

Facial emotion recognition through artificial intelligence

Anas Quteishat^{1,2}, Mohammad Hassan¹, Ashraf Al Sharah², Ahmed Qutaishat³

¹Department of Communication and Computer Engineering, Faculty of Engineering, Al-Ahliyya Amman University, Amman, Jordan

²Department of Electrical Engineering, Faculty of Engineering Technology, Al-Balqa Applied University, Al Salt, Jordan

³General Foundation Program, Sohar University, Sohar, Oman

Article Info

Article history:

Received Oct 29, 2025

Revised Jul 29, 2026

Accepted Aug 12, 2026

Keywords:

Convolutional neural network
Deep learning
Facial emotional recognition
Fuzzy min-max neural network
Haar cascade algorithm

ABSTRACT

This paper presents a fuzzy min-max deep neural network (FMMDNN) for facial emotion recognition (FER). The fuzzy min-max neural network (FMMNN) will replace the classification layer in the convolutional neural network (CNN). The fuzzy min-max (FMM) exhibits appealing features such as online adaptation, fast training, interpretability, and noise robustness. The fully connected softmax layer in the CNN is replaced by the FMMNN, where hyperboxes will be created and used for classification purposes. The FER2013 dataset is used to test the performance of the proposed method. The empirical results show that adding an FMMNN layer to the CNN network improves the performance.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Anas Quteishat

Department of Communication and Computer Engineering, Faculty of Engineering

Al-Ahliyya Amman University

Amman, Jordan

Email: a.quteishat@ammanu.edu.jo

1. INTRODUCTION

The pursuit of linking human communication with machine systems has driven artificial intelligence development and improvements in human-computer interaction within an electronic world [1]–[5]. The advancement of facial emotion recognition (FER) enables this research to interpret human emotional signals which stand as a core component of this work. Researchers, together with technologists use facial expression data to achieve machine-based human emotion perception and understanding so machines can respond empathetically in human-machine interactions [6]. Multiple professional sectors along with operational areas receive implications from the research findings. Applying modern technology to human-computer interaction would render systems capable of interpreting human emotions and creating responses with increased natural understanding.

FER functions as a vital field for human-computer interaction and affective computing since it helps numerous industries, including healthcare, education, and entertainment companies, run their operations. Taxonomic research in FER generates automatic methods that analyze facial expressions to discover emotional states in humans effectively [7]. FER methods during their initial stage extracted specific facial features based on manual face landmarks such as eye-to-eye and eye-to-nose-to-mouth distances. The facial feature extraction process depended on principal component analysis (PCA) and local binary patterns (LBP), among other techniques, to extract data, which then required classification [8]. These techniques demonstrated exciting progress until they failed to deliver expected results when faced with shifting lighting conditions, different head positioning, and partial obscuration of specific face areas.

With the introduction of deep learning, it has become the backbone of artificial intelligence as we know it today. It has found its way to FER and gave significant results because the features are extracted as part of the deep learning structure, and this made the network immune to the noise available in the images. One of the most commonly used deep learning paradigms is convolutional neural network (CNN) [9]–[12]. However, the standard architecture of a CNN for classification typically ends with one or more fully connected layers, followed by a softmax activation function. While powerful, this approach has known limitations such as overconfidence in predictions. The softmax produces a probability distribution that always sums to 1, often yielding high confidence even for ambiguous or out-of-distribution samples [13]. The second limitation is the fact that the fully connected layers are considered as black boxes, making any attempt to explain the solution impossible. Finally, CNNs are unable to handle ambiguity and uncertainty, which is especially concerning in the FER dataset, where an expression can lie between "happy" and "surprise," or be a blend of multiple emotions. The crisp, determined nature of softmax does not model this uncertainty well, resulting in lower-than-expected testing accuracy.

In this paper, the fuzzy min-max neural network (FMMNN) [14]–[17] is introduced to replace the softmax layer in the CNN to solve the shortcomings of the softmax fully connected layer. The fuzzy min-max deep neural network (FMMDNN) will use the CNN ability to extract features and will be trained by these features. In the literature, different studies introduced fuzzy min-max (FMM) in deep learning [16]–[18]. In this paper, the learning ability of FMMNN is used as the fully connected network in the CNN. The details of the proposed network are presented in the next section. The proposed model led to a model that is more interpretable, better at quantifying uncertainty, and potentially more robust to the noisy and ambiguous labels found in datasets like FER2013.

The remainder of the paper is organized as follows. Section 2 presents the methodology that contains the CNN network along with the FMM network in order to create the FMMDNN. The results and discussion are presented in section 3. Lastly, section 4 presents the conclusion of this work.

2. METHOD

One of the most used deep learning neural networks is the CNN, this network automatically extracts the features without any human supervision [19]. In CNNs, features are learned directly from the data during the training process, reducing the need for manual preprocessing of the input data. In fact, this feature is for deep learning techniques in general. CNNs learn hierarchical representations of data, starting from simple features (like dots and lines) in the initial layers to more complex features (like shapes and objects) in deeper layers. This allows CNNs to effectively capture complex patterns in data like images and videos. Figure 1 presents the different layers involved in building a CNN [19].

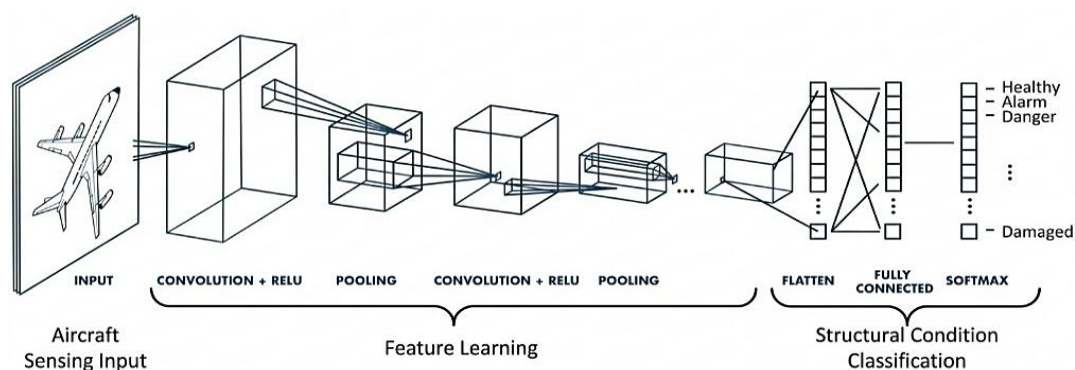


Figure 1. Hierarchical structure of CNNs

In addition, CNNs reuse the same filters across the entire image, significantly reducing the number of features to be learned compared to traditional neural networks. Through their pooling layers, CNNs can recognize patterns regardless of their position in the image. This is particularly useful for tasks like object detection and image classification. They also reduce the size of data representations, decreasing computational complexity and increasing efficiency. Another important feature is parallelization, as CNNs can process multiple pixels simultaneously due to their architecture, leading to faster training times.

compared to sequential data processing methods. This feature is a significant advantage when dealing with large datasets [19].

In a CNN, the input layer first receives the raw data (the main type of input data) provided to the model. This layer simply passes the image to the next layer without any computations, but it must be adjusted to accommodate the image properties (such as height, width, and number of channels) as these properties must be uniform. Then, the image is fed into the convolution layer, which performs a convolution operation between the image and one or more filters (also called kernels) to result in a feature map containing the automatically extracted features. The input images can have one or more channels, and the filters have the same number of channels (each filter has a channel for each channel in the input image). The image and the filter, meaning each channel in the image with its corresponding channel in the filter, are considered. Both the image and the filter are treated as two-dimensional matrices, where the image matrix represents pixel values, and the filter matrix contains weight values. The filter matrix is typically much smaller in height and width. In the convolution operation, the filter slides over the image one stride at a time from the beginning of the image (the top-left corner) to the end of the row, then starts again from the beginning of the row one stride below the previous position, and continues until it has gone through the entire image, as shown in Figure 2.

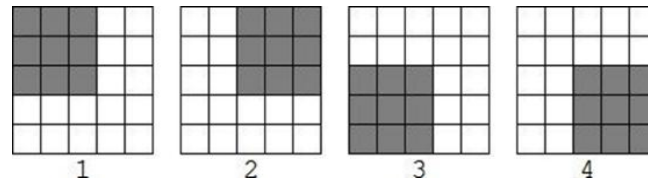


Figure 2. The positions of a 3×3 filter on a 5×5 image with a 2×2 stride

This requires precision in calculating the dimensions of both the filter and the stride so that they fit the input image dimensions. Assuming that the dimensions of the input image are $I_h \times I_w$, the dimensions of the filter are $F_h \times F_w$, and the dimensions of the stride are $S_h \times S_w$, the feature map will be a $M_h \times M_w$ matrix as shown in (1) and (2).

$$M_h = \frac{I_h - F_h}{S_h} + 1 \quad (1)$$

$$M_w = \frac{I_w - F_w}{S_w} + 1 \quad (2)$$

These two amounts must be natural numbers. Otherwise, padding the input image may be needed (change its dimensions by adding zeros around it). Additionally, each stride dimension must not be greater than its corresponding filter dimension to ensure that all parts of the input image are passed through. Each entry in the mentioned matrix corresponds (in order) to one of the positions through which the filter passes. At each position the filter passes through during its striding, an entry-wise product is performed between the filter and the corresponding patch of the image. The entry-wise product operation between two matrices of identical dimensions is performed as (3).

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & a_{13}b_{13} \\ a_{21}b_{21} & a_{22}b_{22} & a_{23}b_{23} \\ a_{31}b_{31} & a_{32}b_{32} & a_{33}b_{33} \end{bmatrix} \quad (3)$$

Then, all the values resulting from the entry-wise product operations are summed with a different value for each filter, called the bias, to produce the corresponding entry of the feature map. So, if X is the input image, W is the filter, and b is the bias, then every entry of the feature map Y is computed as in (4).

$$Y_{m,n} = \sum_{i,j,k} (W_{i,j,k})(x_{m+i,n+j,k}) + b \quad (4)$$

The output of the convolution layer is a set of feature maps, one for each filter. These feature maps highlight the different features learned from the input image. On the other hand, the filters are learnable parameters (like weights in artificial neural networks). The multiple filters learn different patterns

(detect different features). An activation function is applied after the convolution layer, adding non-linearity to the model. One common activation function used in CNNs is the sigmoid (σ), which is used in the context of binary classification and probabilistic interpretation tasks, where it transforms its input into an output in the range of 0 to 1, based on (5).

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (5)$$

Of course, the inputs to these functions are the feature maps from the previous convolution layers, which are matrices. So, these functions are applied to each entry of the input matrix. The output is then passed to the pooling layer, which reduces the size of the data to decrease the complexity of the model and increase its efficiency and resistance to small variations in the input images. Max pooling, which selects the maximum value from a specific region (called a window), is commonly used. The concept of stride is also present here, with the only difference being that the window replaces the filter, as shown in Figure 3.

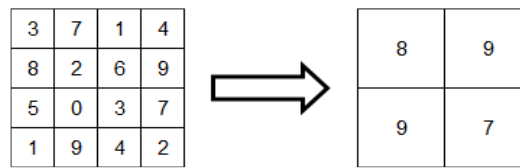


Figure 3. Max pooling with a 2×2 window and a 2×2 stride

A sequence of convolution layer, activation function, and pooling layer is usually repeated more than once in CNNs, immediately after the input layer. But instead of the raw images, the input of each subsequent convolution layer is the output of its previous layer, which will be one or more feature maps. Additionally, multiple feature maps are treated as channels of a single input image when fed into a convolution layer. After them comes the role of the flattening layer to convert the multi-dimensional input matrices into vectors, as shown in Figure 4.

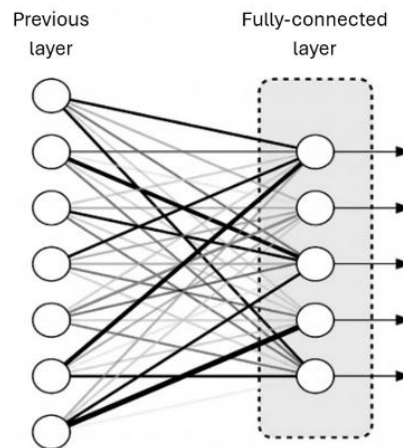


Figure 4. The fully connected layer

In this paper, the fully connected network is a FMMNN, replacing the original softmax network. The FMMNN requires a fixed-size feature vector in order to ensure consistent results within the dataset. For this to be implemented, a feature vector of 256 is used. This is done by ensuring that the global average pooling (GAP) layer has a size of 256, as shown in Table 1.

The feature vector has a fixed length of 256, and this is fed to the FMM for training. The FMM was introduced in 1992 [14], [15]. FMM creates fuzzy hyperboxes as decision regions. Each class is represented by one or more hyperboxes in the feature space depending on a predefined hyperbox size. Hyperboxes are described by minimum and maximum points. A membership function then quantifies how much a given

sample belongs to each hyperbox, hence to each class. The membership value for an input pattern is said to be one if it is fully contained within the hyperbox. The definition of each hyperbox fuzzy set B_j is as in (6).

$$B_j = \{X, V_j, W_j, f(X, V_j, W_j)\} \forall X \in I^n \quad (6)$$

Where W_j and V_j are max and min points, respectively. A hyperbox in three dimensions is shown in Figure 5.

Table 1. Detailed configuration of the CNN feature extractor

Layer type	Kernel/pool size	Stride	Output shape	Parameters	Purpose
Input	N/A	N/A	48×48 (1 feature map)	0	Input image (grayscale)
Conv1	3×3	1	48×48 (32 feature map)	320	Initial feature detection
Conv2	3×3	1	48×48 (32 feature map)	9,248	Deepens feature map
MaxPool1	2×2	2	24×24 (32 feature map)	0	Downsampling and translation invariance
Conv3	3×3	1	24×24 (64 feature map)	18,496	Increased feature complexity
Conv4	3×3	1	24×24 (64 feature map)	36,928	Deepens feature map
MaxPool2	2×2	2	12×12 (64 feature map)	0	Further downsampling
Conv5	3×3	1	12×12 (128 feature map)	73,856	High-level feature extraction
Conv6	3×3	1	12×12 (128 feature map)	147,584	Deepens feature map
MaxPool3	2×2	2	6×6 (128 feature map)	0	Final spatial reduction
Conv7	3×3	1	6×6 (256 feature map)	295,168	Final feature map generation
GAP	6×6	N/A	256 feature map	0	Interface to FMMNN 256 feature map

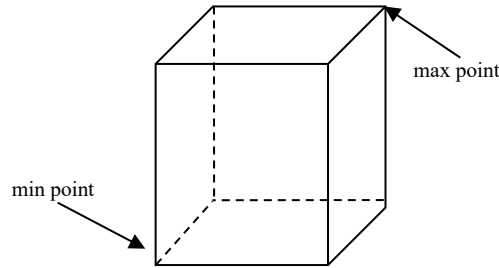


Figure 5. A three-dimensional hyperbox

The combined fuzzy set classifying the K^{th} pattern class, C_k , is defined by applying the definition of a hyperbox fuzzy set as in (7).

$$C_k = \cup_{j \in K} B_j \quad (7)$$

Where K is the number of set hyperboxes linked to class k . The index set of those hyperboxes associated with class k . A key aspect of this method involves identifying and adjusting class boundaries, as depicted in Figure 6.

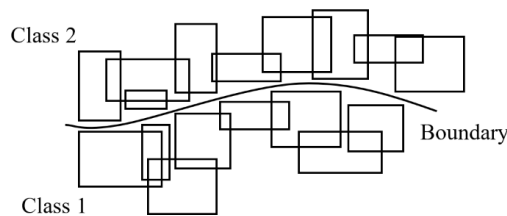


Figure 6. The decision boundary created by FMM

In FMM, if two hyperboxes of the same class overlap, nothing will happen, on the other hand if overlapping occurs between different classes, then a contraction process is initiated. The degree to which a pattern belongs to a hyperbox is measured by the membership function $b_j(H_h)$. The membership function is defined as (8).

$$b_j(A_h) = \frac{1}{2n} \sum_{i=1}^n [\max(0, 1 - \max(0, \gamma \min(1, a_{hi} - w_{ji}))) + \max(0, 1 - \max(0, \gamma \min(1, v_{ji} - a_{hi})))] \quad (8)$$

The contraction and expansion processes use the learning algorithm in FMM. The learning process in FMM is a supervisor process where a pair of input vectors and an associated class is given $\{X_h, d_h\}$. The learning process begins by creating new hyperboxes for each class, as the training patterns are presented to the network, either new hyperboxes are created or the current hyperboxes are expanded and contracted. This is controlled by the size of the hyperbox and the number of classes the data is associated with. The expansion criteria is restricted by a requirement that as in (9).

$$n\theta \geq \sum_{i=1}^n (\max(w_{ji}, x_{hi}) - \min(v_{ji}, x_{hi})) \quad (9)$$

Where $0 \leq \theta \leq 1$ is the hyperbox size. Should no hyperbox be found to meet the expansion criteria, a new hyperbox is created in the network. Classes can be formed that aren't linearly separable through this growth process. New classes can be added without retraining. Overlapping is a possibility when expansion is presented. To solve this issue, an overlap test is conducted, and if the overlap occurred between different classes, then contraction is done; otherwise, there is no need to perform any operation. Algorithm 1 shows the pseudo code for the FMMNN.

Algorithm 1. Pseudo code for the FMMNN

```

Step 1: Initialization
Initialize an empty hyperbox set  $\mathcal{H} = \emptyset$ 
Step 2: For each training sample  $(x, c) \in D$ 
    Membership Evaluation: Compute membership value of  $x$  to all hyperboxes in  $\mathcal{H}$  that
    belong to class  $c$ 
    Hyperbox Selection: Select the hyperbox  $h^*$  with the highest membership value
    Expansion Check: If expanding  $h^*$  with  $x$  satisfies the size constraint:
         $\max(v_j^{new} - u_j^{new}) \leq \theta \forall j$ 
        then expand the hyperbox, else, create a new hyperbox with
             $u = x, v = x$ 
    Overlap Test: Check for overlaps between the expanded hyperbox and hyperboxes
    of other classes
    Contraction: If overlap exists, apply contraction rules to eliminate overlap
    while minimizing distortion
Step 3: Repeat
Continue until all samples are processed
Step 4: Return
Return the final set of hyperboxes  $\mathcal{H}$ 

```

When designing a FMMNN there are several hyperparameters that need to be chosen. The most critical hyperparameter is the hyperbox size θ . As mentioned above θ can be any value from 0 to 1. When θ is very small (a value close to 0), the network will tend to create a hyperbox for each input sample, this will cause overfitting in the network. On the other hand, when θ is large, the network will be smaller and faster to learn but loses the ability to correctly classify patterns in other words, the network is overfitted. The most commonly used size for the hyperboxes is 0.05, this size is considered moderate, so the network can learn, and hyperboxes will be created, expanded, and contracted accordingly. The second hyperparameter is the sensitivity parameter γ . The parameter is used in the membership function; it controls how fast the membership value decreases when the input lies outside a hyperbox boundary. The value of γ varies between 0.5 and 5. In this paper $\gamma = 1$, since this is the typical value chosen in the literature.

3. RESULTS AND DISCUSSION

The experiments conducted were based on the FER2013 dataset, which was introduced as part of an international conference on machine learning representation-learning challenge and made public in 2013 [20]. The dataset contains 35,887 grayscale face images sized 48×48 pixels, labelled with seven emotion classes: anger, disgust, fear, happiness, sadness, surprise, and neutral. Images were gathered from the web and automatically registered so that faces are approximately centered. The original description and dataset statistics (per-class counts) are reported in the challenge paper/announcement. FER2013 became a de facto benchmark for deep learning FER work because it is large, realistic, and easily accessible. These qualities made it a common testbed for convolutional architectures, transfer learning, and data-augmentation strategies [21]. To further improve the dataset, avoid the risk of overfitting, which is a common issue with datasets trained on limited datasets, and to enhance the model's robustness to variations in facial poses and

lighting, a set of data augmentation techniques is applied to the training set [22]. Table 2 shows the geometric and photometric transformations applied randomly to each training image after each epoch.

Table 2. Augmentation techniques applied

Augmentation technique	Parameter/range	Purpose
Random horizontal flip	P = 0.5	Simulates faces looking in the opposite direction, increasing pose invariance
Random rotation	±10 degrees	Accounts for slight head tilts and minor pose variations
Random width/height shift	± 0.1 (10% of total)	Simulates minor misalignments in face detection/cropping
Random zoom	0.9 to 1.1	Simulates variations in distance from the camera
Brightness adjustment	± 0.2	Increases robustness to varying lighting conditions

The augmentation will not affect the total number of images in the dataset, instead, every time an image is presented in a new epoch, a random transformation is applied to this image. This guarantees the CNN never sees the same image twice. The dataset was divided into two sets, 80% of the data was used for training (28,709 samples, divided in even proportions between seven classes) and 20% of the data for testing. Several tools and techniques were used in the model training process; anaconda integrated development environment (IDE) was used to implement the FMMDNN. OpenCV library was used to include the Haar cascade algorithm, and the fully connected network was replaced by the FMMNN. The FMMNN was set to a hyperbox size of 0.05. The model was trained and tested 10 times, the confusion matrix for the average results is shown in Table 3.

Table 3. Confusion matrix for the FER2013 dataset

True/Predicted	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral	Recall (%)
Angry	574.8	47.9	124.54	28.74	57.48	28.74	95.8	60
Disgust	5.55	67.71	6.66	2.22	15.54	3.33	9.99	61
Fear	112.64	30.72	614.4	30.72	112.64	61.44	61.44	60
Happy	53.22	17.74	35.48	1596.6	35.48	17.74	17.74	90
Sad	137.17	24.94	87.29	37.41	773.14	37.41	149.64	62
Surprise	33.28	16.64	91.52	24.96	16.64	632.32	16.64	76
Neutral	135.52	36.96	24.64	36.96	73.92	36.96	887.04	72
Precision (%)	42	67	48	90	48	78	53	

The confusion matrix reveals distinct patterns of misclassification that are common in the FER systems, this provides a deeper understanding of the model's decision boundary. The model shows that the happy emotion has the highest accuracy since around 25% of the images are of happy emotion this gave the model large number of training samples. On the other hand, disgust had the lowest number of samples in the dataset with a percentage of 1.5% of the 35887 images. The model classified disgust with an accuracy of 61% which is similar to the other emotions. This shows that the model is not affected by the low number of samples, this is due to the use of augmentation in the dataset.

To further analyze the performance of the model recall, precision, and F1-score are used. These quantities are defined as follows.

- i) Recall: the ratio of correct predictions for a specific emotion to the total number of images that actually belong to that emotion. Recall is defined as (10).

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (10)$$

- ii) Precision: the ratio of correct predictions for a specific emotion to the total number of predictions that were classified as belonging to that emotion [23]. Precision is defined as (11).

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (11)$$

- iii) F1-score: represents the harmonic mean of precision and recall, and is used as a performance metric to evaluate classification models with imbalanced dataset. It is defined as (12).

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (12)$$

Table 4 shows the precision, recall, and F1-score for the FER dataset. The F1-score is used to evaluate the performance of classification models. Since the FER2013 dataset is an imbalanced dataset, the results in the confusion matrix will not reflect the full story behind the results. The proposed model achieves exceptional performance on happy (90% recall, 90% precision, and 90% F1-score) and strong performance on surprise (76% recall, 78% precision, and 77% F1-score). These emotions are characterized by highly distinct and easily recognizable facial action units (AUs): for instance, happy has cheek raiser (AU6) and lip corner puller (AU12), while surprise has inner brow raiser (AU1), outer brow raiser (AU2), and upper lid raiser (AU5). The CNN feature extractor successfully isolates these unique features, resulting in a well-separated hyperbox in the FMMNN feature space.

Table 4. Classification results of the proposed model

Emotion	Precision (%)	Recall (%)	F1-score (%)
Angry	42	60	49
Disgust	67	61	64
Fear	48	60	53
Happy	90	90	90
Sad	48	62	54
Surprise	78	76	77
Neutral	53	72	61

One can also notice from Tables 1 and 2 that the fear and surprise emotions have significant overlap, this is obvious since the model misclassified 91.52 fear samples as surprise samples. This can be justified by looking at the biological aspects of both fear and surprise emotions. Both emotions share key AUs, notably AU1 and AU2, which cause the eyes to widen. The primary difference lies in the mouth shape (stretched in fear, dropped jaw in surprise). The low-resolution 48×48 images, combined with the inherent subtlety of these differences, lead to overlapping feature vectors in the 256 space. This in turn suggests that the FMMNN hyperboxes for fear and surprise are adjacent or slightly intersect in the feature space, reflecting the ambiguity of the input data itself.

The classes angry (42% precision), fear (48% precision), and sad (48% precision) exhibit the lowest precision scores. This indicates that the model frequently misclassifies other emotions as these three. A closer look at the angry emotion one can notice that the model misclassifies 137.17 sad samples and 112.64 fear samples as angry. This is likely due to the shared presence of brow lowerer (AU4) in both angry and sad expressions, and the general intensity of negative emotions. Similarly, in the sad emotion the model misclassifies 149.64 neutral samples as sad. This is a common challenge, as a "neutral" face with slight fatigue or relaxation can be easily confused with a low-intensity "sad" expression, especially in the absence of dynamic information.

To further the overall accuracy of the system compared with accuracies from the literature are shown in Table 5. The results show that the proposed deep FMM classification system gave results that are very near to the results published in the literature. This shows that when other versions of the FMMNN are used the results will improve, as the newer versions of the network have higher accuracy and have the ability to explain the obtained results.

Table 5. Performance comparison between obtained results and published results

Paper/Model	Test accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
ResNet152 [20]	39	69	68	68
VGG-19 [20]	44	71	72	70
Transfer learning (AlexNet) [24]	66	-	-	-
Deeper CNN [25]	75	-	-	-
Residual attention (RAN) [26]	72	-	-	-
Self-cure network (SCN) [27]	89	-	-	-
FMMDNN	71	61	69	64

As explained earlier, the paper introduces the FMMNN instead of the softmax layer, the main advantage of using the FMMNN is the geometric interpretability and a robust mechanism for uncertainty quantification in the classification process. The CNN is considered a black box since weight matrices are opaque. The FMMNN on the other hand, introduces geometrical interpretability to handle the uncertainty of the classification process. Each class is presented at least one hyperbox, and a decision boundary is created between these hyperboxes. Furthermore, since the hyperboxes are bounded by minimum and maximum points fuzzy if-then rules can be extracted to justify the research decisions.

In the softmax classifier uncertainty is inferred from the probability distribution, however, in FMMNN, uncertainty is explicitly quantified by the fuzzy membership function, which quantifies the degree to which an input feature vector belongs to a hyperbox. To further illustrate this point, two samples are studied from the dataset shown in Table 6. The first sample is a clear sample with a true class “happy”, this sample the softmax probability was 0.98 and the FMMNN membership was 0.99 for the maximum membership. The second sample was an ambiguous sample, in this sample the softmax gave a maximum probability of 0.55, while the FMMNN membership was 0.48 for the correct class (fear) and 0.45 for wrong feature. This high competition between the classes provides a more transparent measure of uncertainty.

Table 6. Comparison between softmax and FMMNN uncertainty handling

Sample	True class	SoftMax max probability	FMMNN max membership	FMMNN second max membership
Clear sample	Happy	0.98 (happy)	0.99 (happy)	0.01 (neutral)
Ambiguous sample	Fear	0.55 (fear)	0.48 (fear)	0.45 (surprise)

4. CONCLUSION

This paper presented a deep learning neural network based on CNN and FMM. The softmax layer in the CNN was replaced by an FMMNN. The proposed network will use the extracted features from the CNN and create and refine hyperboxes. The created hyperboxes are then expanded and contracted according to their associated classes. The proposed network is tested on the FER dataset. The results obtained show that the proposed network can detect emotions in pictures with good accuracy. Furthermore, other types of FMM neural networks can be used to improve the results in future work, and other datasets can be used to train the proposed network.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Anas Quteishat	✓	✓	✓	✓			✓	✓	✓	✓		✓	✓	
Mohammad Hassan		✓			✓	✓		✓		✓	✓			
Ashraf Al Sharah	✓		✓	✓		✓				✓	✓			
Ahmed Qutaishat						✓		✓		✓				

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are openly available in Kaggle at <https://www.kaggle.com/datasets/msambare/fer2013>.

REFERENCES

- [1] J. Pustejovsky and N. Krishnaswamy, “Embodied human computer interaction,” *KI - Künstliche Intelligenz*, vol. 35, no. 3–4, pp. 307–327, Nov. 2021, doi: 10.1007/s13218-021-00727-5.
- [2] Z. Lyu, “State-of-the-art human-computer-interaction in metaverse,” *International Journal of Human-Computer Interaction*, vol. 40, no. 21, pp. 6690–6708, Nov. 2024, doi: 10.1080/10447318.2023.2248833.

- [3] Z. Lv, F. Poiesi, Q. Dong, J. Lloret, and H. Song, "Deep learning for intelligent human-computer interaction," *Applied Sciences*, vol. 12, no. 22, Nov. 2022, doi: 10.3390/app122211457.
- [4] J. Liu, "ChatGPT: perspectives from human-computer interaction and psychology," *Frontiers in Artificial Intelligence*, vol. 7, Jun. 2024, doi: 10.3389/frai.2024.1418869.
- [5] I. Nafisah *et al.*, "Deep learning-based feature selection for detection of autism spectrum disorder," *Frontiers in Artificial Intelligence*, vol. 8, Jun. 2025, doi: 10.3389/frai.2025.1594372.
- [6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. Cambridge, United Kingdom: MIT Press, 2016.
- [7] S. AlZu'bi *et al.*, "A novel deep learning technique for detecting emotional impact in online education," *Electronics*, vol. 11, no. 18, Sep. 2022, doi: 10.3390/electronics11182964.
- [8] M. Khan *et al.*, "A deep learning approach for facial emotions recognition using principal component analysis and neural network techniques," *The Photogrammetric Record*, vol. 37, no. 180, pp. 435–452, Dec. 2022, doi: 10.1111/phor.12426.
- [9] H. S. Gill and B. S. Khehra, "An integrated approach using CNN-RNN-LSTM for classification of fruit images," *Materials Today: Proceedings*, vol. 51, pp. 591–595, 2022, doi: 10.1016/j.matpr.2021.06.016.
- [10] S. Alghyaline, "Optimised CNN architectures for handwritten Arabic character recognition," *Computers, Materials & Continua*, vol. 79, no. 3, pp. 4905–4924, 2024, doi: 10.32604/cmc.2024.052016.
- [11] Y. Zhang *et al.*, "Carrying out CNN channel pruning in a white box," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 7946–7955, Oct. 2023, doi: 10.1109/TNNLS.2022.3147269.
- [12] K. Shaheed *et al.*, "DS-CNN: A pre-trained Xception model based on depth-wise separable convolutional neural network for finger vein recognition," *Expert Systems with Applications*, vol. 191, Apr. 2022, doi: 10.1016/j.eswa.2021.116288.
- [13] T. Pearce, A. Brintrup, and J. Zhu, "Understanding softmax confidence and uncertainty," 2021, *arXiv:2106.04972*.
- [14] P. K. Simpson, "Fuzzy min-max neural networks - part 1: classification," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 776–786, 1992, doi: 10.1109/72.159066.
- [15] P. K. Simpson, "Fuzzy min-max neural networks - part 2: clustering," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, Feb. 1993, doi: 10.1109/TFUZZ.1993.390282.
- [16] T. R. Chavan and A. V. Nandedkar, "A convolutional fuzzy min-max neural network," *Neurocomputing*, vol. 405, pp. 62–71, Sep. 2020, doi: 10.1016/j.neucom.2020.04.003.
- [17] W. Huang, M. Sun, L. Zhu, S.-K. Oh, and W. Pedrycz, "Deep fuzzy min-max neural network: analysis and design," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 8229–8240, Jun. 2024, doi: 10.1109/TNNLS.2022.3226040.
- [18] M. Kalbhor, S. Shinde, D. E. Popescu, and D. J. Hemanth, "Hybridization of deep learning pre-trained models with machine learning classifiers and fuzzy min-max neural network for cervical cancer diagnosis," *Diagnostics*, vol. 13, no. 7, Apr. 2023, doi: 10.3390/diagnostics13071363.
- [19] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, "Review on convolutional neural networks (CNN) in vegetation remote sensing," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 173, pp. 24–49, Mar. 2021, doi: 10.1016/j.isprsjprs.2020.12.010.
- [20] P. Giannopoulos, I. Perikos, and I. Hatzilygeroudis, "Deep learning approaches for facial emotion recognition: a case study on FER-2013," in *Advances in Hybridization of Intelligent Methods*, Cham, Switzerland: Springer, 2018, pp. 1–16, doi: 10.1007/978-3-319-66790-4_1.
- [21] T. Kanade, J. F. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," in *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, 2000, pp. 46–53, doi: 10.1109/AFGR.2000.840611.
- [22] M. R. Hassan *et al.*, "Integrating deep learning models and data augmentation techniques for improved breast cancer detection," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 16, no. 2, pp. 517–534, Jun. 2025, doi: 10.58346/JOWUA.2025.12.031.
- [23] S. Du, Y. Tao, and A. M. Martinez, "Compound facial expressions of emotion," *Proceedings of the National Academy of Sciences*, vol. 111, no. 15, pp. E1454–E1462, Apr. 2014, doi: 10.1073/pnas.1322355111.
- [24] S. Ramalingam and F. Garzia, "Facial expression recognition using transfer learning," in *2018 International Carnahan Conference on Security Technology (ICCST)*, Oct. 2018, pp. 1–5, doi: 10.1109/CCST.2018.8585504.
- [25] O. Arriaga, M. V. -Toro, and P. G. Plöger, "Real-time convolutional neural networks for emotion and gender classification," in *ESANN 2019 - Proceedings, 27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2019, pp. 221–226.
- [26] T. Shen and H. Xu, "Facial expression recognition based on multi-channel attention residual network," *Computer Modeling in Engineering & Sciences*, vol. 135, no. 1, pp. 539–560, 2023, doi: 10.32604/cmescs.2022.022312.
- [27] K. Wang, X. Peng, J. Yang, S. Lu, and Y. Qiao, "Suppressing uncertainties for large-scale facial expression recognition," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020, pp. 6896–6905, doi: 10.1109/CVPR42600.2020.00693.

BIOGRAPHIES OF AUTHORS



Anas Quteishat    received his B.Sc. in Electronic Engineering from Princess Sumaya University for Technology, Jordan in 2003. In 2005 he finished his M.Sc. in Electronic System Design from the Universiti Sains Malaysia. He got his Ph.D. in 2009 in Computational Intelligence from the Universiti Sains Malaysia. He is head of the Department of Communication and Computer Engineering, Al-Ahliyya Amman University. Currently, he is an associate professor at Al-Balqa Applied University. His research interests include artificial neural networks, pattern recognition, intelligent systems, and embedded systems. He has published many research papers in international journals. He can be contacted at email: anas.quteishat@bau.edu.jo.



Mohammad Hassan    is the vice dean of the Faculty of Engineering at Al-Ahliyya Amman University. He has completed his Ph.D. from Baku State University, Azerbaijan. He is an associate professor in the Department of Communications and Computer Engineering at the Faculty of Engineering, Al-Ahliyya Amman University. He is a member of the Jordanian Engineering Association. He has published numerous research papers in various journals and conferences, covering topics such as machine learning, computer networks, intelligent transportation systems, and mobile learning adaptation models. He can be contacted at email: mhassan@ammanu.edu.jo.



Ashraf Al Sharah    has completed his Ph.D. from Tennessee State University, United States. He was a research associate at the CyberViz Research Lab and served as an assistant professor in the Department of Computer Engineering at Al-Ahliyya Amman University. He is now serving as an assistant professor in the Department of Electrical Engineering in Al-Balqa Applied University. His research interests include wireless security, IoT, smart attacks, AI, and game theory. He can be contacted at email: aalsharah@bau.edu.jo.



Ahmed Qutaishat    finished his master's degree from Universiti Utara Malaysia in intelligent systems in 2007. He joined Sohar University in 2011. From 2012 until 2017, he worked as a coordinator in the General Foundation Program at Sohar University. His research area focuses on artificial intelligence, machine learning, and genetic algorithms. He has published research articles in different national and international journals and conferences. He can be contacted at email: aqtaishat@su.edu.om.