Human Emotion Detection Using Convolutional Neural Networks with Hyperparameter Tuning

Aparna Chaparala

1 Introduction

Human emotion recognition has been an active research area for the past few years, due to the increasing demand for applications in perceptual and cognitive sciences and affective computing. It has become an essential component for fields such as computer animations, sociable robots, and neuromarketing. Human emotions can be recognized by using facial expressions and vocal tones. According to Kaulard et al. [1], nonverbal components convey two-thirds of human communication, while verbal components convey only one-third. Various kinds of data including physiological signals, such as electromyograph (EMG), electrocardiogram (ECG), and electroencephalograph (EEG), can also be considered as input for the emotion recognition process. Among these, the facial image is the promising input type as it is noninvasive and provides an ample amount of information for expression recognition. Emotions can be categorized into three types: basic emotions (BEs), compound emotions (CEs), and micro-expressions (MEs). Basic emotions cover neutral, anger, disgust, fear, surprise, sadness, and happiness.

Two categories of approaches for facial expression recognition (FER) are in use: conventional approaches and deep learning-based approaches. When compared to deep learning-based techniques, conventional techniques are advantageous as they require less computational power. Hence, no additional infrastructure is needed. Input images having illumination changes, occlusion, and deflection of the head may influence the face detection task performance and reduce the accuracy of FER. Conventional techniques are not suitable for noisy input data. Deep learning-based techniques address these issues. Of late, convolutional neural networks (CNNs) were proven effective for face detection [2]. As CNNs contain deep layers and use

A. Chaparala (✉)
RVR&JC College of Engineering (A), Chowdavaram, Guntur, AP, India
e-mail: aparna@rvrjc.ac.in

R. R. Raje et al. (eds.), *Artificial Intelligence and Technologies*,
Lecture Notes in Electrical Engineering 806,
https://doi.org/10.1007/978-981-16-6448-9_42

elaborate designs, they can ably handle noisy data automatically [3]. CNNs proved to exhibit better performance than conventional methods for the FER task [4, 5]. The performance of CNN highly depends on the choice of its hyperparameters. It is possible to enhance the CNN's performance by optimizing hyperparameters such as the number of hidden layers, units in each layer, filters, size of the filter, batch size, and learning rate. The present work considers the optimization of hyperparameters that describe the CNN structure. Grid search and random search techniques are commonly used for this purpose [6]. Each of these techniques has its limitations, and both need more time and domain expertise for identifying ideal hyperparameter values. Metaheuristic-based approaches can address these shortcomings as they are stochastic approximation methods. The present work employed the differential evolution (DE) algorithm for tuning the selected hyperparameters.

2 Related Work

Kim et al. [7] proposed to train multiple CNNs. They have shown an improvement in training by changing the network topology and random weight initialization. An interesting method for selecting the CNN structure was presented by Gao et al. [8]. They proposed gradient priority particle swarm optimization (GPSO) with gradient penalties for tuning CNN architecture. Experimental results have shown that the proposed method has gained competitive prediction performance for the emotion recognition task. Bergstra and Bengio [6] proposed to employ a grid or random search for tuning hyperparameters. Since the number of hyperparameters is large, testing is computationally expensive. Snoek et al. [9] have addressed the limitations of trial and error-based techniques for hyperparameter optimization. They have proposed a Bayesian optimization framework. Bochinski et al. [10] have shown that evolutionary algorithms can outperform the existing hyperparameter optimization methods.

3 Methodology

Benchmark dataset for facial expression recognition is split into training set (TS) and testing set (TE). For ensuring that samples of all classes get selected, stratified sampling without replacement is used. Selected samples from TS generate tuning set (TUS). Tuning set is further divided into TUS1 and TUS2. TUS1 is used for hyperparameter optimization. TUS2 is used for validating the outcome of optimization. Differential evolution is performed until the termination condition is met. CNN is trained using the outcome of DE on TS. The holdout method is used for assessing the performance of the trained model. After training, the model's performance is assessed by using TE. Table 1 specifies the architecture of the convolutional neural network used in the present work (Fig. 1).

Table 1 Configuration of convolutional neural network

Convolutional layers	Six convolutional layers are used with a filter size of 64 for first two layers, 128 for next two layers, and 256 for the last two layers. The kernel size is set to 3 × 3 for all convolutional layers. ReLU activation function is used
Max pooling layer	Two max pooling layers are used. First layer after 2 convolutional layers and second layer after next four convolutional layers with a dropout rate of 20%. Each layer is two dimensional and uses a pool size of 2 × 2
Fully connected layers	Two fully connected layers are used. Flattened output of previous layers is given as input to first fully connected layer. A dropout rate of 40% is used. The second fully connected layer with a softmax activation function is the output layer

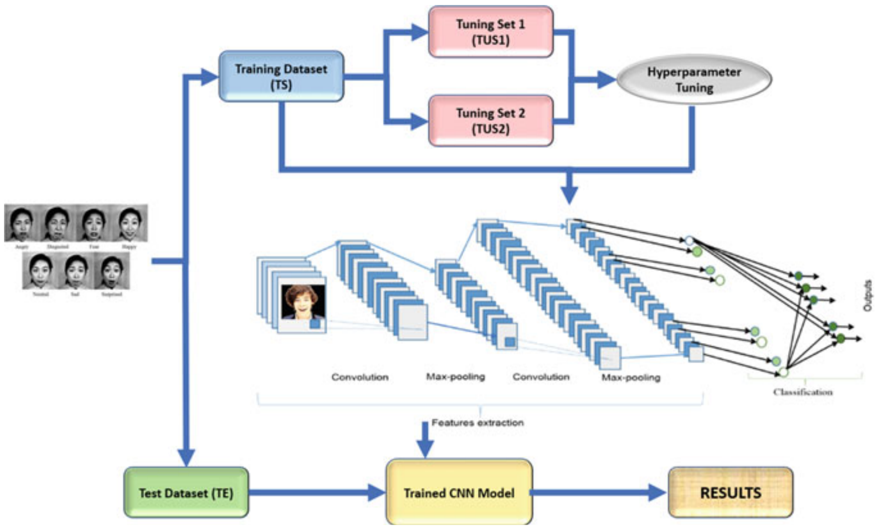

Fig. 1 Architecture of the proposed model

Hyperparameter Tuning—Metaheuristic optimization techniques proved to yield better results when the search space is large and complex [11]. Since the number of hyperparameters is large in CNN, tuning them is computationally expensive. Hence, the proposed model determines the optimal network topology by using the differential evolution (DE) algorithm. A simple, yet powerful, population-based stochastic search technique, differential evolution (DE) [12], has gained much attention and a wide range of successful applications [13, 14], due to its simplicity, ease in the implementation, and quick convergence.

The hyperparameters considered for tuning using DE include number of convolutional layers, filter size, stride, dropout rate, and batch size. A vector comprising the above-mentioned parameters is used as a chromosome for the DE algorithm. Precision and recall values for each of the six basic emotions are

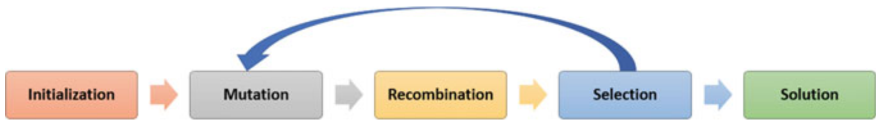

Fig. 2 Differential evolution algorithm scheme

calculated by using the confusion matrix. Fitness function is defined as F = AvgPrec + AvgRec, where AvgPrec is the average of precision values computed for each basic emotion. Likewise, AvgRec is the average of recall values. DE aims to improve the existing solution using the techniques of mutation, recombination, and selection. The general paradigm of differential evolution is shown in Fig. 2.

Initialization—Creation of a population of individuals. The *i*th individual vector (chromosome) of the population at current generation *t* with *d* dimensions is as follows

$$Z_i(t) = \left[Z_{i,1}(t), Z_{i,2}(t), \ldots, Z_{i,d}(t)\right] \tag{1}$$

Mutation—A random change of the vector Z_i components. For each individual vector $Z_k(t)$ that belongs to the current population, a new individual, called the mutant individual, *U* is derived through the combination of randomly selected and pre-specified individuals.

$$U_{k,n}(t+1) = Z_{m,n}(t) + F * \left(Z_{i,n}(t) - Z_{j,n}(t)\right) \tag{2}$$

where the indices *m*, *n*, *i*, *j* are uniformly random integers mutually different and distinct from the current index '*k*' and *F* is a real positive parameter, called mutation factor or scaling factor (usually ϵ [0, 1]).

Recombination (*Crossover*)—Merging the genetic information of two or more parent individuals for producing one or more descendants. Binomial crossover is used in the present work. The binomial or uniform crossover is performed on each component n ($n = 1, 2, \ldots, d$) of the mutant individual $U_{k,n}(t + 1)$. For each component, a random number '*r*' in the interval [0, 1] is drawn and compared with the crossover rate (CR) or recombination factor (another DE control parameter), CR € [0, 1]. If $r <$ CR, then the *n*th component of the mutant individual $U_{k,n}(t)$ will be selected; otherwise, the *n*th component of the target vector $Z_{k,n}(t)$ becomes the *n*th component.

$$U_{k,n}(t+1) = \begin{cases} U_{k,n}(t+1), & \text{if } \mathrm{rand}_n(0,1) < \mathrm{CR} \\ Z_{k,n}(t), & \text{otherwise} \end{cases} \tag{3}$$

Selection—Choice of the best individuals for the next cycle. If the new offspring yields a better value of the objective function, it replaces its parent in the next generation; otherwise, the parent is retained in the population, i.e.,

$$Z_k(t+1) = \begin{cases} U_k(t+1), & \text{if } f(U_k(t+1)) > f\ (Z_k(t)) \\ Z_k(t), & \text{if } f(U_k(t+1)) < f\ (Z_k(t)) \end{cases} \tag{4}$$

where f is the objective function to be minimized. It can be inferred that DE is a powerful population-based heuristic search technique that has empirically proven to be very robust for global optimization over continuous spaces. As the number of control parameters in DE is very few compared to other algorithms, DE is effective and efficient and thus can be treated as a widely applicable approach for solving real-world problems [13, 14].

4 Experimentation

For experimentation, two benchmark datasets, CK+ and Japanese Female Facial Expressions (JAFFE), are used.

CK+ Dataset: This dataset has 593 image sequences representing seven basic expressions (happiness, sadness, surprise, disgust, fear, anger, and neutral) of 123 models. Since the work is focused on recognition of six basic expressions, neutral expression images were ignored. Out of 593, 309 sequences have validated emotion labels that belong to one of the six previously mentioned emotions. They were selected by excluding other sequences. From each image sequence, last two frames were selected making a dataset of 618 images.

Japanese Female Facial Expressions (JAFFE): The JAFFE dataset has 213 images of ten female Japanese models. Each image represents one of the seven basic emotions (including neutral emotion). Images pertaining to neutral expression are not used.

The proposed model is implemented using Keras with a TensorFlow back end in Python 3.6. Experiments are conducted on the selected datasets. Seventy percentage of the samples are used for training, and remaining 30% is used for testing. For validating hyperparameter tuning, 20% of the samples from training dataset are used. The samples are selected by using stratified sampling. Tables 2, 3, 4, and 5 show the confusion matrices of the two datasets used with and without hyperparameter tuning. Prediction accuracies for CK+ dataset and JAFFE dataset are depicted in Fig. 3. For both the datasets, optimization of hyperparameters has improved the accuracy of all basic emotions except fear. Fear has least impact of optimization. For JAFFE dataset, accuracy is decreased by 1%. Proposed model has improved the overall classification accuracy by 4.32% for CK+ dataset and 3.78% for JAFFE dataset.

Table 2 Confusion matrix for CK+ dataset without hyperparameter tuning

	Anger	Disgust	Fear	Happy	Sad	Surprise
Anger	0.7437	0.0834	0.0749	0.0428	0.0552	0
Disgust	0.0875	0.7535	0.0526	0.0187	0.0877	0
Fear	0.1476	0	0.7073	0	0.1019	0.0432
Happy	0.0425	0.0521	0.0472	0.7984	0.0582	0.0016
Sad	0.0637	0.0274	0.1126	0	0.7144	0.0819
Surprise	0.0553	0.0897	0	0.0404	0	0.8146

Table 3 Confusion matrix for CK+ dataset with hyperparameter tuning

	Anger	Disgust	Fear	Happy	Sad	Surprise
Anger	0.7943	0.1033	0.1024	0	0	0
Disgust	0.0948	0.7842	0.027	0	0.094	0
Fear	0.02	0	0.728	0	0.102	0.15
Happy	0.032	0.048	0.029	0.865	0	0.026
Sad	0	0.039	0.122	0	0.7451	0.0939
Surprise	0.0658	0.0597	0	0	0	0.8745

Table 4 Confusion matrix for JAFFE dataset without hyperparameter tuning

	Anger	Disgust	Fear	Happy	Sad	Surprise
Anger	0.6227	0.1026	0.0721	0	0.1789	0.0237
Disgust	0.1285	0.6815	0.0928	0	0.0427	0.0545
Fear	0.0098	0	0.6978	0	0.1352	0.1572
Happy	0.0236	0.0594	0.0587	0.7046	0.0216	0.1321
Sad	0.0252	0.1486	0.1734	0	0.6288	0.024
Surprise	0.0245	0.0438	0.0364	0.1058	0.0467	0.7428

Table 5 Confusion matrix for JAFFE dataset with hyperparameter tuning

	Anger	Disgust	Fear	Happy	Sad	Surprise
Anger	0.6543	0.1165	0.0687	0	0.1605	0
Disgust	0.1346	0.7374	0.0834	0	0	0.0446
Fear	0.0198	0	0.6878	0	0.1352	0.1572
Happy	0	0.0484	0.0297	0.8154	0	0.1065
Sad	0	0.1439	0.1839	0	0.6358	0.0364
Surprise	0	0.0361	0.0265	0.1278	0.0351	0.7745

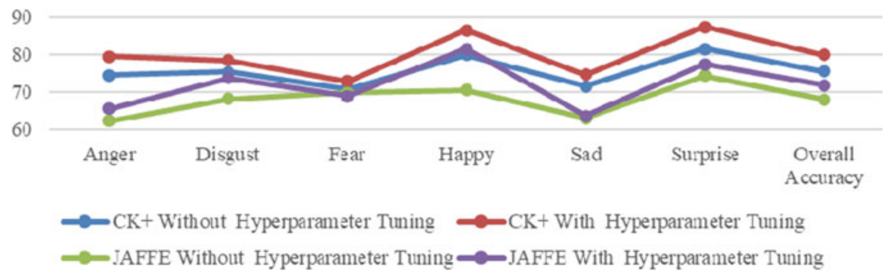

Fig. 3 Classification accuracies for CK+ and JAFFE datasets

5 Conclusion

The present study proposes to optimize the convolutional neural network hyperparameters for improving the human emotion recognition rate from facial expressions. Conventional techniques fail to offer good classification accuracy for noisy input data. As CNNs contain deep layers, they can handle noisy data and are proven suitable for facial expression recognition. However, CNNs demand high computation power making their applicability limited. The performance of CNN highly depends on the choice of its hyperparameters. To enhance the CNN performance for facial expression recognition, its hyperparameters are optimized using the DE algorithm. CK+ and JAFFE datasets are used for assessing the tuned model's performance. The results obtained have shown that hyperparameter tuning has improved the overall accuracy by 4%.

Acknowledgements The author wishes to thank the management of RVR&JC College of Engineering, for funding the present work.

References

1. Kaulard K, Cunningham D, Bülthoff H, Wallraven C (2012) The MPI facial expression database—a validated database of emotional and conversational facial expressions. PLoS ONE 7(3):e32321
2. Li H, Lin Z, Shen X, Brandt J, Hua G (2015) A convolutional neural network cascade for face detection. In: Proceedings of IEEE conference on computer visual and pattern recognition, pp 5325–5334
3. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444
4. Cirean DC, Meier U, Masci J, Gambardella LM, Schmidhuber J (2011) Flexible, high performance convolutional neural networks for image classification. In: Proceedings of the twenty-second international joint conference on artificial intelligence (IJCAI'11), vol 2, AAAI Press, Barcelona, Catalonia, Spain, pp 1237–1242
5. Ko BC (2018) A brief review of facial emotion recognition based on visual information. Sensors 18(2):401

6. Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. J Mach Learn Res 13:281–305
7. Kim BK, Roh J, Dong S-Y, Lee S-Y (2016) Hierarchical committee of deep convolutional neural networks for robust facial expression recognition. J Multimodal User Interfaces 10 (2):173–189
8. Gao Z, Li Y, Yang Y, Wang X, Dong N, Chiang HD (2020) A GPSO-optimized convolutional neural networks for EEG-based emotion recognition. Neurocomputing 380:225–235
9. Snoek J, Larochelle H, Adams RP (2012) Practical bayesian optimization of machine learning algorithms. Adv Neural Inf Process Syst 2951–2959
10. Bochinski E, Senst T, Sikora T (2017) Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. In Proceedings of the 2017 IEEE international conference on image processing (ICIP), Beijing, China, pp 3924–3928
11. Radhika S, Chaparala A (2018) Optimization using evolutionary metaheuristic techniques: a brief review. Brazilian J Oper Prod Manage 15(1):44–53
12. Price K, Storn R (1995) Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces. International Computer Science Institute, Berkeley. Berkeley, CA
13. Sajja R, Rao CS (2014) A new multi-objective optimization of master production scheduling problems using differential evolution. Int J Appl Sci Eng 12(1):75–86
14. Radhika S, Rao CS, Pavan KK (2013) A differential evolution based optimization for Master production scheduling problems. Int J Hybrid Inf Technol 6(5):163–170