Soft Computing (2020) 24:7593–7602
https://doi.org/10.1007/s00500-019-04387-4

METHODOLOGIES AND APPLICATION

Hyperparameter optimization in CNN for learning-centered emotion recognition for intelligent tutoring systems

Ramon Zatarain Cabada[1] · Hector Rodriguez Rangel[1] · Maria Lucia Barron Estrada[1] · Hector Manuel Cardenas Lopez[1]

Published online: 23 October 2019

Abstract
An intelligent tutoring system is used as an efficient self-learning tutor, where decisions are based on the affective state of the user. These detected emotions are what experts call basic emotions and the best-known recognition technique is the recognition of facial expressions. A convolutional neural network (CNN) can be used to identify emotions through facial gestures with very high precision. One problem with convolutional networks, however, is the high number of hyperparameters to define, which can range from a hundred to a thousand. This problem is usually solved by an expert experience combined with trial and error optimization. In this work, we propose a methodology using genetic algorithms for the optimization of hyperparameters of a CNN, used to identify the affective state of a person. In addition, we present the optimized network embedded into an intelligent tutoring system running on a mobile phone. The training process of the CNN was carried out on a PC with a GPU and the trained neural network was embedded into a mobile environment. The results show an improvement of 8% (from 74 to 82%) with genetic algorithms compared to a previous work that utilized a trial and error method.

Keywords Genetic algorithm · Convolutional neural networks · Intelligent tutoring systems

1 Introduction

Intelligent tutoring systems (ITS) help students to improve their learning process (McCartin-Lim et al. 2018). These systems have been used in different areas of teaching such as mathematics (Griffith and Griffith 2017), medicine (Suebnukarn and Haddawy 2004), electronics (Graesser et al. 2018), natural languages (Ghali et al. 2018), among other. It has been shown that using emotional classifiers in ITS to determine the emotional state of students helps the teaching process (Linnenbrink-Garcia and Pekrun 2011). ITS use decision making algorithms. One of the most common algorithms for decision making is fuzzy logic (Fahmi 2018; Fahmi et al. 2018; Fahmi and Amin 2019). Human emotion detection is important in the implementation of decision making in ITS. Human emotions were first classified by Ekman in his work Ekman (1992), where he categorized them as basic human emotions. Human emotion recognition is the process of predicting physical expressions such as facial expressions or brain signals and is characterized by features that allow us to differentiate one emotion from the others. For example, in speech, we have features such as the tone of the voice. In body expressions, there are features like the position and movements of the body. Even heartbeats and brain signals are features that express emotions, but their detection require special and invasive devices (Piho and Tjahjadi 2018). The face is one of the most expressive parts of the human being, since this is his primordial channel of communication to express emotions.

The recognition of emotions through the face has been one of the most addressed topics by researchers in affective computing (Zeng et al. 2009; Calvo and D'Mello 2010; Shan et al. 2009). This is usually referred to as recognition of facial expressions. Within a learning environment, the most important emotions to handle are those that have to do with the teaching–learning process (e.g., bored, frustrated, confused, and engaged). In research work, the recognition of emotions has been carried out using different classification

Communicated by V. Loia.

✉ Hector Rodriguez Rangel
hrodriguez@itculiacan.edu.mx

[1] Division de Estudios de Posgrado e Investigacion, Tecnológico Nacional de México Campus Culiacán, Juan de Dios Batiz 310 pte. Col Guadalupe C.P.80220, Culiacàn, Mexico

techniques such as artificial neural networks (ANN), support vector machines (SVM), or Naïve Bayes (Chakraborty and Kopparapu 2016; Bhakre and Bang 2016). However, in recent years, convolutional neural networks (CNN) have proven to be very successful (González-Hernández et al. 2018; Kumar et al. 2017). One problem with CNNs, however, is the tuning of a large number of hyperparameters. This process is commonly carried out through trial and error or based on previous and similar works. Due to the dimensions of the hyperparameters when using this technique, it is difficult to reach a quasi-optimal topology that allows to ensure the improvement in the accuracy of recognition.

The main novelty and contribution of this paper is the implementation of a methodology that improves the recognition rate of learning-centered emotions (facial expressions), by means of a genetic algorithm for optimization of hyperparameters in a CNN. The emotion recognizer was adapted to an intelligent learning environment (ILE) called Multi-Sensei, which uses cognitive and affective variables in its adaptive learning process. The ILE and the emotion recognizer run inside tablets and cell phones.

This paper is structured as follows: Sect. 2 shows the related work of facial expression recognition, convolutional neural networks, and affective tutoring systems. Section 3 presents the work done for creating an emotional corpus, optimizing the CNN, implementing the CNN in Android mobiles, and integrating the CNN into an intelligent tutoring system. Section 4 shows the result and discussion when testing the optimized CNN, and finally Sect. 5 presents conclusions and future work.

2 Related work

2.1 Facial expression recognition

There are different techniques to perform the recognition of facial expressions. A technique that bases its recognition on the appearance or texture of the face applies operators and filters to the image in order to obtain a set of features that are representative of the face. Local binary patterns (LBP) is a method of this type that takes a grayscale image and divides it into different areas. To obtain a frequency histogram, an LBP operator calculates each area. There are several works with important results that have been implemented based on this method (Zhang et al. 2016; Parkkinen et al. 2016; Xu et al. 2015). Techniques based on geometric distances are methods that work using key points of the face. These points can be located using expression templates, unit actions of the facial action coding system, or distance training by means of examples obtained from facial expression corpora. There are also several important works for recognition of facial expressions based on geometric distances, with excellent results. In Pu et al. (2015), the authors present a facial recognizer of image sequences using a twofold random forest (TRF) classifier for Ekman's basic emotions. In this case, the classifier analyzes facial expressions by means of their action units. Also, in Ghayoumi and Bansal (2016), the authors propose a method that unifies three different techniques: facial action units, geometric features, and graph-based modeling. Also, Salmam et al. (2016) measures the distance of six geometric units through the face. The three types of methods for measuring distances were Euclidean, Manhattan, and Minkowski. JAFEE and CohnKanade datasets were used to train the classifier. In recent years, CNNs were used in the recognition of facial expressions. In the work in Yu and Zhang (2015), a method for the recognition of static facial expressions is presented; the authors use three techniques to detect faces using the facial expressions in the wild (SFEW 2.0) dataset. The techniques are joint cascade detection and alignment, Deep CNN, and mixtures of trees. In addition, a five-layer CNN architecture was proposed, but instead of adding pooling layers in each connection between convolutional layers, a stochastic layer is used. The results had an accuracy of 83% for the happy class, 73% for the surprise class, and below 70% for the rest of the classes. In Ding et al. (2016), the FaceNet2ExpNet network, a convoluted neural network designed for the learning of basic emotions from static images of small databases, is proposed. The authors tested their architecture by observing the visualizations generated during the training epoch. They observed that the layers maintained high-level information of the faces and emotions that were being classified. The evaluations with the CK + dataset obtained a prediction from 90 to 100%, with OuluCASIA from 75 to 94%; with PDT from 83 to 94%, and with SFEW from 10 to 100%. In González-Hernández et al. (2018), the authors tested two CNNs to classify basic emotions. The first CNN explored the impact to reduce the number of layers of deep learning while the second CNN divided the input images horizontally into two parts based on the positions of mouth and eyes. Training and testing were performed on the Karolinska directed emotional faces (KDEF) dataset. The first architecture obtained a precision of 96.63%, while the second architecture obtained a precision of 86.73%. In Burkert et al. (2015), a CNN architecture was proposed that consisted of four main processes. The first one preprocessed the data automatically. The next process performed a grouping, which reduced the sample size of the images and performed a normalization using the LRN1 algorithm. The next two processes were two feature extraction procedures in parallel (FeatEx for its acronym in English). These procedures were the core of the architecture. The implementation was made with the Caffe library and tests were performed using the CKP dataset, obtaining an average accuracy of 99.6%.

2.2 Convolutional neural networks

In the literature, we find a large number of works where convolutional networks are used for image classification, showing a great performance when compared to other image classification techniques. The CNNs are made up of different filters of more than one dimension. Defining the topology of the CNN is a complex task, due to the number of hyperparameters to optimize. For this reason, different approaches have emerged for the definition of an optimal topology. In the work of Bergstra and Bengio (2012), they perform a comparison among grid search, random search, and manual search. They used neural networks and deep belief networks. The comparison among these three methods with a 32-dimensional combination found statistically equal performance on four of seven data sets and superior performance on one of seven. In Snoek et al. (2012), the authors perform the optimization of a convolutional neural network under the CIFAR-10 dataset. They argue that a Bayesian treatment with a Gaussian process kernel is preferred to the approach based on optimization of the Gaussian process hyperparameters. The Gaussian process is a convenient and powerful prior distribution on functions. They also present algorithms to perform experiments in parallel. They mention that though this process has its limitations, in their experiments, beat the state of the art by 3%. In other works, reconfigurable computation patterns have been used in CNN, like in Tu et al. (2017), where they perform a comparison of several network architectures. The authors shifted different computational patterns for power usage optimization. In addition, there have been other works that try to solve the problem of hyperparameter optimization using more traditional approaches such as Bayesian optimization, gradient-based methods, mesh optimization, random optimization, and sequential search.

2.3 Optimization of complex tasks

On the other hand, in the literature, you can find a large number of search/optimization techniques. Among them, there are the basic ones such as grid search or random search (Dinh and Van der Baan 2019). You can also find other techniques such as the use of PSO for the scheduling problem (Deng et al. 2019), or for multi-objective problems such as airport gate assignment (Deng et al. 2017a). It is also valid to use more than one technique as in the work of Deng et al. (2017a) where PSO is used with least square combined with fuzzy information entropy for the fault diagnosis of a motor bearing (Deng et al. 2018). Evolutionary computing has also been used with other techniques to solve complex optimization problems like in the work of Deng et al (2017b). The neural evolution or the use of evolutionary computation in the family of neural networks is not new. It has been applied successfully for several decisions tasks (Miikkulainen et al. 2017; Floreano et al. 2008; Montana and Davis 1989) like control, robotics, artificial life, among others. Neural evolution is usually used to define the weights in smaller nets or combined with gradient descent algorithms for larger nets. In the work of Miikkulainen et al. (2017), they define the deep evolution as a multilayer optimization, using the evolutionary computation as a high-level optimization (i.e., layer, size, activation function, etc.) and the gradient descent algorithms for the low-level optimization (i.e., connection weights).

2.4 Affective tutoring systems

In this section, we have considered only those systems oriented to a particular domain that handle affect recognition as part of their teaching methodology. An example of an affective learning environment is found in JavaTutor (Wiggins et al. 2017). This tutoring system is an intelligent learning environment for teaching the Java programming language and its way of detecting affect is through the user's written dialogue. In Arevalillo-Herraez et al. (2017), an ITS for the teaching of algebra is presented. Throughout the learning process, the tutoring system takes care and monitors the cognitive aspects of the student. A sensor free affect detection module supports the cognitive part. After completing a problem, the student is asked about his affective state in terms of valence, activation, and autonomy. Thompson presented Genetics with Jean in Thompson and McGill (2017). This is an affective and intelligent tutor for the teaching of genetic topics. In the affective model, a dimensional view of the emotions is used; its parts are the affective activation (the degree of activation or excitement experienced by the student) and the valence (how positive or negative the experience is considered). In Lin et al. (2016), the authors propose an educational system of distance instruction where the affect or emotion of the student is recognized. The domain of the tutoring system is digital art. Within its design, a multimodal detection system is proposed which uses the recognition of emotions through facial expressions and the conductivity of the skin through galvanic responses. In Wixon et al. (2014), the recognition of emotions is for four affective states: confidence, agitation, frustration, and interest. The tutoring systems used in the study were MathSpring and Cognitive tutor algebra; both ITS for the basic education of mathematics. In Wang and Lin (2018), a prototype of an ITS is proposed for the teaching of physics subjects. Two modules of emotion recognition were implemented: a semantic recognizer and a facial expression recognizer that uses the EmguCV library with Haar features.

3 CNN for emotion recognition

Next, we present the different steps required to implement a CNN for facial expression recognition in learning-centered

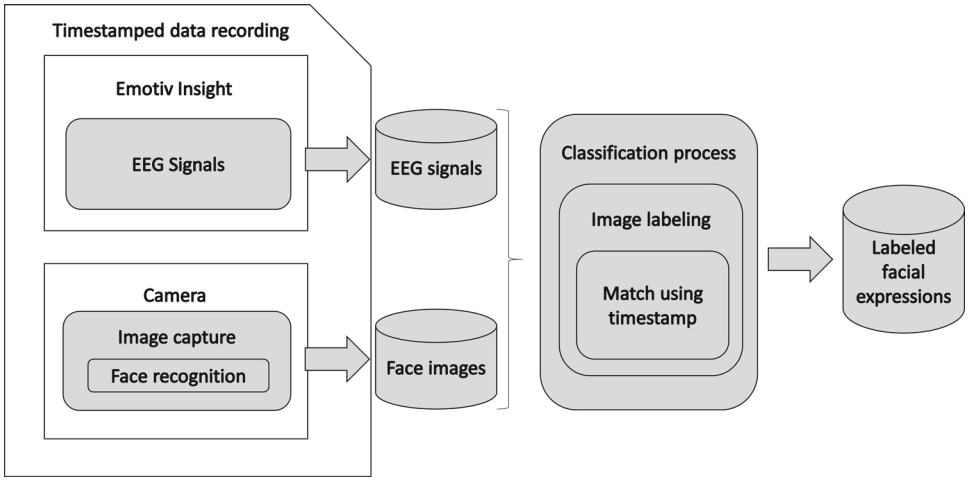

Fig. 1 Method to create the emotional dataset

emotions with application to an ITS. The CNN was optimized using genetic algorithms (GA). The steps that involved the creation of the system were creating a corpus, designing and optimizing the CNN, embedding the CNN to a mobile device (Android), and integrating the CNN to an ITS.

3.1 Creating a corpus

An essential part of any recognition system is the dataset for training. Datasets contain relevant information for any recognition system to be able to discriminate and classify important data. We proposed a method for building a facial expression dataset using the EEG-based brain computer interface (BCI) system Emotiv Insight. This interface system captures the brain activity and provides information about the emotion that the student is presenting. Next, we describe the used device and methods. We searched for a method that could capture expressions within an educational context. In addition, we searched for an activity related to the domain of the ITS that uses the facial recognition system. We set the EEG devices to 38 students, of which, 28 were men and 10 were women. The students were writing code in Java. At the same time, the Emotiv devices obtained their affective state. A labeling of the emotion with respect to each student is carried out automatically by an application. Figure 1 presents the method for creating the emotional dataset. First, the student writes code for a problem. The Emotiv Insight device obtains the student's brain activity (EEG signals) and the webcam takes a photograph of the student's face every 5 s. Second, every student's face is labeled by the system with the actual emotion obtained from the Emotiv device. Third, the student face (photograph) labeled with the emotion is saved into the emotional Dataset. Fourth, we evaluate the matching between emotional labels and the facial expressions saved into the dataset. When we find there is no match between emotion and facial expression, we discard both from the dataset. In the end, the emotional dataset contained 5560 labeled images. The labels used for the classification of the images were boredom, engagement, excitement, focus, relaxation, and interest. This dataset was named database Insight (dbI).

3.2 Designing and optimizing the CNN

A CNN is a class of deep, feed-forward artificial neural network, most commonly applied to image analysis. The image preprocessing in a CNN is less complex compared to other image classification algorithms. This is because the convolutional part of a CNN extracts the main features of an image. The CNNs have a multilayered architecture with three types of layers. The first type of layer, called the convolutional layer, extracts the main features and patterns from an image. The second layer, called the pooling layer, decreases the number of final features from the previous step. Finally, a fully connected layer classifies the features and patterns obtained from the previous layers. Figure 2 shows the three layers of a CNN.

The input is an image, and this image goes through the convolutional layer. Then, the features are reduced in the pooling layer to finally be classified in a fully connected neural network. The hyperparameters of a CNN define its topology. The hyperparameters used in this work are described as follows:

- **Layers:** Number of layers of the CNN.
- **Filter shape:** Filter size used to extract the features and patterns in a convolutional layer
- **Stride:** Number of steps that are taken while moving a filter over an image.
- **Dropout:** Probability of dropping a certain number of neuron connections from the fully connected layers to prevent overfitting.
- **Batch size:** Number of samples used at a given time to train the CNN

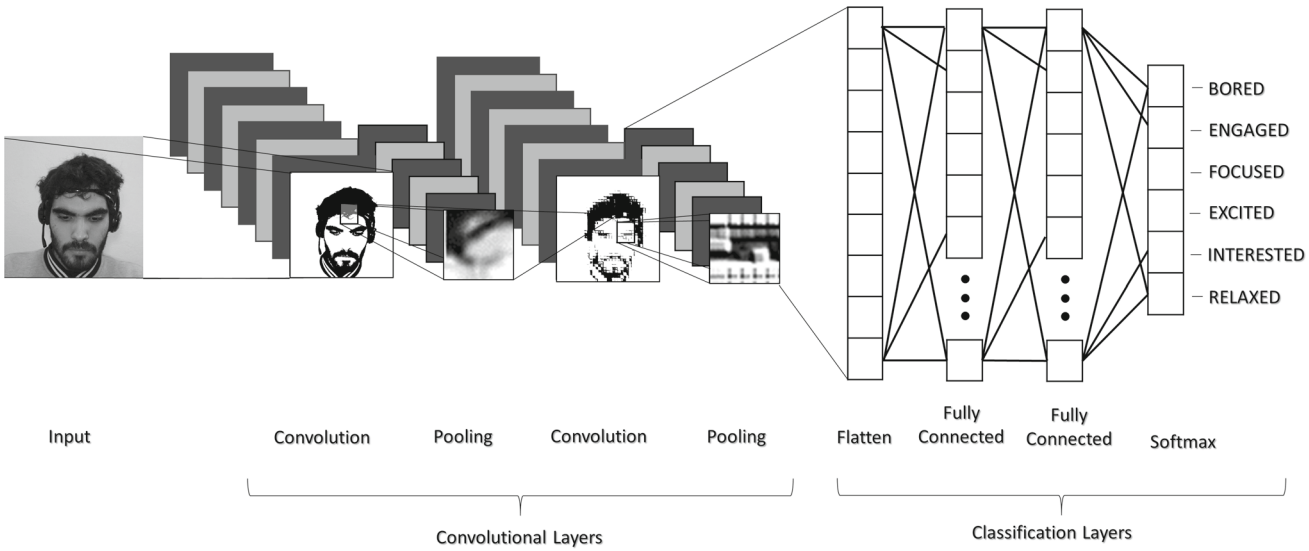

Fig. 2 Common architecture of a convolutional neural network

- **Activation function:** Output of the nodes on a neural network given an input.
- **Optimizer:** Algorithm that helps to minimize the error function with each iteration.
- **Training size:** Number of images from the dataset that are used for the training process.
- **Number of neurons:** Number of neurons in the fully connected layers.
- **Epochs:** Number of iterations for the training of the CNN.

3.2.1 Optimization of the CNN

The optimization on the CNN is made through the change in hyperparameters with the purpose of achieving a higher accuracy in prediction. There have been some optimization methods in CNNs like random search (Salmam et al. 2016), practical Bayesian optimization (Yu and Zhang 2015), among others. For this process of optimization, a GA was used. The purpose of the GA is to evolve a set of individuals called a population to obtain the best solution for a problem. This is achieved by submitting the population to aleatory actions in resemblance to those in biologic evolution. These individuals are rated by a score or fitness. This scoring is calculated by a fitness function which is a mathematical representation of the problem. This fitness also determines the probability of being selected for reproduction depending on the selection method. In GA, an individual is a coded representation of a solution usually called a chromosome which is a set of genes that define the CNN topology. GA follows four main steps which are explained in Algorithm 1. These steps are creating a random population, evaluating the fitness of the population, selecting these individuals to evolve, and evolving the selected individuals. The evolution is performed by crossing over and mutating the selected individuals for the creation of the next generation. This process is repeated for several generations until one of two conditions is met: the fitness of an individual surpasses a threshold or the maximal number of generations is met. The results give a highly fit trained CNN.

Algorithm 1: Basic GA

```
1  begin /*Genetic Algorithm*/
2  Generate population;
3  Evaluate fitness;
4  while not finished do
5      begin /*Generate new Population*/ for Population size do
6          begin /*Reproductive cycle*/
7          Select two individuals from the previous generation for
           crossover (selection probability according to fitness)
8          Crossover both individuals obtaining two offspring
9          Mutate both offspring with certain probability
10         Evaluate Fitness from both offspring
11         Insert both mutated offspring into the new generation
12     end
13     if population has converged then
14         Finished = TRUE
15     end
16 end
```

GA starts with a random population where each individual is coded with floating point values from 0 to 1 used to represent the CNN hyperparameters. One hyperparameter per gene was used to code the CNN topology. The coded (normalization) process was performed using Eq. 1.

Table 1 Type of data by hyperparameter

Type	Hyperparameter	Numeric value
Integer	Number of layers	2, 4, 6, 8, 10
	Number of neurons	1…500
	Batch size	32, 64, 128
	Epoch	1…20
	Stride	2, 3, 4
Class	Activation function	C1, C2, C3, C4
	Optimizers	C1, C2, C3, C4
Percentile	Dropout	0%…90%
	Training size	50%…90%

$$y_i = \frac{X_i - X_{\min}}{X_{\max} - X_{\min}} \tag{1}$$

where X_i represents the data to be normalized, $X_{\min}$ represents the new minimum value, $X_{\max}$ the new maximum value, and y_i the result of the normalization equation. The evaluation of an individual starts by decoding each gene inside the chromosome. This process is performed clearing X_i from Eq. 1. The ceiling function was applied to the result and assigned according to its type. The hyperparameter types are described in Table 1.

CNN hyperparameters were split in topology and training process hyperparameters. For the optimization process, 22 genes were used for topology hyperparameters and 4 genes were used for training process hyperparameters. Figure 3 shows the representation of the chromosome.

The fitness function used to evaluate the individuals in the GA is shown in Eq. 2.

$$\text{fitness} = \frac{\text{accuracy}}{\text{loss}} \tag{2}$$

In this equation, accuracy represents how many correct classifications from the validation set the CNN made. Loss represents the distance between the classification made and the correct classification. Therefore, the higher the accuracy of the CNN the better fitted the individual is for survival and, in the same way, the lower the loss from the classification the higher the fitness of the individual. For the fitness evaluation of all individuals, an algorithm was created for instantiation of the CNN. This is presented in Algorithm 2. Algorithm 2 is divided into three parts: the first part decodes the chromosome. This decoding process defines the layers, the epochs, the training algorithm, among others. The second part defines the topology of the CNN according to the parameters decoded in the first part. The third part compiles and trains the network defined in the previous part. Finally, Algorithm 2 returns the accuracy of the training process.

Algorithm 2: Fitness Evaluation

```
1  begin /*Fitness Evaluation*/
2  Decodify layers;
3  Decodify convolutional layers;
4  Decodify dense layers;
5  Decodify individuals;
6  Decodify batch;
7  Decodify training size;
8  Decodify epochs;
9  Decodify optimizer;
10 begin /*Sequential modeling*/
11 for i in convolutional layers do
12     if i==0 then
13         Add Conv(batch, Decodify activation);
14         Add MaxPool;
15     else
16         if i%2 == 0 then
17             Add MaxPool;
18         else
19             Add Conv(batch, Decodify activation);
20         end
21     end
22 end
23 Add Flatten; for j in dense layers do
24     if j==0 then
25         Add Dense(Decodify neuron number and activation);
26     else
27         if j%2 == 0 then
28             Add Dropout;
29         else
30             Add Dense(Decodify neuron number and activation);
31         end
32     end
33 end
34 Add Dense(Number of Classes,SoftMax);
35 begin /*Training Sesions*/
36 Compile CNN(optimizer);
37 Calculate training sets (training size, dataset);
38 Train(training sets, epoch, batch, validation);
39 return (accuracy/loss);
```

In Algorithm 2, the process for fitness evaluation is demonstrated. The upper section (lines 1–9) of the algorithm involves the decodification of the GA chromosome. In the middle section (sequential modeling—lines 11–34), a FOR-LOOP is used for the construction of the convolutional and dense layers, where logical conditions are used to ensure that the model is properly built. In the final part (lines 35–39) of the algorithm, the output layer is added as a dense layer, using a number of neurons corresponding to the number of output classes with the SoftMax function. Finally, the built network is compiled and trained. A value of fitness is calculated and returned using Eq. 2

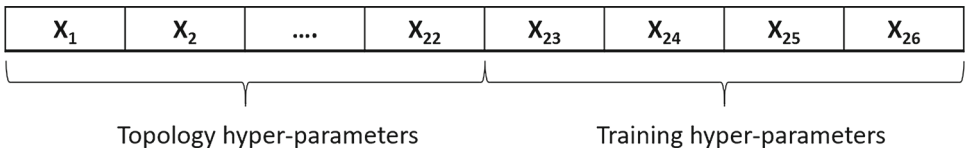

Fig. 3 Representation of the chromosome for hyperparameter selection

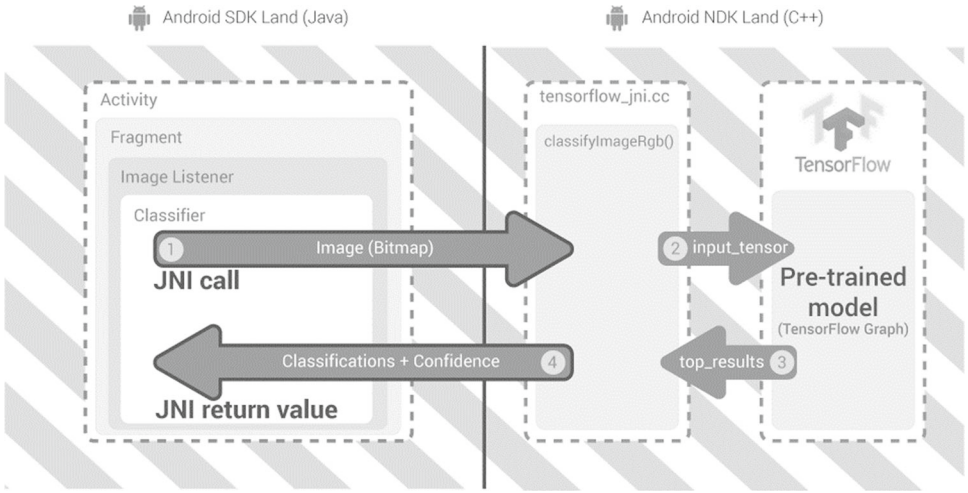

Fig. 4 Process of recognition of emotions through an android exported model

3.3 Adapting the CNN for mobile devices

A trained CNN can be represented as two things: the topology and the weights. A process named serialization can store a CNN. Serialization is the process of converting data structures or objects into a format, which can be saved in long-term storage. For portability, the CNN was serialized by using format protobuf developed by Google with the use of TensorFlow (Abadi et al. 2015). To achieve this, the next steps were followed: first, the CNN was trained using a personal computer with the graphical processor unit (GPU). After the training, a checkpoint was saved using library TensorFlow. This step creates two files containing a graph representing the CNN topology and all its metadata. These files were then serialized and transformed into a single protobuf file. The protobuf file was stored and exported as a component of the ITS in the Android mobile device.

Once the CNN model on protobuf format was stored in the mobile device, TensorFlow was used to deserialize the CNN and create an instance of the previously trained CNN in the ITS. This CNN was instantiated and used to classify the user's facial expressions. Facial expression recognition is related to educational emotions (bored, frustrated, engaged, excited, relaxed, and focused). The process for this was as follows: an image was obtained from the front camera of the mobile device. By using OpenCV (Bradski 2000), the user's face was detected. Then, the user's image was reshaped to 150x150x1 dimensions, as this was the same shape used to train the CNN. The reshaped image was then transformed into a Bitmap. The Bitmap was then converted into a vector and the vector was given through a java native interface (JNI) call to the TensorFlow CNN. The CNN classified the data and sent back an array containing the six emotions. Each emotion is represented with a value that denotes the probability of being recognized. Figure 4 illustrates this process.

3.4 Integration of a CNN into an ITS

We have developed an affective ITS named Multi-Sensei that helps elementary school students to learn and improve their multiplication and division skills. Multi-Sensei adapts the exercises presented to each student depending on the cognitive and affective values, *number of aids*, *time spent in exercise*, *number of errors*, and *actual emotion*. Depending on the fuzzy values of each cognitive and affective variable, a fuzzy logic system determines the complexity of the next exercise. Figure 5 illustrates the evaluation process of Multi-Sensei. For clarity purposes, an example of a fuzzy rule is also shown as part of Fig.5.

4 Tests and discussion

The tests consisted in measuring the accuracy of the emotion recognizer using GA for hyperparameter tuning. The recognizer was compared with four different classification algorithms: a support vector machine (SVM), an artificial neural network (ANN), a K-nearest neighbors (KNN), and a CNN for learning-centered emotion classification presented in a previous work (González-Hernández et al. 2018). The classifiers used three different feature extraction techniques: local binary patterns(LBP), geometric based feature extrac-

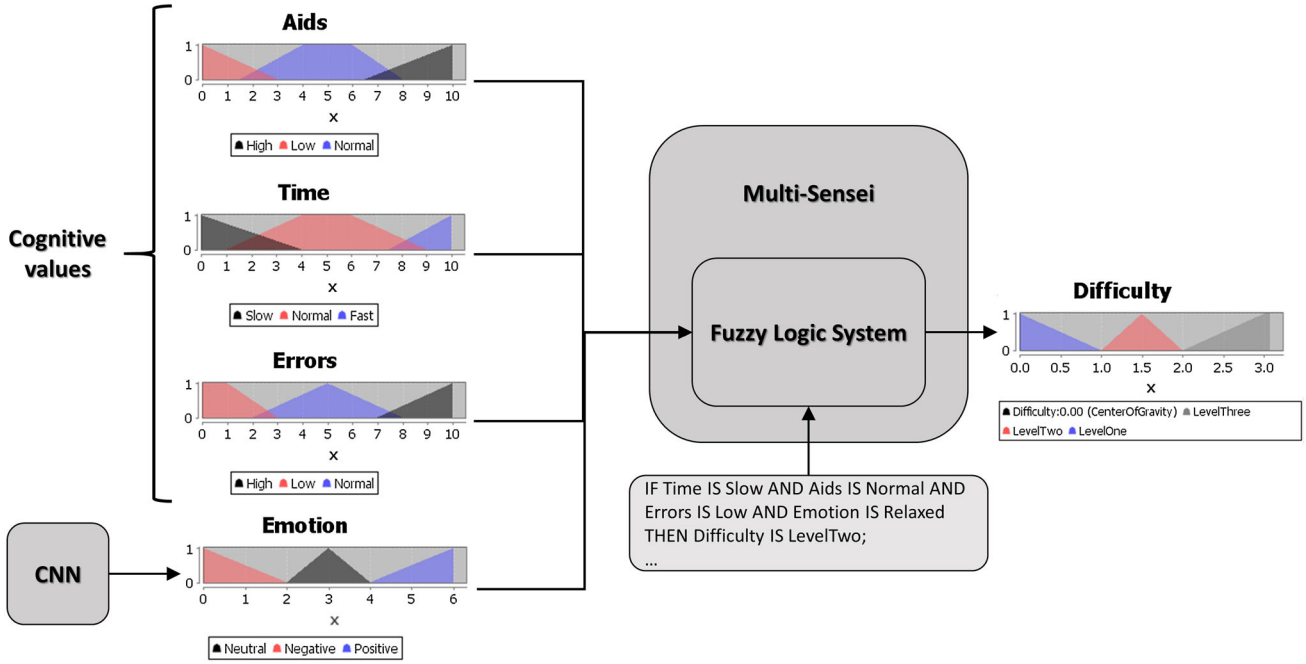

Fig. 5 The affective and cognitive values of the fuzzy logic system in ITS Multi-Sensei

Table 2 Class distribution for dataset build with Emotiv Insight

Emotion	*EmotivEpocInsight*
Bored	1040
Engaged	1955
Excited	1661
Focused	222
Relaxed	28
Interested	150

Table 3 Topology hyperparameters

Layers	Stride	Activation	NN	NN activation	Dropout (%)
2	2	SoftMax	1	SoftMax	0
4	3	Elu	.	Elu	.
6	4	Selu	.	Selu	.
8	.	Relu	.	Relu	.
10	.		500		90

Table 4 Training hyperparameters

Epoch	Batch	Optimizers	Training size (%)
1	32	SGD	50
.	64	RMSProp	.
.	128	Adam	.
.	256	Nadam	.
100			90

tion, and convolutional filters (CF). As mentioned before, this test was made with the use of the dataset dbI, the corpus created with the use of Emotiv Insight. The dataset class distribution (number of images for each emotion) is shown in Table 2.

The dbI has six classes of spontaneous learning-centered emotions that were used to train the CNN. The hyperparameter tuning was made with the use of a GA with a population size of 100 individuals with 24 genomes per individual, a total of 50 generations, a crossover probability of 80%, a mutation rate of 20%, and selection was through tournament. Topology and training hyperparameters with their accepted values for tuning the GA, are shown in Tables 3 and 4.

With the use of the GA, the best-fit individual obtained an 82% accuracy rate compared to 74% using the same dataset in the previous work (González-Hernández et al. 2018). This value represents an 8% improvement only by hyperparameter tuning with GA when compared with CNN classification and is a huge improvement when compared with other methods of classification and different feature extractors. Table 5 shows the results of the comparison of different techniques for learning-centered emotion classification developed in the previous work (González-Hernández et al. 2018). The first four rows correspond to the different methods previously tested, and the fifth row corresponds to the result of this work.

This improvement shows that even with an unbalanced database, we can obtain favorable results. We believe that

Table 5 Method comparison for learning-centered emotion classification

Classifier/feature extractor	LBP (%)	Geometric based (%)	CF (%)
KNN	70	61	65
ANN	74	51	61
SVM	69	61	63
CNN	–	–	74
CNN+AG	–	–	82

the result should improve once the dbI database undergoes a filtering process and a class distribution balancing.

For the tests and experiments, we used Python as the programming language, Keras for model creation with TensorFlow 1.10 as back end on a server using an i7-6700 HQ CPU with 8 cores at 3.5 GHz of core speed, a Nvidia GTX 1060 graphics card with 6 GB of dedicated video memory and 1280 cuda cores, and 16 GB system dedicated RAM.

5 Conclusions

This work presents the use of a GA for hyperparameter tuning in a CNN for the recognition of learning-centered emotions. The results suggest that the use of a GA for automatic hyperparameter definition of CNN helps to improve the accuracy when compared with CNN made with trial and error and other different machine learning classifiers. We still need to perform more tests to compare our CNN with other classifiers also using deep learning algorithms (e.g., long short-term memory networks). Due to the current importance of mobile learning, we implemented our CNN within a mobile device for Android. This will allow us to support different intelligent learning environments for mobile phones, such as Multi-Sensei or others. The CNN can effectively support an intelligent tutoring system to personalize its decisions based not only on cognitive aspects but also on affective aspects of the student. We are currently testing the ITS Multi-Sensei using the emotion recognizer with an optimized CNN. Image processing of the user face was performed using OpenCV library (Bradski 2000). The CNN was implemented with TensorFlow library (Abadi et al. 2015). The GA was written in Python. Multi-Sensei was coded in Java. As for work in the future, we will balance our dbI dataset by increasing the classes *focused*, *relaxed*, and *interested*. This should produce better recognition rates in our CNN. We also want to test our optimized CNN with sentiment analysis and opinion mining. On the other hand, we will test the ITS Multi-Sensei with students from local schools. In this case, we will check the effect of recognizing and handling emotions while the student learns the multiplication and division process.

Compliance with ethical standards

Conflict of interest The authors declare that there is no conflict of interest regarding the publication of this paper.

Human and animal rights This article does not contain any studies with human participants or animals performed by any of the authors.

References

Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, Vanhoucke V, Vasudevan V, Viégas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X (2015) TensorFlow: large-scale machine learning on heterogeneous systems. Software available from tensorflow.org

Arevalillo-Herraez M, Arnau D, Ferri FJ, Santos OC (2017) Gui-driven intelligent tutoring system with affective support to help learning the algebraic method. In: 2017 IEEE international conference on systems, man, and cybernetics (SMC), IEEE, pp 2867–2872

Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. J Mach Learn Res 13:281–305

Bhakre SK, Bang A (2016) Emotion recognition on the basis of audio signal using Naive Bayes classifier. In: 2016 international conference on advances in computing, communications and informatics (ICACCI), IEEE, pp 2363–2367

Bradski G (2000) The OpenCV Library. Dr. Dobb's J Softw Tools 120:122–125

Burkert P, Trier F, Afzal MZ, Dengel A, Liwicki M (2015) DeXpression: deep convolutional neural network for expression recognition

Calvo RA, D'Mello S (2010) Affect detection: an interdisciplinary review of models, methods, and their applications. IEEE Trans Affect Comput 1(1):18–37

Chakraborty R, Kopparapu SK (2016) Validating "Is ECC-ANN combination equivalent to DNN?" for speech emotion recognition. In: 2016 IEEE international conference on systems, man, and cybernetics (SMC), IEEE, pp 004311–004316

Deng W, Zhao H, Yang X, Xiong J, Sun M, Li B (2017a) Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment. Appl Soft Comput 59:288–302

Deng W, Zhao H, Zou L, Li G, Yang X, Wu D (2017b) A novel collaborative optimization algorithm in solving complex optimization problems. Soft Comput 21(15):4387–4398

Deng W, Zhang S, Zhao H, Yang X (2018) A novel fault diagnosis method based on integrating empirical wavelet transform and fuzzy entropy for motor bearing. IEEE Access 6:35042–35056

Deng W, Xu J, Zhao H (2019) An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem. IEEE Access 7:20281–20292

Ding H, Zhou SK, Chellappa R (2016) FaceNet2ExpNet: regularizing a deep face recognition net for expression recognition

Dinh HN, Van der Baan M (2019) A grid-search approach for 4D pressure-saturation discrimination. Geophysics 84(4):IM47–IM62

Ekman P (1992) An argument for basic emotions. Cognit Emot 6(3–4):169–200

Fahmi A (2018) Expected values of aggregation operators on cubic triangular fuzzy number and its application to multi-criteria decision making problems. Eng Math 2(1):1

Fahmi A, Abdullah S, Amin F, Ali A (2018) Weighted average rating (War) method for solving group decision making problem using triangular cubic fuzzy hybrid aggregation (Tcfha) operator. Technical report 1, Punjab University

Fahmi A, Amin F (2019) Precursor selection for sol–gel synthesis of titanium carbide nanopowders by a new hesitant CUBIC fuzzy multi-attribute group decision-making model. New Math Nat Comput 15(01):145–167

Floreano D, Dürr P, Mattiussi C (2008) Neuroevolution: from architectures to learning. Evol Intell 1(1):47–62

Ghali MA, Ayyad AA, Abu-Naser SS, Laban MA (2018) An intelligent tutoring system for teaching english grammar. Int J Acad Eng Res 2(2):1–6

Ghayoumi M, Bansal AK (2016) Unifying geometric features and facial action units for improved performance of facial expression analysis

González-Hernández F, Zatarain-Cabada R, Barrón-Estrada ML, Rodríguez-Rangel H (2018) Recognition of learning-centered emotions using a convolutional neural network. J Intell Fuzzy Syst 34(5):3325–3336

Graesser AC, Hu X, Nye BD, VanLehn K, Kumar R, Heffernan C, Heffernan N, Woolf B, Olney AM, Rus V, Andrasik F, Pavlik P, Cai Z, Wetzel J, Morgan B, Hampton AJ, Lippert AM, Wang L, Cheng Q, Vinson JE, Kelly CN, McGlown C, Majmudar CA, Morshed B, Baer W (2018) ElectronixTutor: an intelligent tutoring system with multiple learning resources for electronics. Int J STEM Educ 5(1):15

Griffith H, Griffith A (2017) Integration of an Intelligent Tutoring Software within an accelerated Engineering mathematics course. In: 2017 IEEE integrated STEM education conference (ISEC), IEEE, pp 131–134

Kumar GAR, Kumar RK, Sanyal G (2017) Facial emotion analysis using deep convolution neural network. In: 2017 international conference on signal processing and communication (ICSPC), IEEE, pp 369–374

Lin Hao-Chiang Koong, Su SH, Chao CJ, Hsieh CY, Tsai SC (2016) Construction of multi-mode affective learning system: taking affective design as an example. Educ Technol Soc 19(2):132–147

Linnenbrink-Garcia L, Pekrun R (2011) Students' emotions and academic engagement: introduction to the special issue. Contemp Educ Psychol 36(1):1–3

McCartin-Lim M, Woolf B, McGregor A (2018) Connect the dots to prove it. In: Proceedings of the 49th ACM technical symposium on computer science education- SIGCSE '18. ACM Press, New York, USA, pp 533–538

Miikkulainen R, Liang J, Meyerson E, Rawal A, Fink D, Francon O, Raju B, Shahrzad H, Navruzyan A, Duffy N, Hodjat B (2017) Evolving deep neural networks. Technical report, Thee University of Texas at Austin

Montana DJ, Davis L (1989) Training feedforward neural networks using genetic algorithms

Parkkinen J, Jaward MH, Parthiban R, Kamarol SKA (2016) Spatiotemporal feature extraction for facial expression recognition. IET Image Process 10(7):534–541

Piho L, Tjahjadi T (2018) A mutual information based adaptive windowing of informative EEG for emotion recognition. IEEE Transactions on Affective Computing pp 1–1

Pu X, Fan K, Chen X, Ji L, Zhou Z (2015) Facial expression recognition from image sequences using twofold random forest classifier. Neurocomputing 168:1173–1180

Salmam FZ, Madani A, Kissi M (2016) Facial expression recognition using decision trees. In: 2016 13th international conference on computer graphics, imaging and visualization (CGiV), IEEE, pp 125–130

Shan C, Gong S, McOwan PW (2009) Facial expression recognition based on local binary patterns: a comprehensive study. Image Vis Comput 27(6):803–816

Snoek J, Larochelle H, Adams RP (2012) Practical bayesian optimization of machine learning algorithms. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ (eds) *A*dvances in neural information processing systems. Curran Associates, Inc., pp 2951–2959. http://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf

Suebnukarn S, Haddawy P (2004) A collaborative intelligent tutoring system for medical problem-based learning. In: Proceedings of the 9th international conference on intelligent user interface-IUI '04. ACM Press, New York, USA, p 14

Thompson N, McGill TJ (2017) Genetics with jean: the design, development and evaluation of an affective tutoring system. Educ Technol Res Dev 65(2):279–299

Tu F, Yin S, Ouyang P, Tang S, Liu L, Wei S (2017) Deep convolutional neural network architecture with reconfigurable computation patterns. IEEE Trans Very Large Scale Integr (VLSI) Syst 25(8):2220–2233

Wang C-H, Lin H-CK (2018) Constructing an affective tutoring system for designing course learning and evaluation. J Educ Comput Res 55(8):1111–1128

Wiggins JB, Grafsgaard JF, Boyer KE, Wiebe EN, Lester JC (2017) Do you think you can? the influence of student self-efficacy on the effectiveness of tutorial dialogue for computer science. Int J Artif Intell Educ 27(1):130–153

Wixon M, Arroyo I, Muldner K, Burleson W, Lozano C, Rai D, Woolf B (2014) The opportunities and limitations of scaling up sensor-free affect detection. In: Proceedings of the 7th international conference on educational data mining. EDM, London, UK, pp 145–152

Xu X, Quan C, Ren F (2015) Facial expression recognition based on Gabor Wavelet transform and histogram of oriented gradients. In: 2015 IEEE international conference on mechatronics and automation (ICMA), IEEE, pp 2117–2122

Yu Z, Zhang C (2015) Image based static facial expression recognition with multiple deep network learning. In: Proceedings of the 2015 ACM on international conference on multimodal interaction-ICMI '15, ACM Press, New York, USA, pp 435–442

Zhang L, Tjondronegoro D, Chandran V, Eggink J (2016) Towards robust automatic affective classification of images using facial expressions for practical applications. Multimed Tools Appl 75(8):4669–4695

Zeng Zhihong, Pantic M, Roisman G, Huang T (2009) A survey of affect recognition methods: audio, visual, and spontaneous expressions. IEEE Trans Pattern Anal Mach Intell 31(1):39–58