third, fourth, and fifth convolutional layer. Jiang et al. [52] proposed the fully-connected layers have a kernel size of 128 for the first and second layers, while three nodes for the third fully-connected layer to reduce the complexity of the training process. Each convolutional and full-connected layer is coupled with the ReLu activation function except for the last fully-connected layer, which is coupled with the Softmax activation function. In [53], LNR is applied after the first and second convolutional layers only. A maximum pooling layer is applied after each LNR layer and also after the fifth convolutional layer to make the model more robust and have a better generalization capability [54]. Table II shows the summary of the modified VGG-M architecture used in this study.

According to Table II, the primacy change that can be observed is the reduction in stride size for the first and second maximum pooling (Pool1 and Pool2) from two to one. This is because a larger feature map size is needed to insert the Spatial Pyramid Pooling (SPP) layer. Note that the size of the input image used in this study is 75 x 75. If the original stride size of the first and second maximum pooling layer is used, the feature map size after going through the second layer will become 3 x 3 only, which is not enough to embed the multiple average pooling processes in the SPP layers. Conversely, if the stride size is changed to one, the feature map size will be 13 x 13, which is enough to implement the multi-scale average pooling in the SPP layer.

TABLE II. Summary of the Modified VGG-M Architecture

Table Head	**Type of Layer**	**Kernel Number**	**Kernel Size**	**Stride**	**Padding**	**Activation Function**
Conv1	Convolution	96	7 x 7	2 x 2	Valid	ReLu
Norm1	LRN	-	-	-	-	-
Pool1	Maximum Pooling	-	3 x 3	1 x 1	-	-
Conv2	Convolution	256	5 x 5	2 x 2	Valid	ReLu
Norm2	LRN	-	-	-	-	-
Pool2	Maximum Pooling	-	3 x 3	1 x 1	-	-
Conv3	Convolution	512	3 x 3	1 x 1	Same	ReLu
Conv4	Convolution	512	3 x 3	1 x 1	Same	ReLu
Conv5	Maximum Pooling	512	3 x 3	1 x 1	Same	ReLu
Pool5	Flatten Layer		3 x 3	2 x 2	-	-
Flat1	Full-connected Layer		-	-	-	-
FC1	Full-connected Layer	128	-	-	-	ReLu
FC2	Full-connected Layer	128	-	-	-	ReLu
FC3	Convolution	3	-	-	-	Softmax

B. Spatial Pyramid Pooling

For general SPP, the input will undergo three down-sampling operations separately which are average pooling, batch normalization, and rectified linear unit activation function (ReLu). The kernel sizes of average pooling are different between the parallel layers. After that, the size of the down-sampling output will be adjusted (resized process) according to the skip-connection layer size. Then, these outputs are combined using a concatenation operator [55] as shown in Fig. 9.

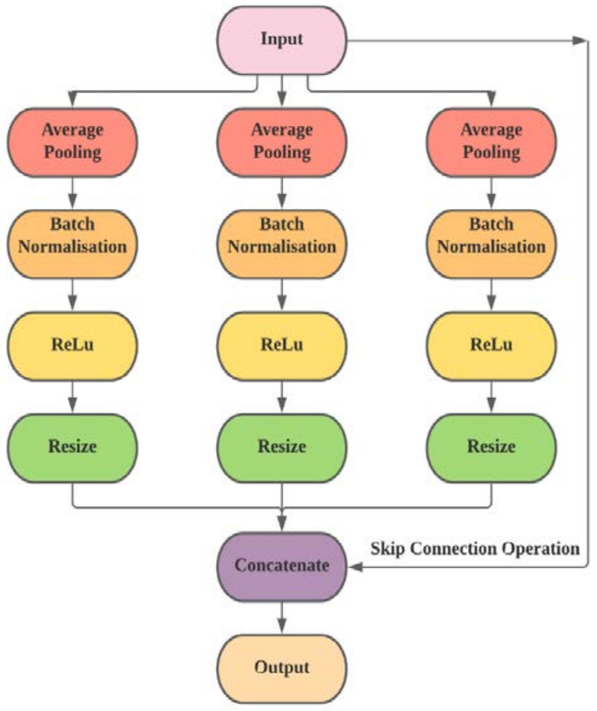

Fig. 9. General SPP Layer.

IV. Result and Discussion

A. Basic Settings

Google Colaboratory (Colab) was used as the platform to conduct the training and training process of the CNN model for micro-expression recognition for detecting a human's true emotion. Colab is a free cloud service developed by Google Research that enables the researcher to write and run Python code through the internet browser. It is a platform that is well suited for machine learning-based research. Another advantage of using Colab is it provides free GPU for the user that can be used for deep learning training. In this study, the programming language used is Python with Tensorflow library 2.4.1 through "Python three Google compute Engine Backend (GPU)" with Ram size of 12.72 GB and disk size of 68.40 GB. The virtual GPU used is Tesla P100-PCIE-16GB.

B. Hyperparameter

According to Lisha et al. [56] and Kandel et al. [57], hyperparameters can be considered as an input to a CNN algorithm which determines the performance of the deep learning algorithm towards the new and unseen data. Several hyperparameters that will be optimized in this work are learning rate, batch size, epochs, and type of optimizer. The type of optimizer determines how the weights are renewed by decreasing the loss or error [58]. In this study, Adamax optimizer is used as the sole optimization algorithm. This is because according to the research from [59], [60], Adamax produces a stable calculation method to renew the weights that ensure the stability of the CNN model.

Batch size is the number of data used for training of CNN model before the weights are updated. Smaller batch size can lead to slower convergence, while a larger batch size enables the CNN model to reach optimum minima. After performing several tests, a batch size of 64 is used throughout the experiment which achieves better stability and convergence compared to other batch sizes. Learning rate determines the rate of updating the weights. We are using 0.0001 as our learning rate for the CNN model. Jaya et al. [61] stated that the learning rate is not very high because a high value will cause the CNN architecture to become very unstable.

Epoch is defined as the number of iterations for a CNN model being trained by the whole datasets. Colab has a limitation that requires the user to interact with the system without idling by more than 90 minutes, after which it will stop the session automatically. Therefore, in [62] states that the number of epoch and learning rate need to be selected carefully so that the number of epoch can be minimized. This study has set the maximum number of epoch to be 120 iterations. Table III shows the summary of the other hyper-parameters that have been set in this study.

C. Dataset

Three ME datasets have been selected for this study, which are CASMEII [63], SMIC [64], and SAMM [65]. The first dataset, CASME II consists of 247 ME video clips from 26 subjects. The resolution of all videos is initially set to 640 x 480 pixels while the cropped image resolution is 340 x 280 pixels. Only five types of emotions are being considered, which are happiness, surprise, disgust, regression, and others [66]. The second dataset, SAMM involving 159 ME video clips from 29 subjects. The initial resolution of the videos is 2040 x 1080 pixels, while the cropped video resolution is 400 x 400 pixels. Rather than five emotion categories, SAMM consists of eight emotion classes, which are angry, contempt, disgust, fear, happiness, sadness, surprise, and others [67],[68]. The last dataset, SMIC comprises of 164 ME video clips from 16 subjects. The size of every image is 640 x 480 pixels. This dataset has three categories of emotion only that include positive, negative, and surprise [64], [66].

TABLE III. Hyperparameter Setting

Hyperparameter	Value
Learning Rate	0.0001
Batch Size	64
Epochs	120
Optimizer	ADAMAX

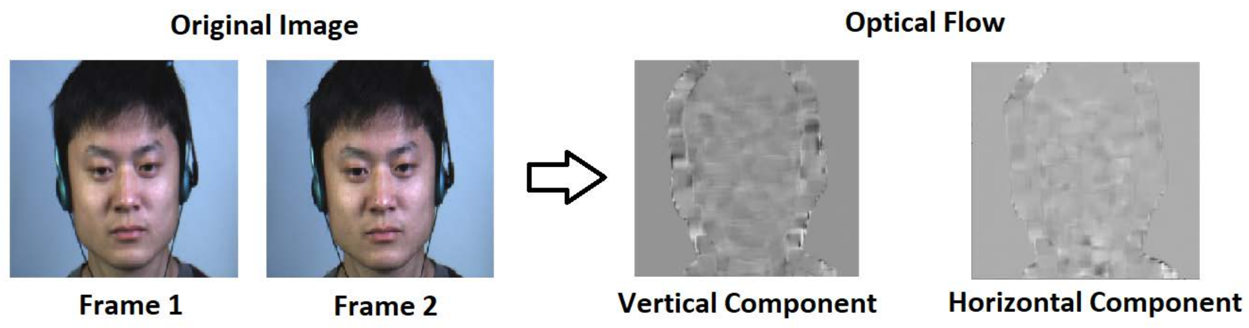

Fig. 10. Samples of Optical Flow between Two Consecutive Frames.

In this study, all images from all three datasets will be resized to 300 x 300 pixels, which will result in the same resolution for the respective vertical and horizontal components of the optical flow image. After that, the optical images will be down-scale again to the CNN input requirement, which is 75 x 75 pixels. According to Song and Zengfu [69], optical flow is the apparent motion between the video frame or image frame. It has a high-level feature in analyzing the visual motion information compared to the original image sequences, which allows it to have a better and more efficient data representation for ME [70]. Fig. 10 shows some examples of the calculated optical flow images, whereby the black and white color shows the presence of motion between the frames, while the grey color indicates that there is no motion for that respective pixels

The proposed system performance was verified by using 570 videos from 71 subjects that comprise of CASME II, SAMM, and SMIC datasets. There is no validation dataset used during the training phase. For the training and testing process, the dataset is divided according to leave-one-subject-out for testing, while the rest subjects will be used as training. This means that the model is trained using 70 subjects and 1 subject is used for testing. The output of this study will be labeled and classified into three classes: positive, negative, and surprise emotions. Positive emotion involves happy micro-expression which is considered as a "good" human emotion, while negative emotion is considered as a "bad" human emotion that can be further broken down into disgust, sadness, and fear. The third class of emotion, surprise is the emotion that a human expresses when he senses any difference between the expectation and the reality. Fig. 11, Fig. 12, and Fig. 13 portray an example of each type of emotion from different datasets.

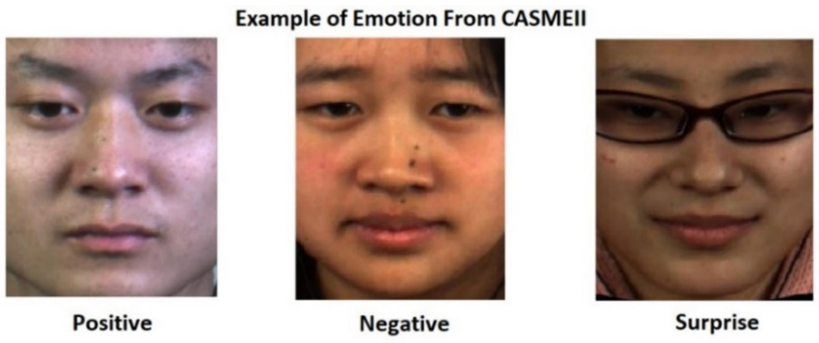

Fig. 11. Example of each Emotion from CASMEII.

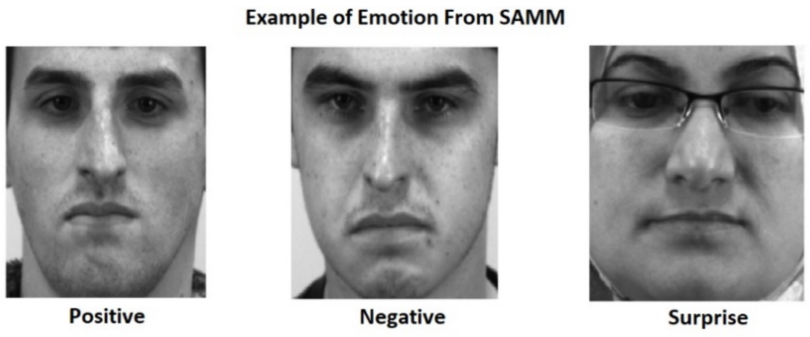

Fig. 12. Example of each Emotion from SAMM.

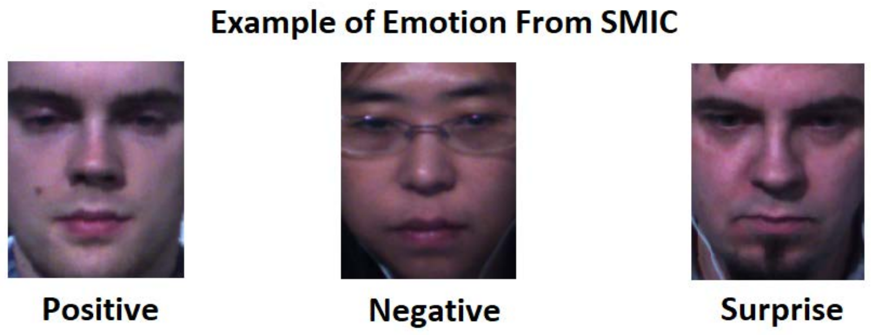

Fig. 13. Example of each Emotion from SMIC.

D. Evaluation Metric

The evaluation metric used in this study is the accuracy of micro-expression recognition for detecting a human's true emotion. According to Duygu [71], accuracy is defined as the ratio of true classification to total classification (true and false classification), which is formulated as in (4), whereby TP is defined as true positive, TN is true negative, FP is false positive, and FN is false negative.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

E. Experimental Setting

Several training processes were performed based on the modified VGG-M architecture for micro-expression recognition to detect human's authentic emotions. For every experiment, the values of other hyperparameters are set to constant to make sure that the experiments are conducted in fair conditions. After that, the Spatial Pyramid Pooling (SPP) layer of various configurations (in terms of the number of layers, kernel size of average pooling, and position where SPP layers are added) will be embedded into the modified VGG-M architecture. The performances for each configuration will be compared and evaluated by computing their recognition accuracy by using combined datasets of CASMEII, SMIC, and SAMM. Fig. 14 shows the summary of the major procedures that were performed to extract the performance accuracy.

There will be eight variants of SPP architectures that will be tested as detailed out below:

- First Variant = two parallel layers, Kernel size: two, four, Position: After first Layer of VGG-M.
- Second Variant = two parallel layers, Kernel size: two, four, Position: After second Layer of VGG-M.
- Third Variant = three parallel layers, Kernel size: two, four, six, Position: After first Layer of VGG-M.
- Fourth Variant = three parallel layers, Kernel size: two, four, six, Position: After second Layer of VGG-M.
- Fifth Variant = four parallel layers, Kernel size: two, four, six, Position: After first Layer of VGG-M.
- Sixth Variant = four parallel layers, Kernel size: two, four, six, eight, Position: After second Layer of VGG-M.
- Seventh Variant = five parallel layers, Kernel size: two, four six, eight, ten, Position: After first Layer of VGG-M.
- Eighth Variant = five parallel layers, Kernel size: two, four, six, eight, ten, Position: After second Layer of VGG-M.

The SPP layers with the different number of SPP layers and kernel size of average pooling will be added in two ways, one of the ways is added after the first layer and another way is inserted after the second layer of the VGG-M module as shown in Fig. 15.

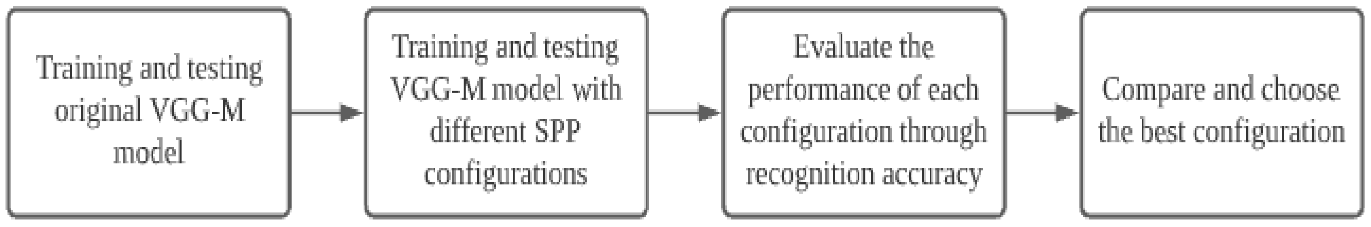
Training and testing original VGG-M model
Training and testing VGG-M model with different SPP configurations
Evaluate the performance of each configuration through recognition accuracy
Compare and choose the best configuration

Fig. 14. Summary of the Experimental Procedures.

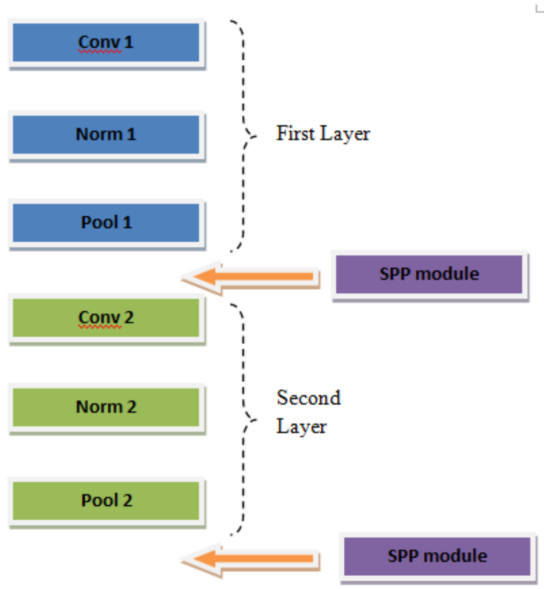

Fig. 15. Placement of the SPP Module.

F. *Dataset Analysis*

Table IV shows the results of modified VGG-M with several settings of parallel branches and placement of the SPP modules. Besides that, the results of combining both hyper-parameters are also reported in the same table. Fig. 16 shows the graph for training process of the original VGG-M and Fig. 17 shows the graph of the training results of VGG-M with SPP inserted after the second CNN layer.

TABLE IV. PERFORMANCE RESULTS OF VARIOUS SPP CONFIGURATIONS

Types of datasets	**Accuracy (%)**								
	Original (without SPP)	***After first layer***				***After second layer***			
		2 SPP	3 SPP	4 SPP	5 SPP	2 SPP	3 SPP	4 SPP	5 SPP
Combined	75.96	74.75	75.21	75.96	75.06	76.42	75.36	76.87	76.27
CASME II	86.21	84.37	86.67	83.45	86.67	88.05	83.91	86.67	86.67
SAMM	71.21	70.2	68.18	69.19	69.7	70.02	71.21	69.7	69.7
SMIC	70.73	69.92	70.73	74.8	69.11	71.14	71.14	73.98	72.36

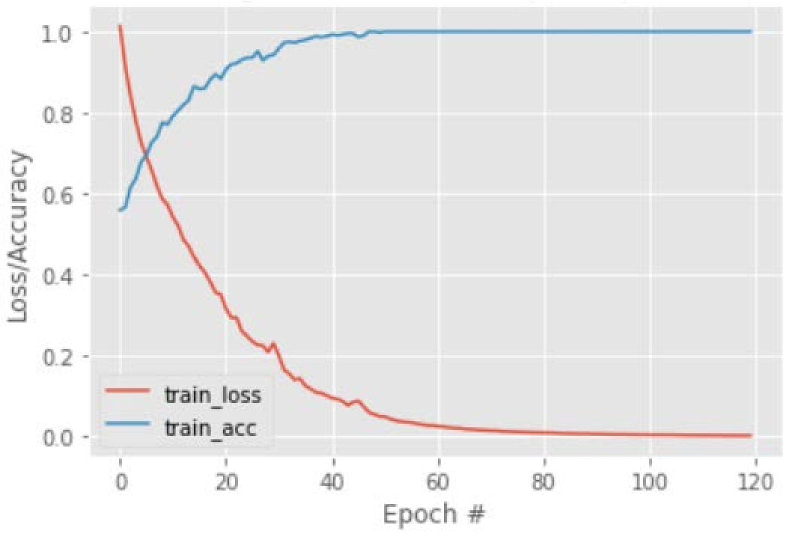

Fig. 16. The Original VGG-M Training Process.

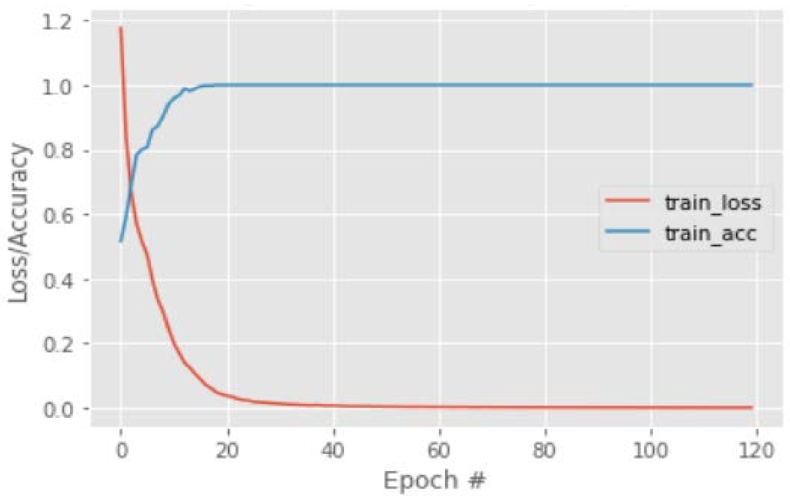

Fig. 17. VGG-M Training Process using SPP after the Second VGG-M Layer.

By optimizing the network configurations (different SPP parallel branches and module placement), the recognition accuracy of the proposed deep model has improved the recognition performance by four percent to seven percent when compared to the original network for the SAMM dataset. Meanwhile, for the SMIC dataset, the recognition accuracy of the model using the combined dataset has improved from 1% to 5%.

G. SPP Configuration and Placement

From Table IV, the results show that the recognition accuracy of the model using the combined dataset, CASME II and SMIC after the addition of the SPP module has improved the performance, except for the case of the SAMM dataset, in which the performance has become worst. Firstly, based on the combined dataset results, the best recognition accuracy is obtained with four SPP parallel layers, added after the second layer with an accuracy of 76.87%, which is an improvement of 0.91% compared to the base model.

For the CASME II dataset analysis, the best recognition accuracy is obtained by using two SPP parallel layers inserted after the second layer with 88.05% accuracies, which is an improvement of 1.84% compared to the base model. Among many datasets and configurations, this is the highest recognition accuracy obtained. While, for the case of the SAMM dataset, the addition of more SPP parallel layers into the base model does not increase the recognition accuracy, whereby the best number of SPP layers is three that is added after the second layer with a recognition accuracy of 71.21%.

On the other hand, for the SMIC dataset, four SPP parallel layers that are embedded after the first layer produced the highest recognition accuracy of 74.80% with an improvement of 4.07% compared to the base model. This is the greatest improvement in terms of recognition accuracy among many configurations that have been tested. Hence, the best setup for each dataset and the combined datasets are shown in Table V.

However, among all the configurations, SPP with four parallel branches produces the best general performance where it produces the best recognition accuracies for the combined datasets and SMIC dataset. Besides that, the best overall placement of the SPP module is if it is added after the second layer of the base model. On average, it produces results with higher recognition accuracy. In addition, most of the highest recognition accuracy is obtained after adding the SPP right after the second layer to the base model.

H. Improvement of the SPP Configuration

To further improve the base model performance, a few new variants of the SPP is introduced as follow:

- Ninth Variant = five parallel layers, Kernel size: two, three, four, five, six, Position: After first Layer of VGG-M
- Tenth Variant = five parallel layers, Kernel size: two, three, four, five, six, Position: After second Layer of VGG-M

TABLE V. SUMMARY OF THE BEST SETUP IN TERMS OF SPP CONFIGURATION AND PLACEMENT

Types of datasets	**Best configuration of SPP**	
	SPP number	***Position***
combined	Four	After second layer
CASME II	Two	After second layer
SMIC	Three	After second layer
SAMM	Four	After first layer

TABLE VI. PERFORMANCE BETWEEN DIFFERENT SETS OF KERNEL SIZE

Types of datasets	**Accuracy (%)**			
	After first layer		***After second layer***	
	first set Kernel	second set Kernel	first set Kernel	second set Kernel
Combined	75.06	74.91	76.27	76.42
CASME II	86.67	84.37	86.67	86.21
SAMM	69.70	71.21	69.70	70.20
SMIC	69.11	69.51	72.36	72.76

TABLE VII. SUMMARY OF THE BEST CONFIGURATION IN TERMS OF THE AVERAGE POOLING KERNEL SIZE FOR DIFFERENT DATASET SETUP

Table Head	**Table Column Head**
Combined	two, three, four, five and six
CASME II	two, four, six, eight and ten
SMIC	two, three, four, five and six
SAMM	two, three, four, five and six

The performance difference between the new variants of SPP with the earlier variants is due to the kernel sizes of average pooling, where the sizes are smaller for the latter variants. In short, the ninth and the tenth variants have bigger kernel sizes compared to the seventh and the eighth variants. Table VI shows the recognition accuracy of the modified model with different kernel sizes.

According to Table V, when adding the SPP layer after the first layer to the base model, the first set of kernel sizes produce better recognition accuracy compared to the second set for the combined datasets and CASME II dataset. On the other hand, it is the opposite trend for the SAMM and SMIC datasets. Besides that, if the SPP is added after the second layer, the second set of kernel sizes produces better accuracy compared to the first set for the combined, SAMM and SMIC datasets, except for the CASMEII dataset.

By comparing the overall results, the second set of kernel sizes result in higher average recognition accuracy for the test done on the combined, SAMM and SMIC dataset (76.42%, 71.21%, and 72.76% respectively). However, the first set of kernel sizes return the best recognition accuracy for the test done on the CASME II dataset (86.67 % accuracy). Hence, it can be concluded that the second set of average pooling kernel sizes is the better alternative compared to the first set as shown in Table VII.

V. CONCLUSION

This study has managed to improve the recognition accuracy of the CNN-based deep learning model by embedding SPP to the base model. Various configurations have been tested to find the optimal setup. For the combined dataset, the best SPP configuration is obtained by adding four parallel branches of the SPP after the second layer of the base model. This configuration has produced a recognition accuracy of 76.87%, which is an improvement of 0.91% over the base model. For the CASME II dataset, two parallel branches of the SPP layers added after the second layer of the base model have produced 88.05% recognition accuracy, which is an improvement of 1.84%. Meanwhile, the best SPP configuration for the SMIC dataset is four parallel branches added after the first layer with an improvement of 4.07%. There is not much performance improvement that can be observed with the addition of the SPP module when the test is done on the SAMM dataset. Generally, the overall best SPP configuration is by embedding four parallel branches of SPP with average pooling kernel sizes of two, three, four, five, and six, added after the second layer of the base model. For future works, more datasets can be combined to produce a more robust micro-expression-based automated emotion recognition system. Other than that, the dataset will be resampled using data augmentation methods to balance the class distribution between the emotion class. Besides that, synthetic data augmentation through the generative adversarial method can be employed to further increase the training samples.

ACKNOWLEDGMENT

The authors would like to acknowledge funding from Universiti Kebangsaan Malaysia (Geran Universiti Penyelidikan: GUP-2019-008) and Ministry of Higher Education Malaysia (Fundamental Research Grant Scheme: FRGS/1/2019/ICT02/UKM/02/1).

REFERENCES

[1] C. Correia-Caeiro, K. Guo, and D. S. Mills, “Perception of dynamic facial expressions of emotion between dogs and humans,” Animal Cognition, vol. 23, no.3, May 2020.

[2] Y. Huang, F. Chen, S. Lv, and X. Wang, “Facial Expression Recognition: A Survey,” Symmetry, vol. 11, no.10, p. 1189, Sep. 2019.

[3] B. Allaert, I. M. Bilasco, and C. Djeraba, “Micro and macro facial expression recognition using advanced Local Motion Patterns,” IEEE Transactions on Affective Computing, pp. 1–12, 2019.

[4] D. Deng, Z. Chen, Y. Zhou, and B. Shi, “MIMAMO Net: Integrating Micro- and Macro-Motion for Video Emotion Recognition,” Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no.03, Apr. 2020.

[5] D. Borza, R. Danescu, R. Itu, and A. Darabant, “High-Speed Video System for Micro-Expression Detection and Recognition,” Sensors, vol. 17, no.12, Dec. 2017.

[6] G. Zhao and X. Li, “Automatic Micro-Expression Analysis: Open Challenges,” Frontiers in Psychology, vol. 10, Aug. 2019.

[7] Y. Wang, Y. Li, Y. Song, and X. Rong, “Facial Expression Recognition Based on Random Forest and Convolutional Neural Network,” Information, vol. 10, no.12, Nov. 2019.

[8] S. O. Slim, A. Atia, M. M.A., and M.-S. M.Mostafa, “Survey on Human Activity Recognition based on Acceleration Data,” International Journal of Advanced Computer Science and Applications, vol. 10, no.3, 2019.

[9] M. A. Zulkifley, M. M. Mustafa, A. Hussain, A. Mustapha, and S. Ramli, “Robust Identification of Polyethylene Terephthalate (PET) Plastics through Bayesian Decision,” PLoS ONE, vol. 9, p. e114518, Dec. 2014.

[10] Y. Lai, “A Comparison of Traditional Machine Learning and Deep Learning in Image Recognition,” Journal of Physics: Conference Series, vol. 1314, p. 012148, Oct. 2019.

[11] Niall O’ Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan and Joseph Walsh, “Deep Learning vs. Traditional Computer Vision,” 2020.

[12] D. Gibert, C. Mateu, and J. Planes, “The rise of machine learning for detection and classification of malware: Research developments, trends and challenges,” Journal of Network and Computer Applications, vol. 153, Mar. 2020.

[13] J. Wang, H. Wang, X. Zhu, and P. Zhou, “A Deep Learning Approach in the DCT Domain to Detect the Source of HDR Images,” Electronics, vol. 9, Dec. 2020.

[14] Z. Ke, C. Le, and Y. Yao, “A multivariate grey incidence model for different scale data based on spatial pyramid pooling,” Journal of Systems Engineering and Electronics, vol. 31, pp. 770–779, Aug. 2020.

[15] C. Dewi, R.-C. Chen, and S.-K. Tai, “Evaluation of Robust Spatial Pyramid Pooling Based on Convolutional Neural Network for Traffic Sign Recognition System,” Electronics, vol. 9, May 2020.

[16] M. A. Zulkifley, N. A. Mohamed, and N. H. Zulkifley, “Squat Angle Assessment Through Tracking Body Movements,” IEEE Access, vol. 7, pp. 48635–48644, 2019.

[17] Y.-H. Oh, J. See, A. C. le Ngo, R. C.-W. Phan, and V. M. Baskaran, “A Survey of Automatic Facial Micro-Expression Analysis: Databases, Methods, and Challenges,” Frontiers in Psychology, vol. 9, Jul. 2018.

[18] X. Huang, G. Zhao, X. Hong, W. Zheng, and M. Pietikäinen, “Spontaneous facial micro-expression analysis using Spatiotemporal Completed Local Quantized Patterns,” Neurocomputing, vol. 175, Jan. 2016.

[19] J. He, J.-F. Hu, X. Lu, and W.-S. Zheng, “Multi-task mid-level feature learning for micro-expression recognition,” Pattern Recognition, vol. 66, Jun. 2017.

[20] X. Huang and G. Zhao, “Spontaneous facial micro-expression analysis using spatiotemporal local radon-based binary pattern,” Oct. 2017.

[21] Li X, Xiaopeng H, Moilanen A., Huang X., Pfister T. and Zhao G., “Towards Reading Hidden Emotions: A Comparative Study of Spontaneous Micro-Expression Spotting and Recognition Methods,” IEEE Transactions on Affective Computing, vol. 9, Oct. 2018.

[22] S.-T. Liong, J. See, R. C.-W. Phan, K. Wong, and S.-W. Tan, “Hybrid Facial Regions Extraction for Micro-expression Recognition System,” Journal of Signal Processing Systems, vol. 90, Apr. 2018.

[23] S. L. Happy and A. Routray, “Fuzzy Histogram of Optical Flow Orientations for Micro-Expression Recognition,” IEEE Transactions on Affective Computing, vol. 10, Jul. 2019.

[24] A. C. le Ngo, J. See, and R. C.-W. Phan, “Sparsity in Dynamics of Spontaneous Subtle Emotions: Analysis and Application,” IEEE Transactions on Affective Computing, vol. 8, Jul. 2017.

[25] F. Xu, J. Zhang, and J. Z. Wang, “Microexpression Identification and Categorization Using a Facial Dynamics Map,” IEEE Transactions on Affective Computing, vol. 8, Apr. 2017.

[26] Ping, L., Zheng, W., Ziyan, W., Qiang, L., Yuan, Z., and Minghai, X., et al., “Micro-Expression Recognition by Regression Model and Group Sparse Spatio-Temporal Feature Learning,” IEICE Transactions on Information and Systems, vol. E99.D, 2016.

[27] Y. Zong, X. Huang, W. Zheng, Z. Cui, and G. Zhao, “Learning From Hierarchical Spatiotemporal Descriptors for Micro-Expression Recognition,” IEEE Transactions on Multimedia, vol. 20, no.11, pp. 3160–3172, Nov. 2018.

[28] X. Zhou, “Understanding the Convolutional Neural Networks with Gradient Descent and Backpropagation,” Journal of Physics: Conference Series, vol. 1004, Apr. 2018.

[29] S. L. Oh, J. Vicnesh, E. J. Ciaccio, R. Yuvaraj, and U. R. Acharya, “Deep Convolutional Neural Network Model for Automated Diagnosis of Schizophrenia Using EEG Signals,” Applied Sciences, vol. 9, Jul. 2019.

[30] Y. Xie, W. Dai, Z. Hu, Y. Liu, C. Li, and X. Pu, “A Novel Convolutional Neural Network Architecture for SAR Target Recognition,” Journal of Sensors, vol. 2019, pp. 1–9, May 2019.

[31] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” Insights into Imaging, vol. 9, Aug. 2018.

[32] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, “Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction,” Sensors, vol. 17, Apr. 2017.

[33] Yang Chen, Shengwu Qin, Shuangshuang Qiao, Qiang Dou, Wenchao Che, Gang Su, Jingyu Yao and Uzodigwe Emmanuel Nnanwuba., “Spatial Predictions of Debris Flow Susceptibility Mapping Using Convolutional Neural Networks in Jilin Province, China,” Water, vol. 12, Jul. 2020.

[34] M. Hashemi, “Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation,” Journal of Big Data, vol. 6, Dec. 2019.

[35] S. Guo, T. Yang, W. Gao, C. Zhang, and Y. Zhang, “An Intelligent Fault Diagnosis Method for Bearings with Variable Rotating Speed Based on Pythagorean Spatial Pyramid Pooling CNN,” Sensors, vol. 18, Nov. 2018.

[36] V. Suárez-Paniagua and I. Segura-Bedmar, “Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction,” BMC Bioinformatics, vol. 19, Jun. 2018.

[37] S. Sharma and R. Mehra, “Implications of Pooling Strategies in Convolutional Neural Networks: A Deep Insight,” Foundations of Computing and Decision Sciences, vol. 44, Sep. 2019.

[38] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation Functions: Comparison of trends in Practice and Research for Deep Learning,” pp. 124–133, Nov. 2018.

[39] J. Feng and S. Lu, “Performance Analysis of Various Activation Functions in Artificial Neural Networks,” Journal of Physics: Conference Series, vol. 1237, Jun. 2019.

[40] Y. Wang, Y. Li, Y. Song, and X. Rong, “The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition,” Applied Sciences, vol. 10, Mar. 2020.

[41] Y. Yu, K. Adu, N. Tashi, P. Anokye, X. Wang, and M. A. Ayidzoe, “RMAF: Relu-Memristor-Like Activation Function for Deep Learning,” IEEE Access, vol. 8, pp. 72727–72741, 2020.

[42] A. F. T. Martins and R. F. Astudillo, “From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification,” Feb. 2016.

[43] A. Aldhahab, S. Ibrahim, and W. Mikhael, “Stacked Sparse Autoencoder and Softmax Classifier Framework to Classify MRI of Brain Tumor Images,” International Journal of Intelligent Engineering and Systems, vol. 13, Jun. 2020.

[44] S. Das and J. Mukherjee, “Automatic License Plate Recognition Technique using Convolutional Neural Network,” International Journal of Computer Applications, vol. 169, Jul. 2017.

[45] S. Samir, E. Emary, K. El-Sayed, and H. Onsi, “Optimization of a Pre-Trained AlexNet Model for Detecting and Localizing Image Forgeries,” Information, vol. 11, May 2020.

[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Communications of the ACM, vol. 60, May 2017.

[47] S. R. Abdani, M. A. Zulkifley, and N. Hani Zulkifley, “A Lightweight Deep Learning Model for COVID-19 Detection,” in 2020 IEEE

Symposium on Industrial Electronics & Applications (ISIEA), pp. 1–5, Jul. 2020

[48] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the Devil in the Details: Delving Deep into Convolutional Nets," pp. 121–129, May 2014.

[49] Nicholas Waytowich, Vernon J Lawhern, Javier O Garcia, Jennifer Cummings, Josef Faller, Paul Sajda and Jean M Vettel., "Compact convolutional neural networks for classification of asynchronous steady-state visual evoked potentials," Journal of Neural Engineering, vol. 15, Dec. 2018.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Jun. 2016.

[51] S. R. Abdani and M. A. Zulkifley, "DenseNet with Spatial Pyramid Pooling for Industrial Oil Palm Plantation Detection," in 2019 International Conference on Mechatronics, Robotics and Systems Engineering (MoRSE), pp. 134–138, Dec. 2019.

[52] H.-J. Jiang, Y.-A. Huang, and Z.-H. You, "Predicting Drug-Disease Associations via Using Gaussian Interaction Profile and Kernel-Based Autoencoder," BioMed Research International, vol. 2019, Aug. 2019.

[53] I. Omara, X. Wu, H. Zhang, Y. Du, and W. Zuo, "Learning pairwise SVM on hierarchical deep features for ear recognition," IET Biometrics, vol. 7, Nov. 2018.

[54] Z. Li, F. Li, L. Zhu, and J. Yue, "Vegetable Recognition and Classification Based on Improved VGG Deep Learning Network Model," International Journal of Computational Intelligence Systems, vol. 13, p. 559, 2020.

[55] S. R. Abdani, M. A. Zulkifley, and M. Mamat, "U-Net with Spatial Pyramid Pooling Module for Segmenting Oil Palm Plantations," Sep. 2020.

[56] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization," Mar. 2016.

[57] I. Kandel and M. Castelli, "The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset," ICT Express, vol. 6, Dec. 2020.

[58] N. M. Aszemi and P. D. D. Dominic, "Hyperparameter Optimization in Convolutional Neural Network using Genetic Algorithms," International Journal of Advanced Computer Science and Applications, vol. 10, no.6, 2019.

[59] D. Yi, J. Ahn, and S. Ji, "An Effective Optimization Method for Machine Learning Based on ADAM," Applied Sciences, vol. 10, no.3, Feb. 2020.

[60] D.-S. Kwon, C. Jin, M. Kim, and W. Koo, "Mooring-Failure Monitoring of Submerged Floating Tunnel Using Deep Neural Network," Applied Sciences, vol. 10, no.18, Sep. 2020.

[61] J. T. Hardinata, H. Okprana, A. P. Windarto, and W. Saputra, "Analisis Laju Pembelajaran dalam Mengklasifikasi Data Wine Menggunakan Algoritma Backpropagation," J-SAKTI (Jurnal Sains Komputer dan Informatika), vol. 3, no.2, Sep. 2019.

[62] S. S. Liew, M. Khalil-Hani, S. Ahmad Radzi, and R. Bakhteri, "Gender classification: a convolutional neural network approach," Turkish Journal of Electrical Engineering & Computer Sciences, vol. 24, 2016.

[63] Yan W.J, Li X.; Wang S.J; Zhao G, Liu Y.J; Chen Y.H. and Fu X., "CASME II: An Improved Spontaneous Micro-Expression Database and the Baseline Evaluation," PLoS ONE, vol. 9, no.1, Jan. 2014.

[64] X. Li, T. Pfister, X. Huang, G. Zhao, and M. Pietikainen, "A Spontaneous Micro-expression Database: Inducement, collection and baseline," Apr. 2013.

[65] A. K. Davison, C. Lansley, N. Costen, K. Tan, and M. H. Yap, "SAMM: A Spontaneous Micro-Facial Movement Dataset," IEEE Transactions on Affective Computing, vol. 9, no.1, Jan. 2018

[66] C. Guo, J. Liang, G. Zhan, Z. Liu, M. Pietikainen, and L. Liu, "Extended Local Binary Patterns for Efficient and Robust Spontaneous Facial Micro-Expression Recognition," IEEE Access, vol. 7, 2019.

[67] D. Y. Choi and B. C. Song, "Facial Micro-Expression Recognition Using Two-Dimensional Landmark Feature Maps," IEEE Access, vol. 8, 2020.

[68] J. Li, C. Soladie, and R. Seguier, "A Survey on Databases for Facial Micro-Expression Analysis," 2019.

[69] S. Wang and Z. Wang, "Optical Flow Estimation with Occlusion Detection," Algorithms, vol. 12, no.5, May 2019.

[70] M. Peng, C. Wang, T. Chen, G. Liu, and X. Fu, "Dual Temporal Scale Convolutional Neural Network for Micro-Expression Recognition," Frontiers in Psychology, vol. 8, Oct. 2017.

[71] D. Kaya, "Optimization of SVM Parameters with Hybrid CS-PSO Algorithms for Parkinson's Disease in LabVIEW Environment," Parkinson's Disease, vol. 2019, May 2019.