applied sciences

MDPI

Article

Multiple Convolutional Neural Networks Fusion Using Improved Fuzzy Integral for Facial Emotion Recognition

Cheng-Jian Lin [1,*], Chun-Hui Lin [2], Shyh-Hau Wang [2,3] and Chen-Hsien Wu [1]

[1] Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung City 411, Taiwan
[2] Department of Computer Science & Information Engineering Nation Cheng Kung University, Tainan 701, Taiwan
[3] Intelligent Manufacturing Research Center National Cheng Kung University, Tainan 701, Taiwan
* Correspondence: cjlin@ncut.edu.tw; Tel.: +886-423-924-505

Received: 29 April 2019; Accepted: 20 June 2019; Published: 26 June 2019

Abstract: Facial expressions are indispensable in human cognitive behaviors since it can instantly reveal human emotions. Therefore, in this study, Multiple Convolutional Neural Networks using Improved Fuzzy Integral (MCNNs-IFI) were proposed for recognizing facial emotions. Since effective facial expression features are difficult to design; deep learning CNN is used in the study. Each CNN has its own advantages and disadvantages, thus combining multiple CNNs can yield superior results. Moreover, multiple CNNs combined with improved fuzzy integral, in which its fuzzy density value is optimized through particle swarm optimization (PSO), overcomes the majority decision drawback in the traditional voting method. Two Multi-PIE and CK+ databases and three main CNN structures, namely AlexNet, GoogLeNet, and LeNet, were used in the experiments. To verify the results, a cross-validation method was used, and experimental results indicated that the proposed MCNNs-IFI exhibited 12.84% higher accuracy than that of the three CNNs.

Keywords: Facial emotion recognition; convolutional neural network; fuzzy integral; particle swarm optimization

1. Introduction

In 2016, when a computer program won the Google DeepMind Challenge, people started having a different understanding of artificial intelligence (AI). AI is no longer just a sci-fi plot in a movie but is being implemented around us. The key technology of AlphaGo is deep learning and it has become more prominent than the original terminology.

Deep learning is used in image recognition. With improvements in hardware, more people are investing in convolutional neural network (CNN). The origin of CNN can be traced back to 1998. LeCun et al. [1] proposed the LeNet-5 model, which uses the back propagation algorithm proposed by Rumelhart et al. [2] in 1986, to adjust the parameters of a neural network. In 1998, Alex Krizhevsky [3] proposed the AlexNet model and won the championship in ImageNet Large Scale Visual Recognition Challenge (ILSVRC). Dropout technology was used for the first time, and the rectified linear unit (ReLU) was used as an activation function to prove that increasing CNN layers can provide superior results.

In 2014, GoogLeNet proposed by Szegedy et al. [4] and VGGNet proposed by Simonyan et al. [5] were competitors in ILSVRC. The commonality between these two models is that the layers are deeper than those in other neural network methods. VGG-16 inherits LeNet and AlexNet's microarchitecture, whereas GoogLeNet boldly changed the network structure and improved the multilayer perceptron convolution layer to convolve with a 1×1 convolution kernel and introduced batch normalization [6].

Although the GoogLeNet depth consisted of only 22 layers, the overall size was considerably smaller than that of AlexNet and VGGNet. The number of parameters of GoogLeNet (5 million) are 12 times that of AlexNet, and the number of parameters of VGGNet are three times that of AlexNet. Therefore, GoogLeNet is a superior choice when memory or computing resources are limited. As the number of network layers deepens, a degradation problem occurs; it is impossible for networks to learn correct features. That is, the accuracy gradually reaches saturation; however, saturation is not caused by overfitting. To solve this problem, He et al. [7] proposed ResNet in 2015 to map low-level features directly to high-level networks in order to ensure that deep networks do not need to learn from scratch. Therefore, ResNet has a representation ability close to previous few layers, which reduces degradation problems and eases deep networks training.

The last layer, the fully connected layer, is the most parameterized part in CNNs, and overfitting problems can easily occur. Lin et al. [8] proposed global average pooling to replace the fully connected layer. Global average pooling has no parameter to train and reduces the overall burden in networks. In the training process, the stochastic gradient descent (SGD) is the most common optimization method used presently; this method randomly selects a fixed number of training samples each time for training. This method can learn effectively; however, its disadvantages include a tendency to fall into the local solution and an over reliance on the learning rate. To solve these problems, researchers have successively proposed various improved optimization methods [9–12].

In the area of pattern recognition, there are generally two types of combinations: classifier selection and classifier fusion. The presumption in classifier selection is that each classifier is "an expert" in some local area of the feature space. Classifier fusion assumes that all classifiers are trained over the whole feature space, and are thereby considered as competitive rather than complementary. In this study, we focus on classifier fusion. The way of combining different classifiers has been proposed and frequently considered in order to achieve better classification performance than using one single classifier. In other words, by combining the outputs of a team of classifiers, this proposed work aims at making more accurate decisions than the one by choosing the single best member within the team. Recently, several different classifier combination algorithms [13–15] have been proposed. Soto et al. [13] presented an intuitive, rigorous mathematical description of the voting process in an ensemble of classifiers. Kuncheva [14] used soft computing in classifier combination, such as neural network, fuzzy set, and evolutionary computation. Kevric et al. [15] developed a combining classifier model based on tree-based algorithms for network intrusion detection. The proposed detection algorithm was to classify whether the incoming network traffics are normal or an attack. Lin et al. [16] presented multiple functional neural fuzzy networks fusion using fuzzy integral (FI). FI is a nonlinear function that is defined with respect to a fuzzy measure and able to merge the information granted from multiple sources. In [16], since the fuzzy density in traditional fuzzy integral is often set according to user experiences, the recognition rate of the fusion model cannot be improved. The purpose of this study is to improve traditional FI. That is, the fuzzy density is determined using an evolutionary method. Furthermore, in the decision-making process, the FI considers both the objective evidence supplied by each information source and the expected worth of each subset of information sources.

In this study, Multiple Convolutional Neural Networks using Improved Fuzzy Integral (MCNNs-IFI) were proposed for recognizing facial emotions. The proposed method involved several trained CNNs as input to the fuzzy integral, and classification results were computed using the output rules of Sugeno or Choquet. Since effective facial expression features are difficult to design, deep learning CNN is used in the study. Each CNN has its own advantages and disadvantages. Therefore, combining multiple CNNs can yield superior results. Moreover, multiple CNNs combined with improved fuzzy integral, in which its fuzzy density value is optimized through particle swarm optimization (PSO), overcomes the majority decision drawback in the traditional voting method.

In Section 2, the proposed MCNNs-IFI method and its architecture are described. In Section 3, experimental results obtained using the Multi-PIE database and CK+ database are presented, and Section 4 presents the conclusion and future prospects.

2. Proposed MCNNs-IFI

Each CNN has its own advantages and disadvantages, in order to retain its superiorities, the multiple convolutional neural networks using improved fuzzy integral (MCNNs-IFI) is proposed in this study. There are two kinds of combinations; one is combining the same CNN architectures with different optimization methods; another is combining different CNN architectures with the same optimization methods. The proposed MCNNs-IFI architecture is presented in Figure 1.

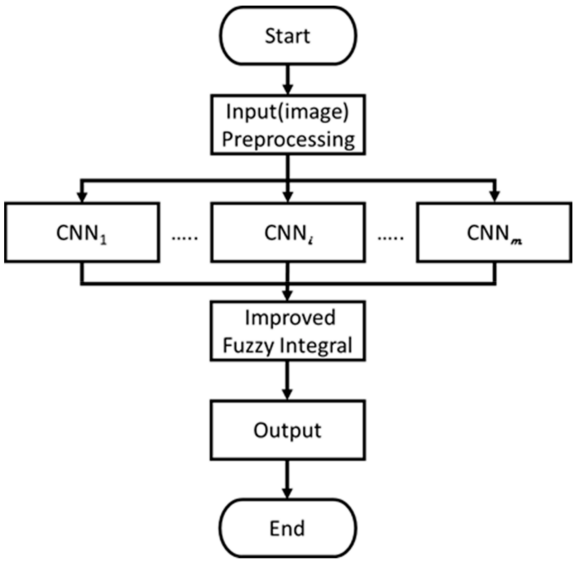

Figure 1. Structure of MCNNs-IFI.

First, train the CNNs of different architectures or different optimization methods with the same set of training data to ensure that an individual CNN produces different outputs, then use the fuzzy integral of fuzzy measure to evaluate each classifier. Last, calculate the combined output results from CNNs to obtain superior recognition accuracy.

In Figure 1, the outputs from m CNNs are used as the input of the improved fuzzy integral. After calculation, the output is obtained by referencing the worth of the performance in each CNN.

2.1. CNN

Basic CNN can be roughly divided into two parts. As depicted in Figure 2, the first part (blue area) uses convolution and pooling to extract features. The second part (orange area) is a fully connected neural network classifier. Traditional CNNs consist of the feature extraction and classifier architecture. However, in recent CNNs, the fully connected part has been replaced by average pooling, which reduces the degree of overfitting and parameters required during training.

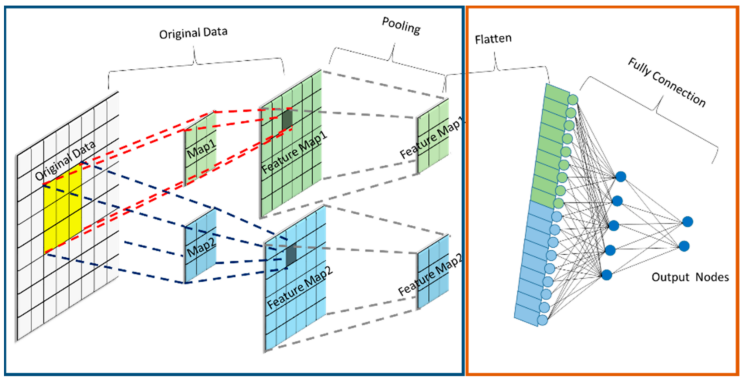

Figure 2. CNN.

The following subsections describe the three important operations in the feature extraction section, namely convolution, activation function, and pooling.

2.1.1. Convolution Layer

Convolution is a feature extraction process, which is based on the concept of a receptive field, and is used by many traditional feature extractions such as Sobel and Gabor. Convolution uses a convolution kernel mask of the sliding window on the input matrix. The formula is as follows:

$$F_{IJ} = \sum_{i=0}^{K_w} \sum_{j=0}^{K_h} x_{ij} \times kw_{ij} \tag{1}$$

where F_{IJ} is the output matrix and K_w and K_h are the width and height of the convolution kernel, respectively. Generally, $K_w = K_h$ is set as a square kernel, x_{ij} is the input matrix, and kw_{ij} is the weight of the convolution kernel, and to maintain the same width and height of the matrix after convolution, the edge of the input matrix is usually padded with 0, and 0 is still 0 after the convolution operation. Therefore, padding 0 will not affect convolution.

2.1.2. Activation Function

The purpose of the activation function is to obtain nonlinear outputs for linearly combined networks. However, when the network is deep and the number of layers is high, the sigmoid, which is used in earlier networks, appears and the gradient disappears during back propagation. Therefore, ReLU is used as the activation function in CNN.

The definition of ReLU is as follows:

$$f(x) = \max(0, x) \tag{2}$$

Figure 3 illustrates that if the input x is smaller than 0, then the output is 0; if the input x is bigger than 0; then x is the direct output.

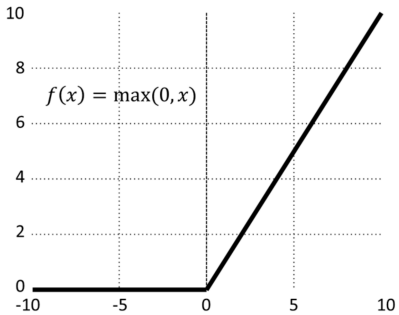

Figure 3. ReLU function.

2.1.3. Pooling Layer

After the convolution operation, the input feature map is compressed to simplify network computation complexity. Feature compression is performed to extract main features.

Pooling is used to reduce the dimension. Pooling involves using a sliding window mask that operates on the input matrix; however, the mask does not overlap in the moving process. That is, the step is equal to the height of the convolution kernel to enable an N × N mask to reduce $1/N^2$ times of the input feature matrix and thereby reduce the dimension.

Pooling is generally divided into the maximum or average pooling. Figure 4 displays the maximum pooling. Here, the largest value in the mask is used as the output, and other values in the mask are directly discarded. In the convolution process, it is usually desirable to retain the most

prominent features, and features that are not prominent in theory should not affect characteristics; therefore, maximum pooling is used for the convolution process. Average pooling calculates the average of all values in the mask, so it usually uses in output layer. Average pooling has a physical meaning in the calculation process, such that a large target can obtain a high output value during the calculation process, as depicted in Figure 5. In addition to reducing the dimension, the pooling method can provide the network a certain degree of rotation and displacement invariance.

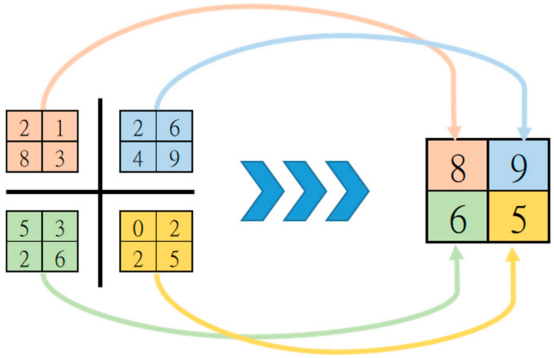

Figure 4. Max pooling.

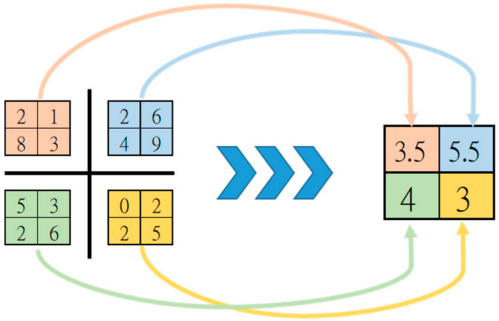

Figure 5. Average pooling.

2.2. Proposed Improved Fuzzy Integral

The fuzzy density in traditional fuzzy integral is often set according to user experiences. Many optimization algorithms can be used to determine the optimal fuzzy density, such as particle swarm optimization (PSO), genetic algorithm (GA), differential evolution (DE), firefly algorithm (FA), and artificial bee colony (ABC). Without crossover and mutation steps, PSO is relatively simple, with fewer parameters to adjust, and easy to implement in software and hardware. Therefore, in this study, the proposed improved fuzzy integral uses PSO to automatically determine the optimal fuzzy density.

The hypothesis $X = \{x_i\}_{i=1:m}$ represents a set of m classifiers which means the numbers of CNN used in this study and $g(x_i)$ indicates the fuzzy measure and is regarded as the worth of the subset x_i. That is, x_i is the output from i^{th} classifier and $g(x_i)$ is the confidence degree of this class. The fuzzy measure value is [0,1]; if the value is 1, then it indicates that this output can be fully trusted. If the value is 0, then the output has no reference value. The fuzzy measure properties are shown as follows:

(1) $g(X) = 1$ indicates that when all classifier outputs are consistent, the results must be trusted.
(2) $g(\emptyset) = 0$ indicates that outputs of all classifiers are not considered.
(3) Fuzzy measure should be an incremental monotonic function:

$$A \subset B \subset X \text{ , then } 0 \le g(A) \le g(B) \le 1 \quad (3)$$

To get the fuzzy measure it should start at fuzzy density which has only one element in g(X) for instance $g(\{x_1\})$ and $g(\{x_2\})$. The relationship between fuzzy measures and fuzzy density is presented in Figure 6. In this figure, $g(\{x_1\})$, $g(\{x_2\})$, and $g(\{x_3\})$ are called fuzzy density and $g(\{x_1, x_2, x_3\})$ is the confidence degree of the specific class combining three CNNs.

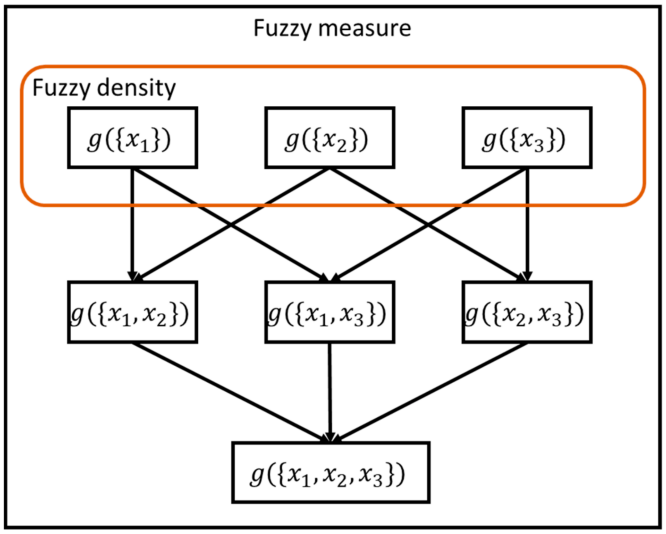

Figure 6. Relationship between fuzzy measures and fuzzy density.

Figure 7 illustrates the flow of fuzzy density and fuzzy measures with PSO algorithm. PSO [17] is a type of an optimization algorithm and it is based on the concept of foraging birds. Birds flying in the air for foraging can be seen as several particles in a solution space of a problem. Then, through multiple iterations, the best solution can be obtained. The fuzzy densities are n × k float numbers from 0 to 1, where n is the number of classifiers (i.e., the number of CNNs), and k is the number of categories in the classification problem (i.e., the number of class); therefore, the dimensions of the solution space are n × k dimensions and range from 0 to 1. In the initialization phase, y particles are randomly generated in the solution space, and each particle p denotes a set of fuzzy densities, as depicted in Figure 8.

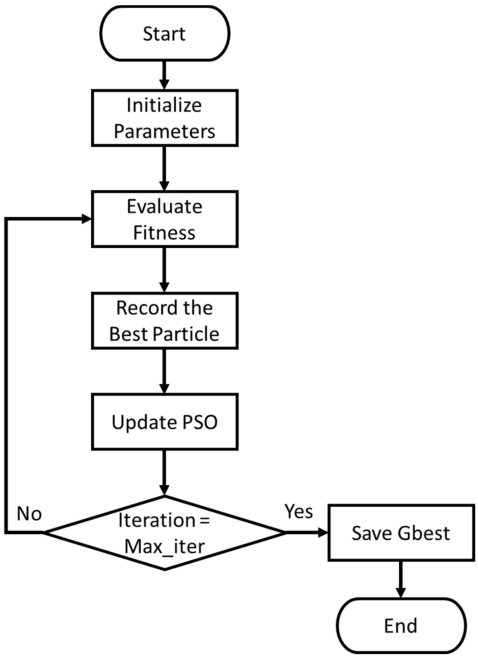

Figure 7. Flow of fuzzy density.

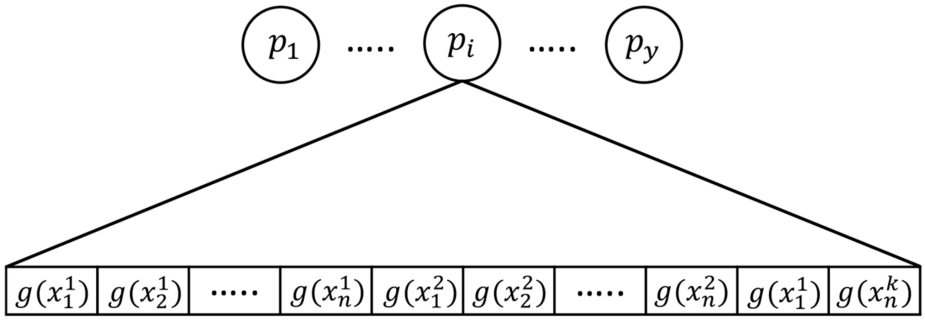

Figure 8. Encoding of particles.

Then, the fitness value of each particle is evaluated by calculating the precision of the verified data as follows:

$$\text{Fitness} = \frac{TP}{TP + FP} \tag{4}$$

where TP is the number of samples that are predicted correctly, and FP is the number of samples that are predicted incorrectly.

In the particle swarm, the position of the overall optimal fitness value was defined as *Gbest*; the position of each particle's best fitness value was defined as $Pbest_i$, where i = {1, 2 . . . , j}. When updating a particle's position, the current fitness was evaluated to check whether the current fitness value of the particle is better than *Gbest* and $Pbest_i$. If yes, then the particle's location was updated and recorded.

$$v_{id}(t) = w \times v_{id}(t-1) + c_1 \times rand() \times (Pbest_i - p_{id}(t)) + c_2 \times rand() \times (Gbest - p_{id}(t)) \tag{5}$$

$$p_{id}(t+1) = p_{id}(t) + v_{id}(t) \tag{6}$$

where $v_{id}(t)$ expresses the velocity of the i^{th} particle in the d^{th} dimension at the t^{th} iteration; w, c_1, and c_2 are variable parameters; $p_{id}(t)$ is the position of the d^{th} dimension of the i^{th} particle at the t^{th} iteration; and $rand()$ is a random value from 0 to 1. From equations (5) and (6), it can be observed that each particle in the process of moving refers to its past experience and group experience to determine the fuzzy density with the highest accuracy until the number of iterations reaches the maximum number of iterations. After knowing the value of fuzzy density, the fuzzy measure can be calculated by the formula revealed below:

$$\mathrm{g}(A \cup B) = \mathrm{g}(A) + \mathrm{g}(B) + \lambda \mathrm{g}(\mathrm{A})\mathrm{g}(\mathrm{B}),\ \mathrm{A}, \mathrm{B} \subset \mathrm{X} \tag{7}$$

λ in Equation (7) can be calculated by Equation (8)

$$\lambda + 1 = \prod_{i=1}^{n} \left(1 + \lambda g^i\right) \tag{8}$$

The number of λ in the improved fuzzy integral is equal to the output dimension, which indicates that each output category can calculate one λ and $\lambda \in (-1, \infty)$ with the following characteristics:

(1) $if\ \sum_{i=1}^{n} g^i = 1\ ,\ then\ \lambda = 0$
(2) $if\ \sum_{i=1}^{n} g^i < 1\ ,\ then\ \lambda > 0$
(3) $if\ \sum_{i=1}^{n} g^i > 1\ ,\ then - 1 \le \lambda < 0$

In this paper, two output rules, Sugeno and Choquet, are selected to implement the fuzzy integral. Before calculating the two output rules, some parameters and sets should be defined. First, the output of the n classifiers $h(x_i)$ are sorted to ensure that the following relationship holds,

$h(x_{\pi_1}) \geq h(x_{\pi_2}) \geq \ldots h\left(x_{\pi_j}\right) \geq \ldots \geq h(x_{\pi_n})$, where π_j denotes the classifier which has the j^{th} highest output value. Therefore, $\pi_2 = 3$ indicates the third classifier's output value is the second largest, and $h(x_{\pi_1})$ represents the maximum output value of the classifier. Then, the following set is defined:

$$A_0 = \{\varnothing\} \tag{9}$$

$$A_i = A_{i-1} + \left\{x_{\pi_i}\right\} \tag{10}$$

Equations (11) and (12) present the improved fuzzy integral output of Sugeno and Choquet rules:

Sugeno FI:

$$Y_s = \vee_{i=1}^{N} (h(x_{\pi i}) \wedge g(A_i)) \tag{11}$$

Choquet FI:

$$Y_c = \sum_{i=1}^{n} h(x_{\pi i})(g(A_i) - g(A_{i-1})) \tag{12}$$

In this study, two fuzzy integral rules are calculated simultaneously, and the higher result among the two is the output of the fuzzy integral.

3. Experimental Results

To evaluate the proposed MCNNs-IFI method, emotional expressions were used as classification targets, and three CNNs were selected for combination, namely LeNet [1], AlexNet [3], and GoogLeNet [4].

To verify that the recognition rates of MCNNs-IFI are superior because of a combination of different network architectures or optimization methods, three trainings were performed with the same architecture and optimization method, and another training that integrated the architecture and optimization methods was performed using the improved fuzzy integral. The comparison between various optimization methods and the same optimization method indicated their difference.

In this study, the Multi-PIE and CK+ databases are used, and seven expressions are initially predefined. To objectively evaluate the accuracy of the fuzzy integral, the cross-validation method was used in Multi-PIE database by dividing data into 10 groups, and the average result of each group was derived to obtain precise accuracy. Moreover, each group of data was divided into training, verified, and testing data. Only training data were involved in the CNN training process, and verified data were used to adjust the fuzzy density. Finally, testing data were used to present accuracy. When the fuzzy density was performed, the maximum number of iterations was 1000, w was set to 0.8–0.2, which linearly decreased according to the number of iterations, and c_1 and c_2 were set to two. In CK+ database, each group of data was split into training and testing data. All the initial parameters such as w, c_1, c_2 and number of maximum iterations were set under the same conditions as Multi-PIE database.

3.1. Multi-PIE Face Database

Figure 9 illustrates the different angles and brightness facial expressions of the Multi-PIE dataset. Front facial photos were selected in the experiment and categorized into seven expressions: normal, happy, surprised, squinting, smiling, disgusted, and shouting (Figure 7). Tables 1–3 depict the results of different architectures with the same optimization method and the combination of CNNs using MCNNs-IFI. Tables 4–6 reveal the results of the same architecture with different optimization methods and the combination of CNNs using MCNNs-IFI. The 10-folder CV column in Tables 1–6 indicates the 10 groups of verified data. In those tables, the combination of CNNs using MCNNs-IFI shows higher accuracy.

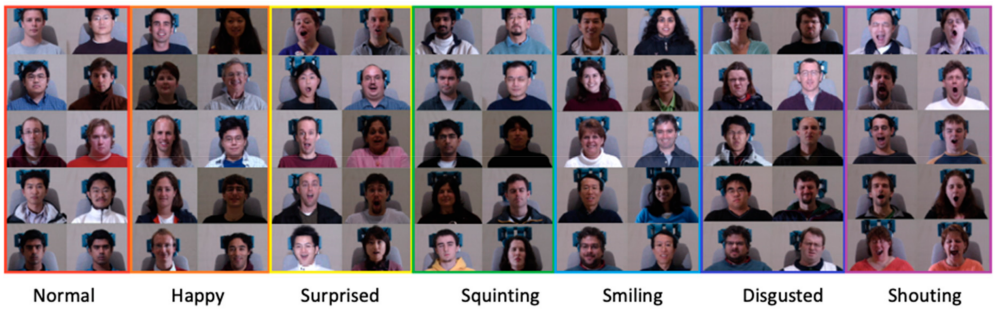

Figure 9. Multi-PIE database.

Table 1. Recognition results of different architectures with Adagrad optimization method.

10-folder CV	AlexNet (Adagrad)	GoogleNet (Adagrad)	LeNet (Adagrad)	AlexNet(Adagrad) + GoogleNet(Adagrad) + LeNet(Adagrad) + IFI
Round 1	85.82%	84.25%	90.75%	96.18%
Round 2	88.14%	87.11%	96.54%	97.75%
Round 3	84.71%	83.89%	98.71%	98.76%
Round 4	85.54%	82.71%	98.32%	98.43%
Round 5	87.25%	84.25%	93.61%	97.46%
Round 6	93.57%	84.68%	97.50%	98.89%
Round 7	87.18%	78.39%	95.11%	97.11%
Round 8	84.18%	84.57%	84.18%	91.61%
Round 9	85.36%	82.61%	93.96%	95.71%
Round 10	87.50%	87.57%	93.93%	97.07%
Average	86.93%	84.00%	94.26%	96.79%

Table 2. Recognition results of different architectures with Adam optimization method.

10-folder CV	AlexNet (Adam)	GoogleNet (Adam)	LeNet (Adam)	AlexNet(Adam) + GoogleNet(Adam) + LeNet(Adam) + IFI
Round 1	92.00%	86.00%	99.00%	100.00%
Round 2	92.21%	90.85%	99.64%	99.79%
Round 3	89.92%	93.25%	99.89%	99.89%
Round 4	92.21%	92.25%	99.75%	99.78%
Round 5	92.60%	93.60%	99.75%	99.92%
Round 6	92.60%	92.32%	99.71%	99.78%
Round 7	91.42%	94.00%	99.67%	99.82%
Round 8	92.00%	94.53%	99.82%	99.85%
Round 9	91.03%	91.42%	99.67%	99.78%
Round 10	91.67%	91.67%	99.96%	99.96%
Average	91.77%	91.99%	99.69%	99.85%

Table 3. Recognition results of different architectures with stochastic gradient descent (SGD) optimization method.

10-folder CV	AlexNet (SGD)	GoogleNet (SGD)	LeNet (SGD)	AlexNet(SGD) + GoogleNet(SGD) + LeNet(SGD) + IFI
Round 1	90.79%	93.53%	99.96%	99.96%
Round 2	93.21%	93.57%	99.86%	99.95%
Round 3	92.53%	93.75%	99.89%	99.91%
Round 4	91.00%	91.32%	99.79%	99.91%
Round 5	92.36%	85.07%	99.71%	99.89%
Round 6	93.25%	94.42%	99.64%	99.96%
Round 7	91.75%	92.18%	99.82%	99.90%
Round 8	91.46%	91.07%	99.75%	99.89%
Round 9	93.06%	95.32%	99.64%	99.97%
Round 10	91.36%	90.71%	99.82%	99.96%
Average	92.08%	92.09%	99.79%	99.93%

Table 4. Recognition results of AlexNet architectures with different optimization method.

10-folder CV	AlexNet (Adagrad)	AlexNet (Adam)	AlexNet (SGD)	AlexNet(Adagrad + Adam + SGD) + IFI
Round 1	85.82%	92.00%	90.79%	94.21%
Round 2	88.14%	92.21%	93.21%	95.89%
Round 3	84.71%	89.92%	92.53%	96.79%
Round 4	85.54%	92.21%	91.00%	95.89%
Round 5	87.25%	92.60%	92.36%	96.43%
Round 6	93.57%	92.60%	93.25%	97.21%
Round 7	87.18%	91.42%	91.75%	96.07%
Round 8	84.18%	92.00%	91.46%	95.93%
Round 9	85.36%	91.03%	93.06%	96.14%
Round 10	87.50%	91.67%	91.36%	95.96%
Average	86.93%	91.77%	92.08%	96.05%

Table 5. Recognition results of GoogleNet architectures with different optimization method.

10-folder CV	GoogleNet (Adagrad)	GoogleNet (Adam)	GoogleNet (SGD)	GoogleNet (Adagrad + Adam + SGD) + IFI
Round 1	84.25%	86.00%	93.53%	96.96%
Round 2	87.11%	90.85%	93.57%	97.21%
Round 3	83.89%	93.25%	93.75%	97.07%
Round 4	82.71%	92.25%	91.32%	96.14%
Round 5	84.25%	93.60%	85.07%	96.25%
Round 6	84.68%	92.32%	94.42%	97.18%
Round 7	78.39%	94.00%	92.18%	97.64%
Round 8	84.57%	94.53%	91.07%	96.25%
Round 9	82.61%	91.42%	95.32%	97.68%
Round 10	87.57%	91.67%	90.71%	95.96%
Average	84.00%	91.99%	92.09%	96.84%

Table 6. Recognition results of LeNet architectures with different optimization method.

10-folder CV	LeNet (Adagrad)	LeNet (Adam)	LeNet (SGD)	LeNet (Adagrad + Adam + SGD) + IFI
Round 1	90.75%	99.00%	99.96%	100.00%
Round 2	96.54%	99.64%	99.86%	99.89%
Round 3	98.71%	99.89%	99.89%	99.97%
Round 4	98.32%	99.75%	99.79%	99.93%
Round 5	93.61%	99.75%	99.71%	99.84%
Round 6	97.50%	99.71%	99.64%	99.82%
Round 7	95.11%	99.67%	99.82%	99.83%
Round 8	84.18%	99.82%	99.75%	99.85%
Round 9	93.96%	99.67%	99.64%	99.96%
Round 10	93.93%	99.96%	99.82%	100.00%
Average	94.26%	99.69%	99.79%	99.90%

3.2. CK+ Database

CK+ database contains 593 sequences across 123 subjects and the images are taken with different brightness levels in Figure 10. All facial photos are categorized into eight expressions: anger, contempt, disgust, fear, happiness, neutral, sadness and surprise and the results are displayed from Tables 7–12. The results of different architectures with the same optimization method and the combination of CNNs using MCNNs-IFI are shown in the first three tables. Tables 10, 11 and 12 reveal the results of the same architecture with different optimization methods and the combination of CNNs using MCNNs-IFI. In those tables, the combination of CNNs using MCNNs-IFI exhibits higher accuracy among the compared methods.

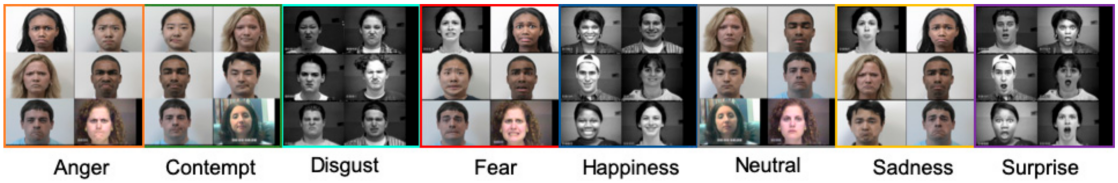

Figure 10. CK+ database.

Table 7. Recognition results of different architectures with Adagrad optimization method.

Methods	AlexNet (Adagrad)	GoogleNet (Adagrad)	LeNet (Adagrad)	AlexNet(Adagrad) + GoogleNet(Adagrad) + LeNet(Adagrad) + IFI
Accuracy	83.33%	88.07%	97.69%	97.94%

Table 8. Recognition results of different architectures with Adam optimization method.

Methods	AlexNet (Adam)	GoogleNet (Adam)	LeNet (Adam)	AlexNet(Adam) + GoogleNet(Adam) + LeNet(Adam) + IFI
Accuracy	93.97%	91.92%	97.3%	98.58%

Table 9. Recognition results of different architectures with SGD optimization method.

Methods	AlexNet (SGD)	GoogleNet (SGD)	LeNet (SGD)	AlexNet(SGD) + GoogleNet(SGD) + LeNet(SGD) + IFI
Accuracy	90.89%	91.41%	97.43%	97.94%

Table 10. Recognition results of AlexNet architectures with different optimization method.

Methods	AlexNet (Adagrad)	AlexNet (Adam)	AlexNet (SGD)	AlexNet(Adagrad + Adam + SGD) + IFI
Accuracy	83.33%	93.97%	90.89%	95.76%

Table 11. Recognition results of GoogleNet architectures with different optimization method.

Methods	GoogleNet (Adagrad)	GoogleNet (Adam)	GoogleNet (SGD)	GoogleNet (Adagrad + Adam + SGD) + IFI
Accuracy	88.07%	91.92%	91.41%	96.92%

Table 12. Recognition results of LeNet architectures with different optimization method.

Methods	LeNet (Adagrad)	LeNet (Adam)	LeNet (SGD)	LeNet (Adagrad + Adam + SGD) + IFI
Accuracy	97.69%	97.3%	97.43%	98.07%

3.3. Discussion

Figure 11 shows that applying the proposed MCNNs-IFI in various CNNs is feasible and effectively provides a high overall recognition rate. Figures 12 and 13 depict the two testing photos, and their recognition results using SGD as the optimal method are presented in Tables 13 and 14. While the complexity is increased by adding more CNNs, the advantages of each CNN can be retained using the proposed method. Besides, with regards to the recognition results, only one CNN can predict correct results, and the other two classifiers provide incorrect results; if a combination of three classifiers is used in the traditional voting concept, the final result will be incorrect.

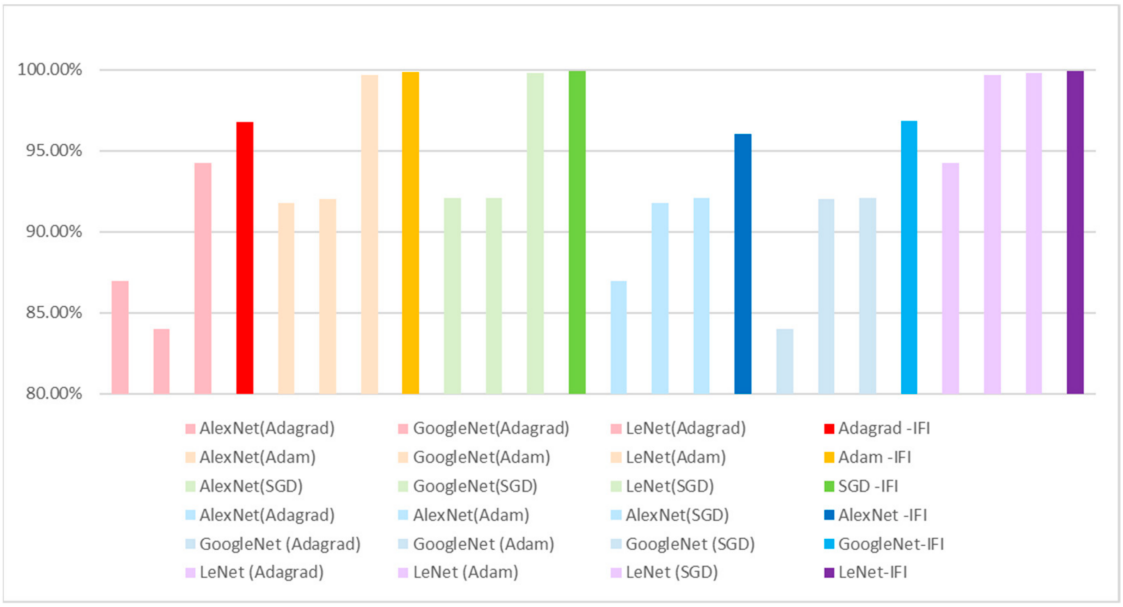

Figure 11. Recognition accuracy in different CNNs.

Figure 12. Multi-PIE database example photo (8)-135.

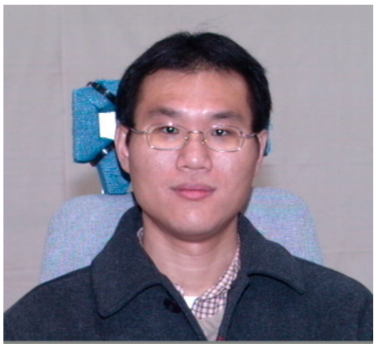

Figure 13. Multi-PIE database example photo (1)-360.

Table 13. Example photo (8)-135 recognition results.

Classifier	Normal	Happy	Surprised	Squinting	Smiling	Disgusted	Shouting
Ground truth	1	0	0	0	0	0	0
LeNet [1]	0.99	0.00	0.01	0.01	0.00	0.00	0.00
AlexNet [3]	0.17	0.00	0.00	0.83	0.00	0.00	0.00
GoogLeNet [4]	0.01	0.00	0.00	0.99	0.00	0.00	0.00
Proposed method	0.57	0.00	0.00	0.53	0.00	0.00	0.00

Table 14. Example photo (1)-360 recognition results.

Classifier	Normal	Happy	Surprised	Squinting	Smiling	Disgusted	Shouting
Ground truth	1	0	0	0	0	0	0
LeNet [1]	0.34	0.01	0.01	0.00	0.62	0.00	0.02
AlexNet [3]	0.02	0.00	0.00	0.00	0.90	0.07	0.00
GoogLeNet [4]	0.95	0.04	0.00	0.00	0.00	0.00	0.00
Proposed method	0.67	0.02	0.00	0.00	0.43	0.02	0.00

4. Conclusions

In this paper, the proposed MCNNs-IFI was implemented for recognizing facial emotions. It takes output data from several trained CNNs as input data to the fuzzy integral, and the results of the classification are computed depending on the output of Sugeno or Choquet rules. Furthermore, the optimal fuzzy density value in the fuzzy integral was determined through PSO. In experimental results, the Multi-PIE and CK+ databases were used to classify facial emotions and the results indicated that the proposed MCNNs-IFI was at most 12.84% and 14.61% accurate over AlexNet, GoogLeNet, and LeNet in Multi-PIE database and CK+ database, respectively.

Future studies should automatically adjust the number of CNNs in the process and automatically filter and eliminate lower or similar classifiers to achieve the best combination of classifiers.

Author Contributions: Conceptualization, C.-H.L. and C.-J.L.; methodology, C.-J.L.; software, C.-H.W.; data curation, S.-H.W. and C.-J.L.; writing—original draft preparation, C.-H.L. and C.-J.L.; funding acquisition, C.-J.L.

Funding: This research was funded by the Ministry of Science and Technology of the Republic of China, grant number No. MOST 107-2221-E-167-023.

Conflicts of Interest: The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

1. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2323. [CrossRef]
2. Rumelhart, E.; Hinton, E.; Williams, J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
3. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
4. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS'12), Lake Tahoe, NV, USA, 3–6 December 2012; Volume 1, pp. 1097–1105.
5. Liu, S.; Deng, W. Very Deep convolutional neural network based image classification using small training sample size. In Proceedings of the 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), Kuala Lumpur, Malaysia, 3–6 November 2015; pp. 730–734.
6. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015; Volume 37, pp. 448–456.
7. Wu, S.; Zhong, S.; Liu, Y. Deep residual learning for image steganalysis. *Multimed. Tools Appl.* **2018**, *77*, 10437–10453. [CrossRef]
8. Lin, M.; Chen, Q.; Yan, S. Network in network. In Proceedings of the International Conference on Learning Representations (ICLR), Scottsdale, AZ, USA, 2–4 May 2013; pp. 1–10.
9. Asness, C.S.; Moskowitz, T.J.; Pedersen, L.H. Value and momentum everywhere. *J. Financ.* **2013**, *68*, 929–985. [CrossRef]
10. Duchi, J.; Hazan, E.; Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* **2011**, *12*, 2121–2159.

11. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2017**, arXiv:1609.04747, 1–14.
12. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015; pp. 1–15.
13. Soto, V.; Suárez, A.; Martínez-Muñoz, G. An urn model for majority voting in classification ensembles. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 5–10 December 2016; pp. 1–9.
14. Kuncheva, L.I. Combining classifiers: Soft computing solutions. In *Pattern Recognition, From Classical to Modern Approaches*; Pal, S.K., Pal, A., Eds.; World Scientific: Singapore, 2001; Volume 15, pp. 427–452.
15. Kevric, J.; Jukic, S.; Subasi, A. An effective combining classifier approach using tree algorithms for network intrusion detection. *Neural Comput. Appl.* **2017**, *28* (Suppl. 1), 1051–1058. [CrossRef]
16. Lin, C.J.; Kuo, S.S.; Peng, C.C. Multiple functional neural fuzzy networks fusion using fuzzy integral. *Int. J. Fuzzy Syst.* **2012**, *14*, 380–391.
17. Zhang, Y.; Li, H.G.; Wang, Q. A filter-based bare-bone particle swarm optimization algorithm for unsupervised feature selection. *Appl. Intell.* **2019**, 1–10. [CrossRef]