Bachelor of Science in Computer Science
June 2022

Facial Emotion Recognition using Convolutional Neural Network with Multiclass Classification and Bayesian Optimization for Hyper Parameter Tuning

Lokesh Bejjagam
Reshmi Chakradhara

Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science. The thesis is equivalent to 10 weeks of full time studies.

The authors declare that they are the sole authors of this thesis and that they have not used any sources other than those listed in the bibliography and identified as references. They further declare that they have not submitted this thesis at any other institution to obtain a degree.

Contact Information:
Author(s):
Lokesh Bejjagam
E-mail: loba21@student.bth.se

Reshmi Chakradhara
E-mail: rech21@student.bth.se

University advisor:
Dr.Shahrooz Abghari, Universitetsadjunkt
Department of Computer Science

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Background:
In everyday life, communication and interaction is very important aspect of every living being. Emotions are one of the important factor that enhances the process of human communication. The facial expressions and gestures convey nonverbal communication cues that play in vital role in interpersonal relations. Hence, understanding the emotions of a human being is very trivial in building relationships. Facial Emotion Recognition plays a vital role in recognising emotions because of its ability to mimic human coding skills. It can aid understanding which emotions a user is experiencing in real-time and it has also gained an immense importance in the present generation with its applications in various fields like medicine, computer vision and image recognition. An emotion recognition system can be built by utilizing the benefits of deep learning techniques. To develop an emotion recognition model with better accuracy, various factors like the type of algorithm used to build it, the size of the training set, parameter tuning algorithms used etc are important. For a good accuracy model, hyper parameter tuning is an essential aspect of the deep learning process. A good choice of hyper parameter optimization can really make a model succeed in meeting desired accuracy. Studying, experimenting and training the model with hyper parameter tuning algorithms like Bayesian Optimization will help in developing a good facial emotion recognition model as well as shows a remarkable scientific progress in the field of deep learning.

Objectives:
The thesis aims to develop a deep learning model for facial emotion recognition using Convolutional Neural Network algorithm and Multiclass Classification along with Hyper-parameter tuning using Bayesian Optimization to improve the performance of the model. The developed model recognizes seven basic emotions in images of human beings such as fear, happy, surprise, sad, neutral, disgust and angry using FER-2013 dataset.

Methods:
An experiment will be performed on the dataset. The data will be evaluated using statistical analysis with the help of correlation. The labeled data will be preprocessed. The dataset will be split into 7 pairs of train and test sets of varying sizes. The deep learning model will be trained with each training set to find the proper training size of the dataset which gives highest training accuracy. The deep learning model will be validated with the test set to find the best test accuracy. The obtained results of the experiment will be analyzed.

Based on our findings regarding the proper training size of the dataset, an experiment will be conducted to check the performance of deep learning model after hyper-parameter tuning using the Bayesian Optimization.The proper training size of the dataset found from RQ1 will be used to train the deep learning model. The labeled data will be pre-processed. We will develop the deep learning model using Convolutional Neural Network algorithm along with Multiclass classification.

Results:
The optimal data ratio for the training and testing was found to be 80% and 20%, respectively for the deep learning model to provide the best accuracy for emotion recognition. It has given an accuracy of 75.47%. Hyperparameter tuning using Bayesian optimization has improved the model's accuracy by 3.37% with the best accuracy of the model resulting in 78.84%, with the FER2013 dataset. The parameters used for hyperparameter tuning are batch_size, epochs and optimizer with their values set to 32, 100 and 'Nadam' respectively.

Conclusions:
Performance of the model can be improved with an optimal dataset training size along with a hyperparameter tuning algorithm. The positive impact of the these two factors has helped in developing an efficient facial emotion recognition model using deep learning with an overall accuracy of 78.84%.

Keywords: Computing Methodologies, Machine Learning, Machine Learning Approaches, Convolutional Neural Network, Facial Emotion Recognition.

Acknowledgments

We would like to give our grateful appreciation to our supervisor Shahrooz Abghari for his marvelous supervision, guidance and encouragement. Sincere gratitude is extended to his generous participation in guiding, constructive feedback, kind support, and advice during our thesis work.
We are extremely thankful to our examiner Prashant Goswami for his noble guidance, feedback and support for our project plan. His valuable lectures regarding research have helped us in understanding the bachelor thesis much in much better way.
We are very thankful to VenuGopal Puripanda for his mentorship, constant guidance and support throughout this project.

Authors:
Lokesh Bejjagam
Reshmi Chakradhara

Contents

List of Figures

List of Tables

List of Acronyms

ML	Machine Learning
DL	Deep Learning
RQ	Research Question
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
GA	Genetic Algorithms
FC	Fully Connected
CNN	Convolutional Neural Network
RBM	Restricted Boltzmann Machines
ANN	Artificial Neural Network
NLP	Natural Language Processing
SVM	Support Vector Machine
ELU	Exponential Linear Unit
FER	Facial Emotion Recognition
LSTM	Long Short -Term Memory
ReLU	Rectified Linear Unit

Chapter 1

Introduction

Emotions are the mental reactions shown by the body to a specific thing or action through behavioral and physiological changes in the human body. Emotion recognition enables the identification of basic human emotions such as anger, fear, disgust, sadness, happiness, and surprise based on some input data. The movement of facial features [19] creates facial expressions. Every person's expressions are different. According to Mehrabian [29], spoken words transmit 7% of the communication, voice intonation conveys 38% of the message, and facial emotions convey 55% of the message. Emotion recognition has its applications in healthcare, entertainment, e-learning, marketing, human monitoring, and security.

Machine learning (ML) is the study of algorithms that can improve performance, data prediction, data learning and accuracy automatically through experience and by the use of data. ML is very useful in detecting and recognizing features accurately with the help of algorithms. ML has wide range of applications in medicine, computer vision, speech recognition and image recognition. One of the best applications of ML is human emotion recognition. ML algorithms are helpful to develop emotion recognition models which can be trained to recognise human emotions accurately.

Deep learning (DL) is one of the branches of ML which completely makes use of artificial neural networks which helps to implement particular technical thoughts of the human brain into a machine. DL allows computational models that are composed of multiple processing layers to learn and train machines to understand huge data and their representations with multiple levels of abstraction. In more detail, DL technique learns categories sequentially through its hidden layer architecture. Each neuron or node in the network reflects a different component of the whole, and together they form a complete visual representation. Each node or hidden layer is assigned a weight that reflects the strength of its association with the output, and the weights are changed as the model evolves [20]. DL algorithms make use of the raw data to extract features, group objects and meaningful data patterns. Algorithms like Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Restricted Boltzmann Machines (RBM) are used in DL to solve real-time problems efficiently. DL uses data in various forms including image, video, text, conversations and speeches. DL does not require manual feature extraction, which is advantageous when using an algorithm to extract important features. CNN algorithm is used to analyze images for emotion detection and is in charge of feature extraction and classification of images based on facial features to assist in detecting human emotions.

In this thesis work, we will develop a DL model using CNN algorithm along with the Multiclass classifier to perform emotion recognition based on the facial features.

Hyper-parameter tuning using Bayesian Optimization and a proper training size dataset are used to improve the accuracy of the model. The dataset used will be FER2013, which is taken from Kaggle and consists of approximately 32,298 gray scale images of human facial emotions [4]. The labeled data is classified into 7 different classes of emotions including anger, disgust, fear, happiness, sadness, surprise and neutral using the Multiclass classifier. The classified dataset will be split into training set and testing set and emotion recognition is performed the CNN algorithm. The obtained results will be evaluated to find out the accuracy of the model with the FER2013 dataset. Training set is the set of samples or a subset of a dataset used to train a model. Test set is the set of samples or a subset of a dataset used to test the trained model to evaluate the model's performance.

1.1 Aim and Objectives

1.1.1 Aim

The aim of the thesis is to develop a deep learning model for facial emotion recognition using Convolutional Neural Network algorithm and Multiclass Classification along with Hyper-parameter tuning using Bayesian Optimization to improve the performance of the deep learning model.

1.1.2 Objectives

The main objectives of the project are:

1. This study deals with an experimental research to develop a deep learning model to identify the emotion of a human face based on facial features using TensorFlow.

2. To develop a deep learning model using convolutional neural network along with multiclass classification and FER2013 dataset.

3. The data will be preprocessed. The DL model will be trained with the train set and validated with the test set.

4. The experiment also includes Hyper-parameter tuning using Bayesian Optimization along with a suitable training size dataset to improve the performance.

1.2 Research Questions

Considering that the DL model will be developed using CNN algorithm along with Multiclass Classification and the FER2013 dataset, it is taken from Kaggle and consists of approximately 32,298 gray scale images of human facial emotions. It is classified into seven categories anger, disgust, fear, happiness, sadness, surprise and neutral that will be used in this thesis study. The research questions (RQ) listed below have been defined:

RQ1: What is the optimal dataset training size for the DL model to provide the best accuracy for emotion recognition?
Motivation: The research question is chosen to determine the optimal training size of the dataset for the DL model in order to increase the model's accuracy. The train and test sets are comprised of different sizes of the dataset in order to conduct the experiment. After that, the train and test sets are used to train the DL model. The obtained results will be analysed to determine which DL model gives the highest accuracy.

RQ2: Considering the initial parameter setups of the proposed model, to what extent can parameter tuning improve the model's accuracy for emotion recognition?
Motivation: The research question is selected to examine and determine the effect of applying hyper-parameter tuning using Bayesian Optimization in order to boost model accuracy while decreasing model run time. An experiment is carried out to validate the DL model's performance using CNN and Multiclass classification, as well as Bayesian optimization for hyper-parameter tuning. The obtained results will be analyzed, and the outcome will be explained.

Chapter 2

Background

2.1 Machine Learning

ML [17] is the study of algorithms that can improve performance, data prediction, data learning and accuracy automatically through experience and the use of data. It is considered to be a component of artificial intelligence. ML algorithms create a model based on training data to make predictions or judgments without having to be explicitly programmed to do so.
The main use of ML is to take in input data and apply statistical analysis to anticipate the outcome based on the type of data provided. These ML algorithms have a wide range of applications, including scientific predictions, medicine, computer vision, speech recognition and image recognition. DL being the subset of ML, has the capability to work with numerous levels of algorithms. Each one offers a unique approach to interpreting data in a neural network. It is worth delving into neural network, deep learning and algorithms.

2.2 Neural Network

A neural network [7] is a collection of interconnected layers that mimics the functioning of neural networks in the human brain in certain aspects. The term "neural network" refers to a system of neurons that can be either organic or artificial in nature. Because neural networks can adapt to changing input, they can produce the best possible result without requiring the output criteria to be redesigned. Artificial intelligence-based DL is represented by neural networks [23].

2.3 Deep Learning

DL [1] is a deep neural network which is defined as a network that is made up of many layers, each of which has many individual nodes. A neural network aids us in doing our task with less precision, however deep learning aids us in completing our task with efficacy due to the presence of numerous layers. Because a neural network is less complicated, it takes less time to train it, but a deep learning network may take a long time to train. DL allows a system to mimic the functions of a human brain and make judgements in the same way that our brain does. A DL system learns by seeing and making conclusions from various types and patterns of data. There are many types of deep neural networks available.

2.3.1 Convolutional Neural Network

The CNN is one of the most often used deep neural networks. Convolution is a mathematical linear action between matrices that gives it its name. It is a type of Artificial Neural Network (ANN) that is frequently used to assess visual imagery. They're used in image and video recognition, recommender systems, classification, segmentation, medical image analysis, natural language processing Brain–computer interfaces. Particularly impressive were the applications that deal with picture data, such as the largest image classification data set (Image Net), computer vision, and natural language processing (NLP), and the results obtained [12].
The architecture of CNN is magnificient. The 3 important layers of CNN as shown in Figure 2.1 below are the input layer, hidden layers and an output layer respectively known as the convolutional layer, pooling layer, and fully-connected layer [11]. The figure shows how the input given to the model is processed by the CNN to produce evaluated outputs after the model is trained.

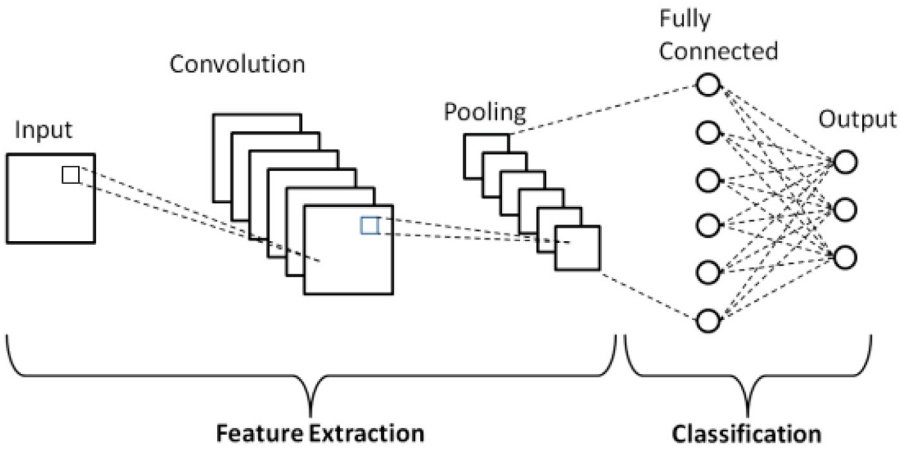

Figure 2.1: Schematic diagram illustrating the architecture of basic Convolutional Neural Network [28].

2.3.2 Layers of Convolutional Neural Network

The three important layers of CNN are as follows (referencing to the Figure 2.1 above):

2.3.2.1 Convolutional Layer

The convolutional layer is the initial layer that extracts the different characteristics from the input photos. The convolution mathematical operation is done between the input picture and a filter of a certain size MxM in this layer. The dot product between the filter and the sections of the input picture with regard to the size of the filter is taken by sliding the filter across the input image (MxM).

The Feature map is the result, and it contains information about the image such as its corners and edges. This feature map is then supplied to further layers, which learn a variety of different features from the input picture.

2.3.2.2 Pooling Layer

The Pooling Layer follows a Convolutional Layer. This layer's major goal is to lower the size of the convolved feature map in order to reduce computational expenses. This is accomplished by reducing the connections between layers and operating independently on each feature map. There are numerous sorts of Pooling procedures, depending on the mechanism utilized.

The largest element is obtained from the feature map in Max Pooling. The average of the components in a predetermined sizediImage segment is calculated using Average Pooling. The smallest element obtained from the feature map in Min Pooling. Sum Pooling calculates the total sum of the components in the designated section. The Pooling Layer is typically used to connect the Convolutional Layer and the Fully Connected (FC)Layer.

2.3.2.3 Fully Connected Layer

The weights and biases, as well as the neurons, make up the FC layer, which is used to link the neurons between two layers. The last several layers of a CNN Architecture are generally positioned before the output layer. Because there are so many hidden layers with varied weights for each neuron's output, it's difficult to follow the data after this stage. This is where all of the data reasoning and computing takes place.

The preceding layers input images are flattened and supplied to the FC layer in this step. After that, the flattened vector is sent via a few additional FC levels, where the mathematical functional operations are normally performed. The classification process gets started at this point.

2.4 Hyper-Parameter Tuning

2.4.1 Hyper-Parameters

A hyperparameter in ML whose value is used to influence the learning process. The values of additional parameters, such as node weights, are determined by training. There are two types of hyperparameters. The first type are Model hyperparameters which cannot be inferred while fitting the machine to the training set as they refer to the model selection task. The topology and size of a neural network are examples of model hyperparameters. The second type are Algorithm hyperparameters which have no effect on the model's performance but affect the speed and quality of the learning process. Learning rate, batch size, and mini-batch size are examples of algorithm hyperparameters. A whole data sample is referred to as batch size, whilst a smaller sample set is referred to as mini-batch size [18].

Different hyperparameters are required by different model training techniques, although some basic algorithms (such as ordinary least squares regression) do not. The training algorithm learns the parameters from the data given through these hyperparameters.

2.4.2 Categories of Hyper-Parameters

Different hyperparameters can be used to improve the model's performance for different types of datasets and models.

In general, hyperparameters may be divided into two groups:

2.4.2.1 Optimization Hyperparameters

These hyperparameters are used to optimize the model, as the name implies [14].

1. Learning Rate: This hyperparameter defines how much freshly collected data will overrule previously accessible data. If the value of this hyperparameter is large, the model will not be successfully optimized since there is a potential it will hop over the minima. On the other side, if the learning rate is set too low, convergence will take a long time.

2. Batch Size: The training set is separated into multiple batches to speed up the learning process. If the batch size is bigger than, the learning time will be longer and more memory will be required to execute the matrix multiplication. There will be more noise in the error computation if the batch size is less.

3. Number of Epochs: In DL, an epoch represents a whole cycle of data to be learnt. In the iterative learning process, epochs are extremely significant. The number of epochs can be increased as long as the validation error is reduced. If the validation error does not improve after a certain number of epochs, it is a hint that the number of epochs should be reduced. It's sometimes referred to as "early stopping."

2.4.2.2 Specific Model Hyperparameters

The structure of the model also includes several hyperparameters [14].

1. Number of Hidden Units: In DL models, this hyperparameter is used to define the model's learning capability. We must define a lot of hidden units for complicated functions, but we must be careful not to overfit the model.

2. Number of Layers: In CNN models, the model's performance improves as the number of layers increases.

2.4.3 Hyper-Parameter Tuning Algorithms

2.4.3.1 Bayesian Optimization

We selected Bayesian optimization for hyperparameter tuning for our thesis work since the grid search parameter tuning technique is extremely slow, and the random search parameter tuning algorithm is somewhat faster but does not provide the optimal outcomes after hyperparameter tuning [22]. Bayesian optimization is a sequential design method that assumes no functional forms for global optimization of black-box functions. It is typically used to optimize functions that are difficult to

assess. Bayesian optimization is a useful strategy for finding the extrema of functions that are computationally difficult to solve. It may be used to solve functions that don't have closed-form expressions. It may also be used for functions that are difficult to compute, have difficult derivatives to analyze, or are non-convex.

The prior distribution of the function f(x) is combined with sample information (evidence) to generate the posterior of the function, which is then used to identify where the function f(x) is maximized according to a criterion. A utility function u, also known as an acquisition function, represents the criterion. In order to maximize anticipated utility, the function u is utilized to calculate the next sample point. When searching for a sampling region, both exploration (sampling from areas of high uncertainty) and exploitation must be considered (sampling from that with high values). This will aid in the reduction of sampling. Furthermore, performance will improve even if the function has multiple local maxima.

The most important process used for Bayesian Optimization is as follows:

- Gaussian Process : It is an ML approach that was created using the Gaussian stochastic process and Bayesian learning theory. A stochastic process influences the attributes of functions, whereas a probability distribution specifies random variables that are scalars or vectors (for multivariate distributions). It has a multivariate Gaussian distribution for every finite sub-collection of random variables. A statistical model of the function is assumed by the Gaussian process, which asserts that identical input creates similar output. Gaussian process is characterized by its mean function m x $\rightarrow$ R and its covariance function k: x $\times$ x $\rightarrow$ R, much like Gaussian distribution is defined by mean and covariance.

2.5 Multiclass Classification

In ML, Multiclass or multinomial classification is the process of categorizing occurrences or samples into one of three or more classes based on the data (classifying instances into one of two classes is called binary classification) [3]. Due to the expanding amount of data available today, data categorization is one of the most challenging issues. Because of the unpredictability and complexity of real-world data, the topic of data categorization is gaining in relevance. A classification method necessitates a high level of precision and consistency, which is difficult for human programmers to achieve. As a result, there is a pressing need to create automated computer-based categorization systems capable of classifying the essential items. We have used multiclass classification in our thesis work to distinguish the exact features from the 7 emotion classes namely anger, disgust, fear, happiness, sadness, surprise and neutral.

2.6 Evaluation Metrics

Confusion Matrix

Confusion matrix [25], also known as an error matrix, is a table structure that allows the performance of an algorithm to be shown. Each row of the matrix contains real class occurrences, whereas each column represents expected class instances, or vice versa. It is used to calculate accuracy, precision, recall and F1 scores. The terms used in the confusion matrix are as follows:

1. True Positive (TP): It is the case when a data point's actual and predicted classes are both 1.

2. True Negative (TN): It is the case when a data point's actual and predicted classes are both 0.

3. False Positive (FP): It is the case when the actual class of a data point is 0 and the predicted class is 1.

4. False Negative (FN): It is the case when the actual class of a data point is 1 and the predicted class is 0.

Accuracy

The ratio of the true predicted values to total predicted values is known as accuracy [25]. The higher the accuracy, the better is the performance of the model.

Precision

The percentage of relevant instances among all retrieved instances is defined as precision [25].

Recall

The proportion of recovered occurrences among all relevant examples is known as recall, or sensitivity [25].

F1-score

The F1-score is the harmonic mean of precision and recall [25].

Loss

The loss occured in the training process, also called as training loss, it indicates the error on the training set of data. It is the total number of errors occured for training set while training the model.

test_loss

The loss occured in the testing process, indicates the error on the test set of data. It is the total number of errors occured for test set while testing the model after training the model with the training set.

Chapter 3

Related Work

Bellamkonda [13] have carried out research on Facial Emotion Recognition (FER) by hyper-parameter tuning of CNN using GA. The dataset used for the research is FER2013 [4], to recognize seven emotions fear, happy, surprise, sad, neutral, disgust and angry. The experimental study deals with impact of GA methods in hyper-parameter tuning.

Chalavadi et al. [9] have done their research in human emotion recognition using CNN and GA. A gradient descent algorithm is used to train the CNN classifiers (to find a local minimum) during fitness evaluations of GA chromosomes. The global search capabilities of GA and the local search ability of gradient descent algorithm are exploited to find a solution that is closer to global-optimum. They have showed that the combining evidences of classifiers generated using genetic algorithms helps to improve the performance. Their research was carried on UCF50 dataset [8].

Boughida et al. [15] have done their research in developing a human emotion recognition model based on Gabor filters and GA. Gabor features are extracted from regions of interest of the human face detected using facial landmarks. In addition, a genetic algorithm was designed to optimize Support Vector Machine (SVM) hyper-parameters and select the best features simultaneously. The research was carried on the JAFFE [6] and CK+ [5] datasets respectively.

Grigorasi and Grigore [30] have done their research on optimizing the hyperparameters of a CNN in order to increase accuracy in the context of FER using random search algorithm applied on a search space defined by discrete values of hyperparameters. The dataset used for the research is FER2013 [4], to recognize seven emotions fear, happy, surprise, sad, neutral, disgust and angry. The research deals with the study of impact of random search algorithm for hyper-parameter tuning on the CNN model.

Pane et al. [27] have done their research to improve the accuracy using Random Forest classifier along with grid search optimization for tuning the parameters. The emotions recognized are happy, angry, sad and relaxed. The dataset used for this research is DEAP [2] dataset. From the papers mentioned above, parameter tuning algorithms like GA, Random Search, Grid Search have been used to improve the performance accuracy of the DL models. Literature review is done to study the various parameter tuning algorithms.

It has been proved that the grid search parameter tuning approach is excessively slow, whilst the random search parameter tuning algorithm is somewhat quicker but does not give best results after hyper-parameter tuning [24]. For our thesis work, along with a proper dataset training size, we have chosen to perform Bayesian Opti-

mization for hyper-parameter tuning to improve the DL model's accuracy for human emotion recognition.

Chapter 4

Method

The method chosen to be performed for the thesis is experimentation. The DL model is trained with the FER2013 dataset. We have used a proper training size dataset and performed Bayesian Optimization for hyperparameter tuning to increase the accuracy of the human emotion recognition model. The model is trained and validated to find the best accuracy and the obtained results have been analysed. The steps followed in the method is as follows:

1. Data preprocessing to obtain quality data and perform efficient data analysis by removing mismatched data types, mixed data values, data outliers, noisy data and missing data.
2. Splitting the dataset into various sizes of training sets and test sets.
3. Normalizing the results using BatchNormalization.
4. Performing early stopping for avoiding overfitting training data and used ReduceLROnPlateau to reduce learning rate for metric quantity's improvement.
5. Training the DL model with each training set and analyzing the results to find the best training size dataset.
6. Performing Bayesian Optimization for hyperparameter tuning to improve the accuracy of the model.
7. Testing the model with the test set and analysing the obtained results.

   The Figure 4.1 below illustrates the flowchart showing the various steps followed during the process of experimentation in the method section and consists of 7 main steps:

   - Data preprocessing.
   - Dataset train-test split.
   - Normalizing data.
   - Early Stopping and ReduceLROnPlateau.
   - Building the model with 7 pairs of train - test sets to find optimal dataset ratio.
   - Bayesian optimization for hyperparameter tuning.
   - Building the algorithm using optimal train set size and test it with test set.

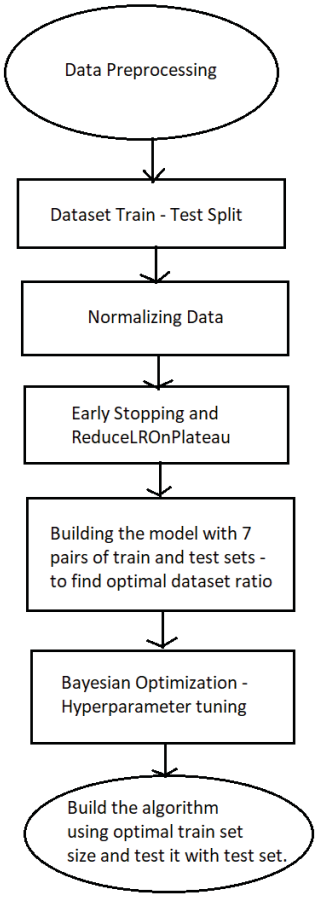

Figure 4.1: Flowchart illustrating the experimentation steps followed in the method process.

4.1 Data Preprocessing

The dataset used for the thesis is FER2013 dataset [4]. It is taken from Kaggle and consists of approximately 32,298 gray scale images of human facial emotions which are classified into seven categories anger, disgust, fear, happiness, sadness, surprise and neutral. The dataset is loaded using the Pandas library and instances are viewed using head method.
We perform data preprocessing on the dataset to remove mismatched data types, mixed data values, data outliers, noisy data and missing data. This produced quality data and helped us in performing efficient data analysis. Below is Figure 4.2 showing a barchart depicting the 7 emotion classes of the FER2013 dataset before data balancing. The x-axis representing the number of classes and the y-axis representing the count of samples present in the dataset.

The model built using an imbalanced data can have a negative impact on the

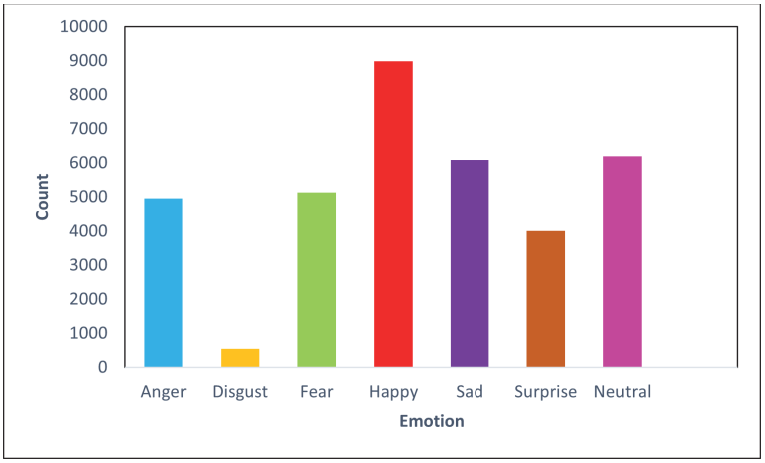

Figure 4.2: Barchart illustrating the FER2013 dataset's seven emotion classes.

performance of the model, as in such cases the model will learn the majority class more and may ignore or learn the minority class less which might lead to poor performance. To handle this issue, the dataset was balanced using the resampling technique. If the class has more number of instances (happy), we performed under-sampling by deleting few instances from the class. If the class has less number of instances (anger, disgust, fear, sadness, surprise and neutral), we performed over-sampling by adding copies of the instances to the class. Adding copies of samples was done as there should be enough samples for the model to learn and evaluate the outcomes and without this step the dataset will be imbalanced and classification will become tough as the majority class will be learned by the model and the rest classes will not be trained well. Hence, we have resampled all the classes of the dataset where each class was assigned with 8,500 image instances equally. Below is the Figure 4.3 showing a barchart depicting the 7 emotion classes of the FER2013 dataset after data balancing.

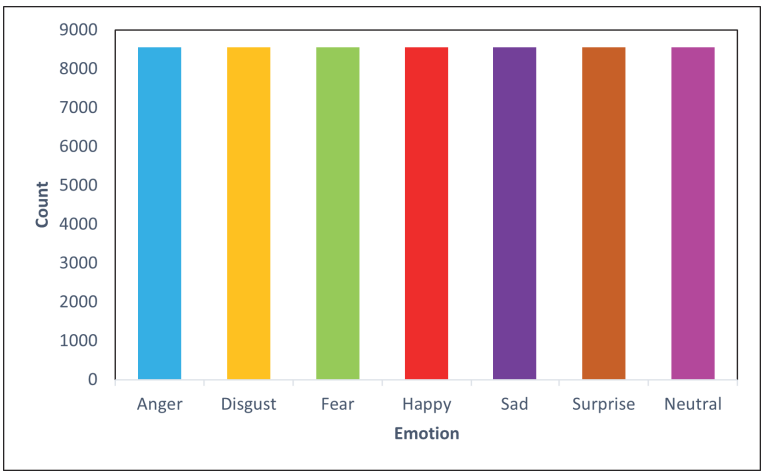

Figure 4.3: Barchart representing 7 emotion classes of the FER2013 dataset after data balancing with 8500 samples in each emotion class.

4.2 Dataset Train-Test Split

Using the Scikitplot library, the train set split function was used to split the whole dataset into multiple sizes of 7 train sets and 7 test sets. We have considered training set sizes ranging from 20% - 80%. In the later steps, the DL model has been trained with each size of the training set starting from (20, 30, 40, 50, 60, 70, 80) percentages of the total dataset and the obtained multiple results for each training set size have been analysed to study and find out the best training set size with which the DL model produced the best accuracy.

4.3 Normalizing Data

As the neural networks are very sensitive to unnormalized data, we have normalized the results for producing better results. This has helped to make the neural network work faster with much more stability of the layers. The main goal of normalizing the photos with 255 is to minimize exploding gradients caused by the wide range of pixels [0, 255], as well as to enhance convergence performance.

The next step followed in this process is BatchNormalization. By removing the batch mean and dividing by the batch standard deviation, batch normalization normalizes the output of a preceding layer. The outputs are scaled to train the network more quickly and speeds up the learning process. It also eliminates issues caused by incorrect parameter initialization. It regulates the inputs to activation functions, making them feasible and gives overall better results. With a regularization effect, it adds noise to minimize overfitting. Layer outputs are scaled to have a mean of 0 and variation of 1.

If we consider a network that learns an x-to-y function, the change in the distribution of the input x is referred to as the internal covariance shift. If anything changes, our network will be inefficient and unable to generalize. As a result, we will have to train all over the place. The network is efficient if the inputs to each layer are distributed evenly. To fight the internal covariance shift, batch normalization standardizes the distribution of layer inputs. It regulates the quantity of movement of the concealed units.

As it is a CNN model, we have utilized dropouts at regular intervals for generalization purpose. In the BatchNormalization process, we chose Exponential Linear Unit (ELU) as the activation function which is a ReLU-based activation function that defines function smoothness when the inputs are negative. It is used as it eliminates the problem of dying Rectified Linear Unit (ReLU), and also outperformed LeakyReLU in this scenario. The 'he_normal' kernel initializer is also chosen to be used since it is more appropriate for ELU. We made sure to utilize less dropout because dropout itself usually adds noise. We have set padding value to 'SAME' while training the model because the input size is same as the output size.

4.4 Early Stopping and ReduceLROnPlateau

4.4.1 Early Stopping

Early stopping is a form of regularization to prevent overfitting while training a model. We have used this technique to prevent overfitting of dataset while training our DL model. The technique of overfitting uses rules to provide the guidance regarding number of iterations that can be run before the model starts to overfit the data sets. The important parameters of the early stopping callback that we have used while training model are as follows:

- monitor: It is the quantity to be monitored. we have set this parameter to test_accuracy by default to monitor this quantity. test_accuracy is the test accuracy obtained by the trained model after testing the model.
- min_delta : It is the minumum change monitored in the quantity to qualify for improvement. We have set this parameter value to 0.00005.
- patience: It is the number of epochs without improvement after which the training stops. We have set this parameter value to 11.
- verbose: This parameter gives us the choice regarding how we want to view our output of the neural network while its training. We have set this parameter value to 1 in order to make the output visible for each update. If we set it to 0, it will show nothing.
- restore_best_weights: This parameter tells whether to use best model's weight or last epoch weight. We have set this parameter to True which takes best model's weight as the value.

4.4.2 ReduceLROnPlateau

ReduceLROnPlateau is used to reduce the learning rate when a metric has stopped showing improvement. For the model to show progress, the learning rate is usually reduced by a factor of 2-10 once the learning of the model shows stagnation. This scheduler reads the quantity of the metrics, and if there is no improvement for a 'patience' number of epochs, then the learning rate is reduced.

The important parameters of the ReduceLROnPlateau scheduler that we have used while training model are as follows:

- monitor: It is the quantity to be monitored. we have set this parameter to test_accuracy by default to monitor this quantity. test_accuracy is the test accuracy obtained by the trained model after testing the model.
- factor: It is the value by which the learning rate will be reduced. We have set the value of this parameter to 0.5.
- patience: It is the number of epochs without improvement after which the training stops. We have set this parameter value to 7.

- min_lr: It indicates the lower bound of the learning rate. We have set its value to 1e-7.
- verbose: This parameter gives us the choice regarding how we want to view our output of the neural network while its training. We have set this parameter value to 1 in order to make the output visible for each update. If we set it to 0, it will show nothing.

4.5 Training and Testing the model

The DL model's training and testing have been performed with different sizes of the training set and test set. The package used for train-test split was imported from sklearn library. With the use of TensorFlow backend, we have built the DL model by training the model with various sizes of the training sets and test sets. We have considered training set sizes starting from test_size value set to 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 respectively. The accuracy of the model for each training set was calculated and analyzed. The model is then tested with respective test set and the obtained results have been analyzed which is discussed in the Results section. The training set size with which the DL model gives the best accuracy was used along with hyperparameter tuning in the next step to find out the best accuracy of the facial emotion recognition model.

4.6 Bayesian Optimization - Hyperparameter tuning

After finding the result of the first experiment, we have used the optimal dataset training size along with hyperparameter tuning method to find out the impact of parameter tuning in improving the model's accuracy for emotion recognition. Bayesian Optimization is the method used along with scikit library for hyper parameter tuning. The library was imported and the required parameters for tuning were defined using a search space function which is used to set the parameters that are going to be tuned for model performance evaluation.

The correct parameters defined for enhancing efficiency through tuning are epochs, batch_size and optimizer. The model was subjected to RepeatedStratifiedKFold cross validation using the sklearn toolkit to generate test sets with the same distribution of classes. The number of splits utilized in cross validation is 5. Before separating into batches, each class's samples were shuffled. Following the experiment to determine the effect of parameter adjustment on the emotion detection model, the performance metrics were evaluated accordingly.

Chapter 5

Results and Analysis

The DL model had been built and it was then trained and tested with multiple training sets and test sets of varying sizes. The respective training set sizes considered for the experimentation are 20%, 30%, 40%, 50%, 60%, 70%, 80% and their respective testing set sizes are 80%, 70%, 60%, 50%, 40%, 30%, 20%. Below are Table 5.1 and Table 5.2 showing the training set, test set and their obtained training set and test set accuracies, precision scores, recall scores and F1-scores of each train-test split.

Table 5.1: Accuracies of the model on training sets and test sets.

Training Set(%)	**Test Set(%)**	**Train Accuracy(%)**	**Test Accuracy(%)**
20	80	69.23	69.82
30	70	71.17	72.27
40	60	71.61	73.81
50	50	72.65	74.62
60	40	72.84	74.88
70	30	73.49	75.07
80	**20**	**74.51**	**75.47**

Table 5.2: Precision, Recall and F1 scores of the model on test sets.

Training Set(%)	**Test Set(%)**	**Precision(%)**	**Recall(%)**	**F1-score(%)**
20	80	65.00	58.00	62.00
30	70	66.00	67.00	66.00
40	60	66.00	69.00	68.00
50	50	69.00	70.00	70.00
60	40	71.00	71.00	70.00
70	30	72.00	72.00	72.00
80	**20**	**72.00**	**73.00**	**73.00**

After training and testing the model with different sizes of the training set and test set, the best train-test split ratio with which the DL model had obtained a maximum test accuracy of 75.47% is found out to be 80% training set and 20% test set. Initially the accuracy rate was 69.82% and there was a gradual increase later.

The precision score has shown a gradual increase, starting from 65.00% with the increase in the size of the training set. The highest precision score of 72.00% had been obtained with 80% training set and 20% test set.

The recall score has shown a gradual increase, starting from 58.00% with the increase in the size of the training set. The highest recall score of 73.00% had been obtained with 80% training set and 20% test set.

The F1 score has shown a gradual increase, starting from 62.00% with the increase in the size of the training set. The highest F1 score of 73.00% had been obtained with 80% training set and 20% test set.

Therefore, the optimal dataset training size for the DL model to provide the best accuracy for emotion recognition is found to be 80% training set and 20% test set. The first experiment has been followed by the second experiment where the obtained optimal dataset training size has been used along with Bayesian Optimization for hyperparameter tuning of the DL model to improve the DL model's accuracy for facial emotion recognition.

Table 5.3: Evaluation metrics of the DL model after Bayesian Optimization

Evaluation metric	**Value (%)**
test_accuracy	78.84
Accuracy	77.21
Precision score	75.00
Recall score	73.00
F1-score	73.00
Loss	68.71
test_loss	64.92

Table 5.3 above shows the evaluation metrics of the DL model after Bayesian Optimization. The DL model has obtained the highest performance of 75.47% accuracy with 80% training set and 20% test set. After hyperparameter tuning using Bayesian Optimization, the model has improved the performance of the DL model by 3.37% giving the best accuracy of 77.21% training accuracy and 78.84% test accuracy. Precision, recall and F1 scores are 75.00%, 73.00% and 73.00% respectively. The improvement of model performance is due to the usage of optimal dataset training size combined with Bayesian Optimization for hyperparameter tuning of - batch size, epochs, and optimizer parameters with values of 32, 100, and 'Nadam' accordingly.

Chapter 6

Discussion

The DL model has been built using CNN and Multiclass classification along with Bayesian Optimization for hyperparameter tuning on the FER2013 dataset. The model has been trained and tested 5 times using RepeatedStratifiedKFold cross validation to achieve accurate results. The FER2013 data has been preprocessed and split into varying sizes of training sets and test sets for finding the proper training size set with which the model gave best accuracy. The obtained results have been analyzed and compared to find the best accuracy of the DL model. Our thesis work is intended to help develop a facial emotion recognition model with good accuracy rate. The model will be helpful in recognising emotions of human faces in various fields like medicine and computer vision. In this section, the RQ's are answered in detail.

RQ1: What is the optimal dataset training size for the DL model to provide the best accuracy for emotion recognition?

Answer: The training size of the dataset is an optimal factor which influences the accuracy of the model. Training a model with a minimal dataset training size may result in a low accuracy model. Hence, training the model with sufficient amount of data will help increase the overall performance of the model and will produce better accuracy after testing the model with the test set.

By training and testing the DL model numerous times, the appropriate dataset training size was determined. When the size of the training size was increased or decreased, it showed a considerate impact on the accuracy of the model. The obtained results were compared to find out the best accuracy of the model as shown in the Results section. The DL model produced best accurate result when the dataset was split into 80% training data and 20% test data as shown in Table 5.1. Therefore, this is the optimal dataset training size for the DL model to provide the best accuracy for emotion recognition.

RQ2: Considering the initial parameter setups of the proposed model, to what extent can parameter tuning improve the model's accuracy for emotion recognition?

Answer: We summarized the effect of parameter tuning on the model's accuracy for the emotion recognition task in this thesis work. We studied the various performance metrics of the DL model after performing hyperparameter tuning with

Bayesian Optimization. The model has been trained and tested multiple times to obtain accurate results. Parameter tuning has not only boosted the model accuracy but also decreased the model run time. The experiment produced the following results as shown below in Figure 6.1, Figure 6.2, Figure 6.3:

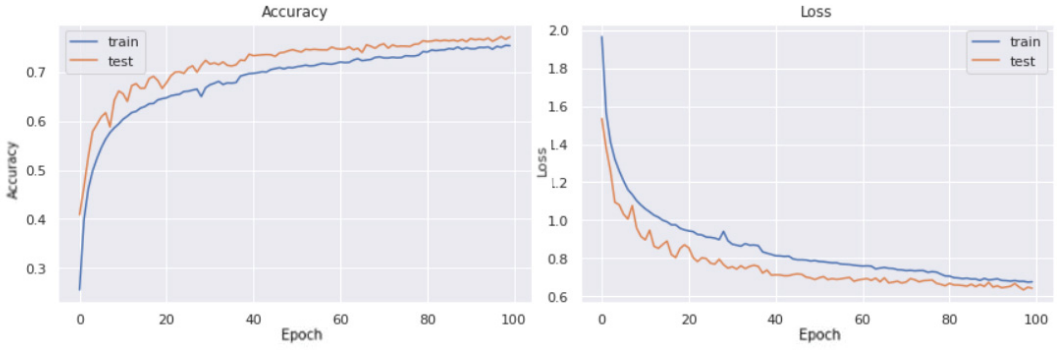

Figure 6.1: Emotion recognition model accuracy and loss graphs (x-axis: number of epochs, y-axis: model's accuracy/loss)

The graphs above in Figure 6.1 depicts the accuracy and loss of the developed emotion recognition model. The x-axis represents the number of epochs and the y-axis represents the accuracy of the model. The blue line indicates the accuracy of the training set and the orange line indicates the accuracy of the test set in both accuracy and loss graphs. The epochs history has shown gradually increase in the accuracy and has achieved 77.21% accuracy on training set and 78.84% on test set. Batch_size of 32 and 64 have been used in this process. Batch_size of 32 has performed the best and has shown an improvement towards the overall accuracy of the DL model. The optimizers used is Nadam.

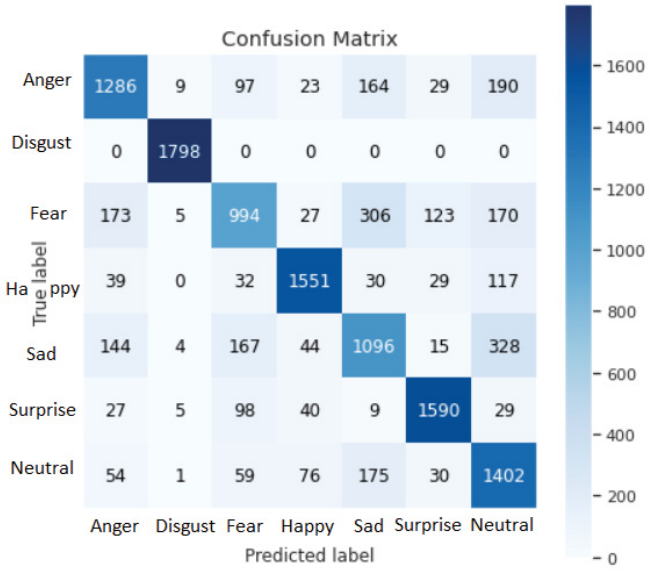

Figure 6.2: Confusion matrix of the emotion classes (x-axis: predicted label, y-axis: true label)

In the above Figure 6.2, the confusion matrix of the predicted classes is shown. The x-axis depicts the predicted label and the y-axis depicts the true label of the classes.

The confusion matrix is used to assess how well classification models perform with a given set of test data. Using the confusion matrix, we can calculate the model's different parameters such as accuracy, precision, and so on. The real and predicted labels have a range of 0 to 6 emotion classes. The classifier has predicted a total of 12,585 predictions. There are 9,717 right guesses and 2,868 wrong guesses out of 12,585 total predictions. Disgust emotion has showed positive results with other emotion classes too as there were zero wrong predictions regarding this class. Surprise and happy emotion classes are ranking the next highest with right predictions. Neutral emotion being predicted well, it ranks the next in the emotion class. Sad and fear emotion classes rank the last with least predictions respectively.

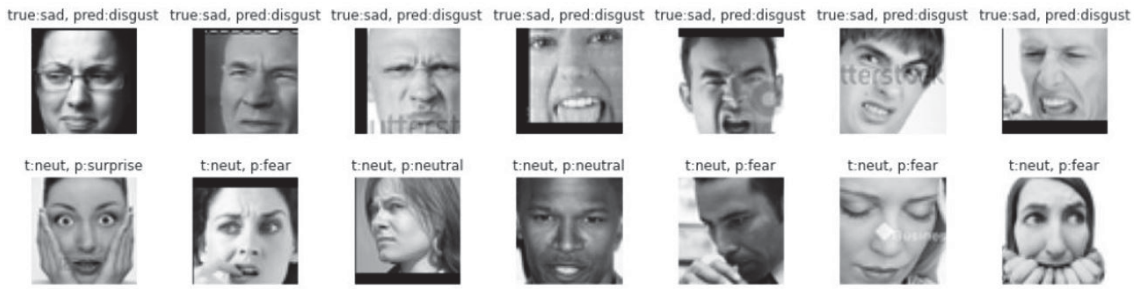

Figure 6.3: Outcome of the emotion recognition model

The above Figure 6.3 shows the outcome of using the FER model. The model has recognised most of the images right. As shown in the confusion matrix, the model has made few wrong predictions regarding neutral and sad emotions. If we look at the Figure 6.3, few images look more like neutral rather than sad and our model even predicted it neutral and the last image in the second row is neutral but is predicted as sad.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Facial Emotion Recognition is the latest technology that has gained a lot of significance in the recent years. It is essentially helpful in the field of medicine and computer vision with its enhancement and development over the next few years. In this thesis we have developed a deep learning model for facial emotion recognition using Convolutional Neural Network and Multiclass classification along with hyperparameter tuning using Bayesian optimization to improve the performance of the model. The optimal data ratio for the training and testing was found to be 80% and 20%, respectively for the deep learning model to provide the best accuracy for emotion recognition. Hyperparameter tuning using Bayesian optimization has improved the model's accuracy by 3.37%. The deep learning model has given the best accuracy of 78.84% with the FER2013 dataset. Hence, the impact of hyperparameter tuning using Bayesian Optimization on the model's accuracy is quite significant and helpful as it paved ways for many future works which can be later to improve the scope of the research area, which is human emotion recognition using Deep Learning.

7.2 Future Work

For future works, Bayesian Optimization can be implemented along with facial emotion recognition tasks in much better way by considering group of multiple parameters than with selective parameters to increase the accuracy of the model. The accuracy of the model can also be increased by creating a customized database to the model and training the images on a larger database. Work can be carried out to compare the results of our thesis work to other potential optimization method solutions to analyze performance of optimization method. By making use of the identified emotions in affective computing, a practical experimentation can also be done to manage several tasks such as in enhancing driver assistance systems and lie detection models to detect lies by reading human face emotions.

References

[1] "Deep learning | Nature." [Online]. Available: https://www.nature.com/articles/nature14539

[2] "EEG Dataset." [Online]. Available: https://www.kaggle.com/samnikolas/eeg-dataset

[3] "Extreme Learning Machine for Regression and Multiclass Classification | IEEE Journals & Magazine | IEEE Xplore." [Online]. Available: https://ieeexplore-ieee-org.miman.bib.bth.se/abstract/document/6035797/

[4] "FER-2013." [Online]. Available: https://www.kaggle.com/msambare/fer2013

[5] "Papers with Code - CK+ Dataset." [Online]. Available: https://paperswithcode.com/dataset/ck

[6] "Papers with Code - JAFFE Dataset." [Online]. Available: https://paperswithcode.com/dataset/jaffe

[7] "(PDF) Machine Learning Algorithms -A Review." [Online]. Available: https://www.researchgate.net/publication/344717762_Machine_Learning_Algorithms_-A_Review

[8] "UCF50 Action Recognition Dataset." [Online]. Available: https://www.kaggle.com/vineethakkinapalli/ucf50-action-recognition-dataset

[9] "Human action recognition using genetic algorithms and convolutional neural networks," *Pattern Recognition*, vol. 59, pp. 199–212, Nov. 2016, publisher: Pergamon. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0031320316000169

[10] "Comparative Analysis of Data Visualization Libraries Matplotlib and Seaborn in Python," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 10, no. 1, pp. 277–281, Feb. 2021. [Online]. Available: http://www.warse.org/IJATCSE/static/pdf/file/ijatcse391012021.pdf

[11] A. Ajit, K. Acharya, and A. Samanta, "A Review of Convolutional Neural Networks," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, Feb. 2020.

[12] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, Aug. 2017.

[13] S. S. Bellamkonda, "Facial Emotion Recognition by Hyper Parameter tuning of Convolutional Neural Network using Genetic Algorithm."

[14] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox, "Hyperopt: a Python library for model selection and hyperparameter optimization," *Computational Science &amp$\mathsemicolon$ Discovery*, vol. 8, no. 1, p. 014008, Jul. 2015, publisher: IOP Publishing. [Online]. Available: https://doi.org/10.1088/1749-4699/8/1/014008

[15] A. Boughida, M. N. Kouahla, and Y. Lafifi, "A novel approach for facial expression recognition based on Gabor filters and genetic algorithm," *Evolving Systems*, vol. 13, no. 2, pp. 331–345, Apr. 2022, company: Springer Distributor: Springer Institution: Springer Label: Springer Number: 2 Publisher: Springer Berlin Heidelberg. [Online]. Available: https://link.springer.com/article/10.1007/s12530-021-09393-2

[16] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, number: 7825 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41586-020-2649-2

[17] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Automated Machine Learning: Methods, Systems, Challenges*, ser. The Springer Series on Challenges in Machine Learning. Cham: Springer International Publishing, 2019. [Online]. Available: http://link.springer.com/10.1007/978-3-030-05318-5

[18] F. Hutter, J. Lücke, and L. Schmidt-Thieme, "Beyond Manual Tuning of Hyperparameters," *KI - Künstliche Intelligenz*, vol. 29, no. 4, pp. 329–337, Nov. 2015, company: Springer Distributor: Springer Institution: Springer Label: Springer Number: 4 Publisher: Springer Berlin Heidelberg. [Online]. Available: https://link-springer-com.miman.bib.bth.se/article/10.1007/s13218-015-0381-0

[19] C. Jain, K. Sawant, M. Rehman, and R. Kumar, "Emotion Detection and Characterization using Facial Features," in *2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE)*, Nov. 2018.

[20] X. Jia, "Image recognition method based on deep learning," in *2017 29th Chinese Control And Decision Conference (CCDC)*, May 2017, pp. 4730–4735, iSSN: 1948-9447.

[21] F. J. J. Joseph, S. Nonsiri, and A. Monsakul, "Keras and TensorFlow: A Hands-On Experience," *Advanced Deep Learning for Engineers and Scientists*, pp. 85–111, 2021, publisher: Springer, Cham. [Online]. Available: https://link-springer-com.miman.bib.bth.se/chapter/10.1007/978-3-030-66519-7_4

[22] J. Kim, S. Kim, and S. Choi, "Learning to Warm-Start Bayesian Hyperparameter Optimization," *arXiv:1710.06219 [cs, stat]*, Oct. 2018, arXiv: 1710.06219. [Online]. Available: http://arxiv.org/abs/1710.06219

[23] G. Krishna, "Micro-Expression Extraction For Lie Detection Using Eulerian Video (Motion and Color) Magnification."

[24] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: A big comparison for nas," *arXiv preprint arXiv:1912.06059*, 2019.

[25] H. M and S. M.N, "A Review on Evaluation Metrics for Data Classification Evaluations," *International Journal of Data Mining & Knowledge Management Process*, vol. 5, no. 2, pp. 01–11, Mar. 2015. [Online]. Available: http://www.aircconline.com/ijdkp/V5N2/5215ijdkp01.pdf

[26] W. McKinney, "pandas: a Foundational Python Library for Data Analysis and Statistics."

[27] E. S. Pane, A. D. Wibawa, and M. H. Purnomo, "Improving the accuracy of EEG emotion recognition by combining valence lateralization and ensemble learning with tuning parameters," *Cognitive Processing*, vol. 20, no. 4, pp. 405–417, Nov. 2019, company: Springer Distributor: Springer Institution: Springer Label: Springer Number: 4 Publisher: Springer Berlin Heidelberg. [Online]. Available: https://link-springer-com.miman.bib.bth.se/article/10.1007/s10339-019-00924-z

[28] Phung and Rhee, "A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets," *Applied Sciences*, vol. 9, p. 4500, Oct. 2019. Copyright 2019 by the Name of authors. Licensee MDPI, Basel, Switzerland. Reprinted with permission.

[29] G. Suciu, "How coooool–an attempt to incorporate aspects of verbal speech in digital communication," *PATHS OF COMMUNICATION IN POSTMODERNITY*.

[30] A. Vulpe-Grigoraşi and O. Grigore, "Convolutional neural network hyperparameters optimization for facial emotion recognition," in *2021 12th International Symposium on Advanced Topics in Electrical Engineering (ATEE)*. IEEE, 2021.

Appendix A

Supplemental Information

A.1 Best parameters for CNN Algorithm

{'batch_size' = 32,
'epochs' = 100,
'optimizer' = 'Nadam'}

A.2 Libraries

The libraries used in this thesis work are as follows:

1. **Numpy**[1] NumPy [16]is the most important python module for scientific computing. It's a python library that includes a multidimensional array object, various derived objects (such as masked arrays and matrices), and a variety of routines for performing fast array operations, such as mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and more. We have used this library in our thesis work as the implementation includes dealing with array object and their operations.

2. **Pandas**[2]

   Pandas [26] is a data manipulation and analysis software package for the python programming language. It includes data structures and methods for manipulating numerical tables and time series in particular. This library was used in our thesis work to deal with numerical tables and values.

3. **Scikitplot**[3]

   Scikit-plot [14] is a Python library that may be used to show data, models (during training), and experiment outcomes at various phases of a machine learning project. Scikit-plot is a modest attempt to give the ability to create beautiful and rapid graphs and plots. This library was used in the dataset train-test split process while training the model.

[1] `https://numpy.org/`
[2] `https://pandas.pydata.org/`
[3] `https://pypi.org/project/scikit-plot/`

4. **Seaborn**[4]

   Seaborn [10] is a python module for creating statistical visuals. It is based on matplotlib and tightly interacts with pandas data structures. Seaborn assists you in exploring and comprehending your data. Its charting functions work with dataframes and arrays containing whole datasets, doing the necessary semantic mapping and statistical aggregation internally to generate useful graphs. Its dataset-oriented, declarative application programming interface allows you to concentrate on the meaning of your charts rather than the mechanics of drawing them. This library was used in our thesis work for creating statistical plot visuals from the obtained data.

5. **Matplotlib**[5]

   Matplotlib [10] is a python toolkit for creating two-dimensional graphs and plots with python scripts. It features a module called pyplot that simplifies charting by allowing users to adjust line styles, font characteristics, and axis formatting, among other things. It can display a broad range of graphs and plots, including histograms, bar charts, power spectra, error charts, and so on. It is used in conjunction with NumPy to provide an open source MatLab alternative environment. This library was used to create bar charts, two dimensional graphs and axis formatting.

6. **Sklearn**[6]

   In python, Scikit-learn (Sklearn) [14] is the most usable and robust ML package. It uses a python consistency interface to deliver a set of fast ML and statistical modeling capabilities, such as classification, regression, clustering, and dimensionality reduction. NumPy, SciPy, and Matplotlib are used extensively in this python-based toolkit. This library was used to implement RepeatedStratifiedKFold cross validation during training process.

7. **TensorFlow**[7]

   TensorFlow [21] is a ML and artificial intelligence software library that is free and open-source. It may be used for a variety of applications, but it focuses on deep neural network training and inference. This library was used to implement deep neural network training techniques and was also used for backend support.

8. **Keras**[8]

   Keras [21] is a python-based open-source neural network library that operates on Theano or Tensorflow. It's built to be modular, quick, and simple to use. The Keras High-Level API is responsible for how we create models, define layers, and set up numerous input-output models. Keras also builds our model with loss and optimizer functions, as well as the training process using the

[4] `https://seaborn.pydata.org/`
[5] `https://matplotlib.org/`
[6] `https://scikit-learn.org/`
[7] `https://tensorflow.org/`
[8] `https://keras.io/`

fit function, at this level. This library was used to create the DL model, for defining the layers and to implement bayesian optimization for hyper parameter tuning.

Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden