iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 Issue: II Month of publication: February 2026

DOI: https://doi.org/10.22214/ijraset.2026.77624

www.ijraset.com

Call: ☎08813907089 | E-mail ID: ijraset@gmail.com

Facial Key Point Detection Using Convolutional Neural Network

Sandhya Yamala
Computer Science and Engineering

Abstract: These days, facial keypoint detection is a hot issue with many people drawn to its applications, which include the services like how old are you on Snapchat? Finding the facial key points in a particular face is the goal of facial keypoint detection, which is extremely difficult because every person has a completely diverse set of facial traits. Deep learning concepts, such as neural networks and cascaded neural networks, have been used to this issue. Furthermore, these structures produce much superior outcomes than cutting-edge techniques like dimension reduction algorithms and feature extraction. It's challenging to address the problem of facial keypoint recognition. Individual variations in facial traits can be observed due to factors such as 3D posture, size, location, viewing angle, and lighting conditions, and even within a single person. Although there has been significant progress in addressing these problems, there are still many areas where computer vision research may be strengthened. In our research, we want to use deep architectures to find the key points in each image to reduce losses for the task of detection and speed up training and testing for practical uses. As baselines, we have built two fundamental neural network architectures: a convolutional neural network and a hidden layer neural network. Additionally, we have suggested a method that uses factors other than raw input to determine the coordinates of face key points more accurately. The study's findings demonstrate the value of deep structures for face key point detection tasks, and employing the convolutional neural network model has marginally enhanced detection performance over baseline techniques.

***Keywords: Deep Learning, facial recognition, computer vision,* convolutional neural network, key point detection**.

I. INTRODUCTION

Visual communication relies heavily on the human face. Through facial analysis, people can infer a great deal of nonverbal cues, including identity, intentions, and feelings. The identification of benchmark faces key points is typically a crucial first step in computer vision to automatically extract these face features. Also, the precise identification of these crucial spots forms the foundation of most facial analysis techniques. Key point locations can provide valuable information on the shape of the face, which is useful for algorithms such as head pose estimation and facial expression recognition. Eye gaze tracking and eye recognition can be facilitated by using the facial key points surrounding the eyes to estimate the pupil center's location in advance. The three-dimensional head model is typically integrated with the important points from the two-dimensional image for face recognition. This can assist in minimizing notable alterations and increase the accuracy of the recognition. Applications such as entertainment, security monitoring, medical, and human-computer interaction can all benefit from the accessibility of facial data gathered by locating key areas on the face. Automatically identifying the location of facial key points in a facial image or video is the aim of the face key point detection algorithm. Either the advantage points, which explain the distinct positioning of the face parts, or the interpolation points, which link these advantage points to the contours and face parts, are these crucial locations.

The rapid advancements in the field of computer vision have led to an increasing number of research studies and industrial applications centered on the detection of facial key points. A computer vision task called "key point detection" seeks to locate an object, usually a human, and key points (such as the legs, arms, and head) inside the designated space. Many cutting-edge technologies rely on point detection at their core. Examples of these include facial recognition on smartphones, object tracking assistance for autonomous vehicles, and medical picture analysis.

Imagine a world in which your computer can perceive and comprehend relationships between objects in the visual world in a way that is comparable to that of a human eye. This transformational vision relies on keypoint detection, which enables computers to recognize and localize unique characteristics in images. These focal points function as reference points, enabling machines to comprehend the intricate visual. Numerous applications, such as facial emotion categorization, facial alignment, face tracking in films, and medical diagnosis applications, rely heavily on the ability to identify key points within a given face image. The problem now is to identify face key points quickly and reliably such that they may be used as part of a preprocessing method.

Detection of the important areas of the face, such as the eyes, corners of the mouth, and nose, which are relevant for different types of tasks, such as face filters, emotion recognition, and pose recognition, with the help of convolutional neural network and techniques of computer vision to perform facial Key point detection.

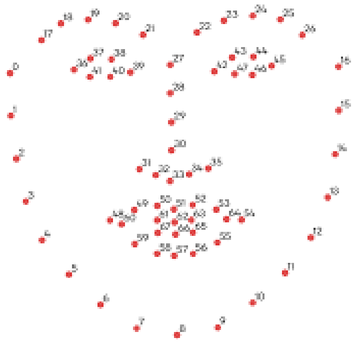


Figure 1: Sample image showing the facial key points.

In this research, we will employ deep structures for facial key point identification, which can effectively learn from many faces and greatly reduce the variation between faces of various people or conditions. Our approach uses two popular models as baselines: the convolutional neural network and the hidden layer neural network. Above all, the computation complexity for facial key point detection has been reduced because we applied the deep learning technique. Deep learning has the capacity to both better capture local characteristics and lower computation complexity by using sparsely connected layers and varying the number of different size filters. Pretrained models show significant improvement in location prediction when compared to our baseline models.

II. LITERATURE REVIEW

This section will review the various methodologies used in past efforts to detect facial important points. Initially, we will cover publications that relied primarily on image analysis to address this issue. We will then look at studies that included author and spreader information in their analysis.

The authors Levi & Hassncer [1] propose a new automatic age and gender classification system with the help of a convolutional neural networks algorithm. For this work they use the adience dataset [2] which consists of 26580 images. The suggested network design is utilized in our tests for age and gender categorization. The network consists of three convolutional layers and two fully connected layers, with a modest number of neurons. This is in comparison to the considerably bigger designs used in [3]. Their decision to use a smaller network architecture stem from both their aim to limit the danger of overfitting and the nature of the challenges they are seeking to address. Age categorization on the audience set necessitates differentiating between eight classes, whereas gender requires just two. This is in comparison to the 10,000 identification classes needed to train a network for facial recognition. The network processes all three color channels directly. Images are rescaled to 256 x 256 and cropped to 227 x 227 before being sent to the network. In the paper titled "A Robust System for Facial Emotions Recognition Using Convolutional Neural Network" [4] the authors integrated the Japanese Female Facial Expression dataset [5] with the Karolinska Directed Emotional Faces dataset [6]. The Japanese Female Expression collection includes 213 static photos of ten models. The photos are grayscale, with a resolution of 256*256. All photos have been posted. The collection contains 4900 photos from various models. All photos are grayscale and have a resolution of 572*762. The collection includes photos of 70 models. Males and females are equally distributed. The photographs were taken under identical lighting circumstances, without makeup, glasses, or earrings. Because deep learning models learn from data, our proposed system performs many pre-processing stages on each image to improve prediction. Before being included in the training dataset, each image was run through the face detection algorithm. To handle the high quantity of data required by CNN, they duplicated the data by applying different filters to each image. Pre-processed pictures of size 80*100 are sent into the first layer of CNN. They employed three convolutional layers, followed by a pooling layer and three dense layers. The thick layer has a dropout rate of 20%. The model was trained using two publicly available datasets: JAFFE and KDEF. 90% of the data was utilized for training and 10% for testing. They reached a maximum accuracy of 78.1% by combining the datasets. Furthermore, they developed a real-time emotion classification application using the suggested system and a graphical user interface.

The researchers [7], study compares the performance of 11 common machine and deep learning algorithms for classification tasks with six IoT-related datasets. The algorithms are compared based on performance criteria such as precision, recall, f1-score, accuracy, execution time, ROC-AUC score, and confusion matrix. An experiment is done to evaluate the speed with which developed models converge. Experiments showed that Random Forests outperformed other machine learning models in all performance parameters, whereas ANN and CNN produced more intriguing findings among deep learning models. We tested many models with various setups for each method to achieve optimal results for each dataset. We used 10-fold cross-validation to calculate the average accuracy of the test sets.

We strived to strike a compromise between performance indicators (particularly accuracy) and model execution times when developing machine and deep learning models. We conducted tests to analyze the performance of several machines and deep learning algorithms, including LR, GNB, KNN, DT, RF, SVM, SGD Classifier, Adaboost, ANN, CNN, and LSTM. This study distinguishes itself from others by focusing on Internet of Things-related datasets for classification challenges. We assessed our models using two distinct experiments. Two experiments were conducted: one to evaluate model performance using evaluation measures and another to assess model learning speed. The study found that RF outperformed other machine learning algorithms on most criteria, although GNB had a faster execution time. Deep learning algorithms, specifically ANN and CNN, produced the most effective outcomes.

III.METHODOLOGY

In this, we will outline the design process of a facial key point detection system. The methodology for facial key point detection typically involves several steps, including data collection, pre-processing, model selection or design, training, evaluation, and deployment. Firstly, it will introduce the datasets chosen for detecting the facial key points and, the process of feature extraction, processing, and analysis. Following this, we will cover model selection and the implementation of facial detection system. The experiments conducted are also detailed their results are discussed in the concluding section.

The main of the work is that Convolutional neural networks (CNNs) are used in face key point recognition systems to automatically identify and locate important facial features, or key points, in an input image. Examples of these landmarks include the corners of the mouth, nose, and eyes. For a variety of facial analysis tasks, including face alignment, facial tracking, facial emotion detection, and augmented reality applications, these key points provide essential information. These goals may be satisfied by a face key point detection system that uses CNNs, opening a variety of applications in computer vision, biometrics, human-computer interaction, and other areas.

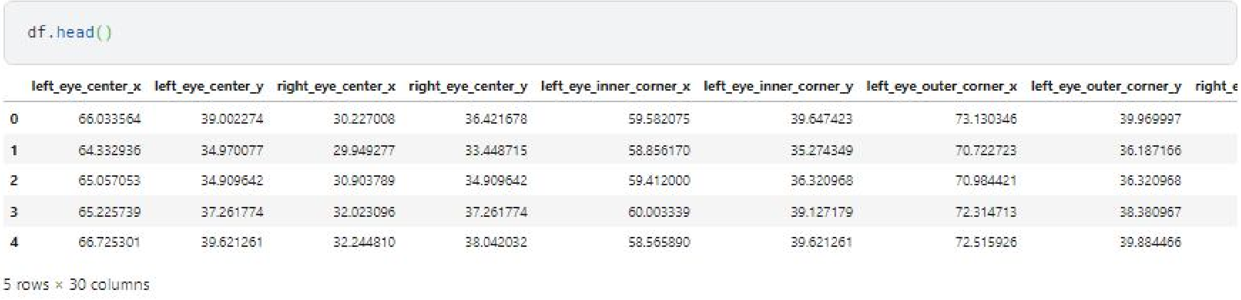
A. Dataset

For this work, we collected data from Kaggle which is an open-source platform for datasets. The data set name is “facial key points detection” [8]. There are 7049 photos in the training dataset. This dataset contains the x and y coordinates of the key points (15 fields), with the last field (Image) consisting of pixels as numbers (0-255) separated by spaces. The photos measure 96 × 96 pixels.

There are 15 key points, which represent the following elements of the face:

left_eye_center, right_eye_center, left_eye_inner_corner, left_eye_outer_corner, right_eye_inner_corner, right_eye_outer_corner, left_eyebrow_inner_end, left_eyebrow_outer_end, right_eyebrow_inner_end, right_eyebrow_outer_end, nose_tip, mouth_left_corner, mouth_right_corner, mouth_center_top_lip, mouth_center_bottom_lip.

There are various number of features present in the dataset. Few of the features of the data are shown below:



```
df.head()
```

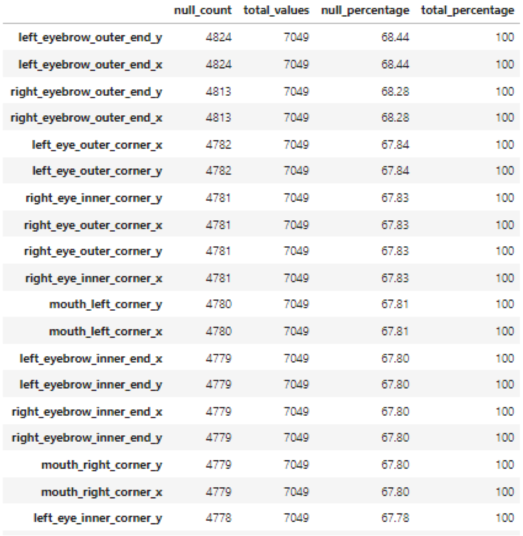
	left_eye_center_x	left_eye_center_y	right_eye_center_x	right_eye_center_y	left_eye_inner_corner_x	left_eye_inner_corner_y	left_eye_outer_corner_x	left_eye_outer_corner_y	right_e
0	66.033564	39.002274	30.227008	36.421678	59.582075	39.647423	73.130346	39.969997	
1	64.332936	34.970077	29.949277	33.448715	58.856170	35.274349	70.722723	36.187166	
2	65.057053	34.909642	30.903789	34.909642	59.412000	36.320968	70.984421	36.320968	
3	65.225739	37.261774	32.023096	37.261774	60.003339	39.127179	72.314713	38.380967	
4	66.725301	39.621261	32.244810	38.042032	58.565890	39.621261	72.515926	39.884466	

5 rows × 30 columns

Figure 2: Features of the dataset.

B. Data visualization

Data visualization is the representation of data using common graphics, such as charts, plots, infographics, and even animations. These visual displays of information communicate complex data relationships and data-driven insights in a way that is easy to understand. The below figure shows the numerical values of null values in the data.

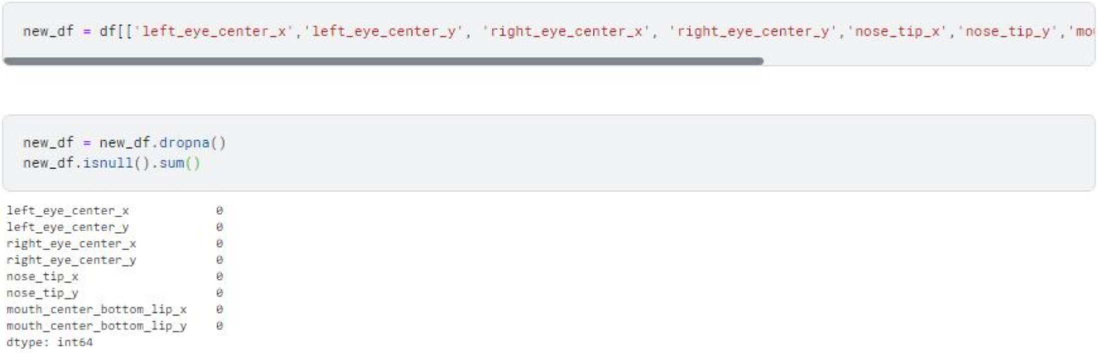


	null_count	total_values	null_percentage	total_percentage
left_eyebrow_outer_end_y	4824	7049	68.44	100
left_eyebrow_outer_end_x	4824	7049	68.44	100
right_eyebrow_outer_end_y	4813	7049	68.28	100
right_eyebrow_outer_end_x	4813	7049	68.28	100
left_eye_outer_corner_x	4782	7049	67.84	100
left_eye_outer_corner_y	4782	7049	67.84	100
right_eye_inner_corner_y	4781	7049	67.83	100
right_eye_outer_corner_x	4781	7049	67.83	100
right_eye_outer_corner_y	4781	7049	67.83	100
right_eye_inner_corner_x	4781	7049	67.83	100
mouth_left_corner_y	4780	7049	67.81	100
mouth_left_corner_x	4780	7049	67.81	100
left_eyebrow_inner_end_x	4779	7049	67.80	100
left_eyebrow_inner_end_y	4779	7049	67.80	100
right_eyebrow_inner_end_x	4779	7049	67.80	100
right_eyebrow_inner_end_y	4779	7049	67.80	100
mouth_right_corner_y	4779	7049	67.80	100
mouth_right_corner_x	4779	7049	67.80	100
left_eye_inner_corner_y	4778	7049	67.78	100

Figure 3: Numerical representation of null values.

C. Data Preprocessing

From the above analysis and plots, it is clear that only the features representing the left eye, right eye, nose tip, and mouth center bottom have a very minimal percentage of null values. They have less than 0.5% of null values. All the other features have at least 67% of null values. As the total number of null values is high, we are dropping the columns which are having a high number of null values.



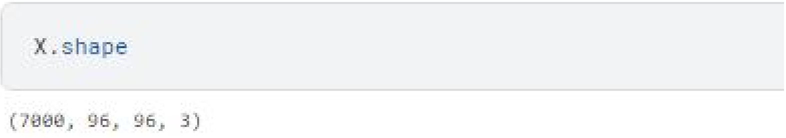
```
new_df = df[['left_eye_center_x','left_eye_center_y', 'right_eye_center_x', 'right_eye_center_y','nose_tip_x','nose_tip_y','mo
```

```
new_df = new_df.dropna()
new_df.isnull().sum()
```

```
left_eye_center_x          0
left_eye_center_y          0
right_eye_center_x         0
right_eye_center_y         0
nose_tip_x                 0
nose_tip_y                 0
mouth_center_bottom_lip_x  0
mouth_center_bottom_lip_y  0
dtype: int64
```

Figure 4: Removing the null values of the data.

After removing the null values, we have only 8 features in the dataset, which we will use for the implementation purpose. After preprocessing we get 7000 data points with 8 features. The training set consists of 7000 images with dimensions 96*96*3, which are shown below.



```
X.shape
```

```
(7000, 96, 96, 3)
```

Figure 5: Shape of the training data.

IV. IMPLEMENTATION

The implementation phase for facial key point detection typically involves several steps, including choosing the classification algorithm, training the model, validating the model, and testing the accuracy of the developed model for the new data. Firstly, it will introduce the algorithm chosen for detecting the facial key points and, the process of data split for training, validation, and testing.

A. Algorithm

This is a technique for artificial intelligence that mimics human brain activity. Algorithms that can learn from both supervised and unsupervised data make up machine learning's deep learning component. A deep neural network or deep neural learning is another term for deep learning. Higher-level characteristics are extracted from the raw input by use of a larger number of layers. Because deep learning models are commonly referred to as deep neural networks, this is because the majority of deep learning algorithms are based on the layout or architecture of neural networks.

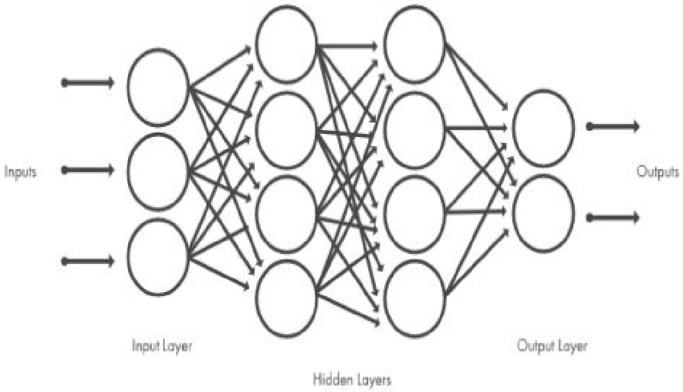


Figure 6: Structure of Neural networks.

CNN, also known as Convolutional Neural Networks, is the most widely used Deep Learning classification technique for image categorization.

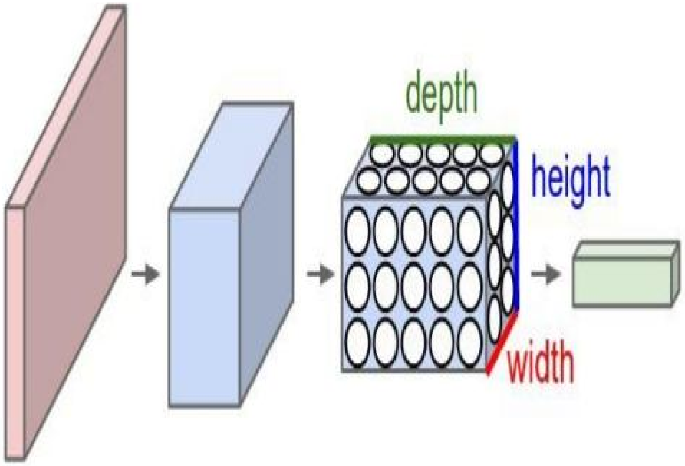


Figure 7: CNN Structure.

Convolutional neural networks (CNNs) are a particular kind of neural network that are used to process and analyze visual input, including pictures, in an efficient manner. In feature extraction and hierarchical representation learning, CNNs are made up of many layers, each of which has a distinct function. By extracting certain characteristics or objects from an image, CNN utilises the learned weights to distinguish between distinct images. ConvNet is the term for it. Reducing the pictures to a format that is simpler to process is ConvNet's operating principle.

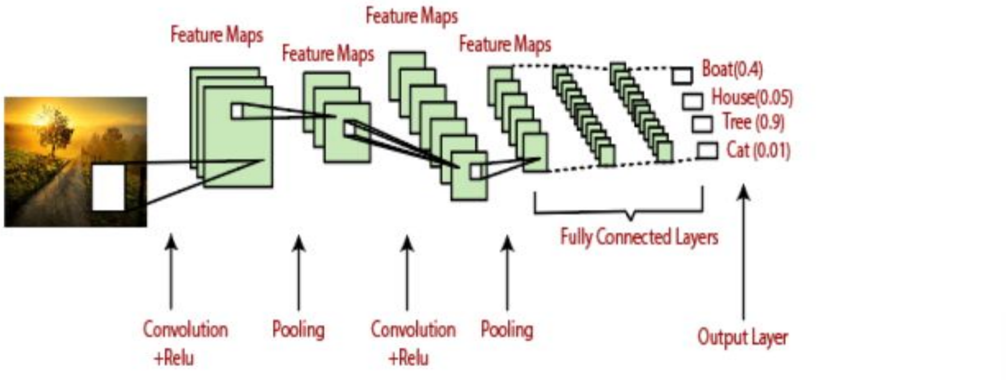


Figure 8: Architecture of CNN algorithm.

There is just one input layer and one output layer in a CNN setup with several hidden levels. The following is a list of the layers that make up the CNN hidden layers.

1) Input Layer.
2) Convolution Layer.
3) Pooling Layer.
4) Activation Function.
5) Fully Connected Layer.
6) Output Layer

B. Data splitting

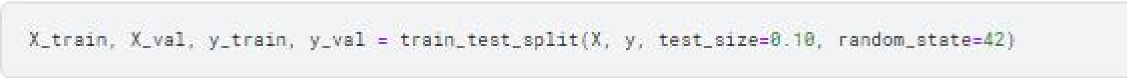
Data splitting, which divides the available information into distinct subsets for training, validation, and testing, is an essential step in machine learning and deep learning processes. This is a quick synopsis of the standard data split:

1) Training Set

The model is trained using this subset of the data. By analyzing the patterns in the data, the model learns and modifies its parameters to reduce the discrepancy between expected and actual results. Usually, between 80 and 90 percent of the dataset is made up of training sets. For our experiment, we are choosing the 90:10 ratio i.e. 90% of the data for training and the remaining 10% of the data for validation. For training we got 6300 images.

2) Validation Set:

During training, the model's performance is observed, and hyperparameters are adjusted using the validation set. To enhance generalization performance, it assists in identifying overfitting and modifying the model architecture or training procedure. The model is not trained using the validation set, which is kept apart from the training set. It typically makes up between 10 and 20 percent of the dataset. For validation, we got 700 images.



```
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.10, random_state=42)
```

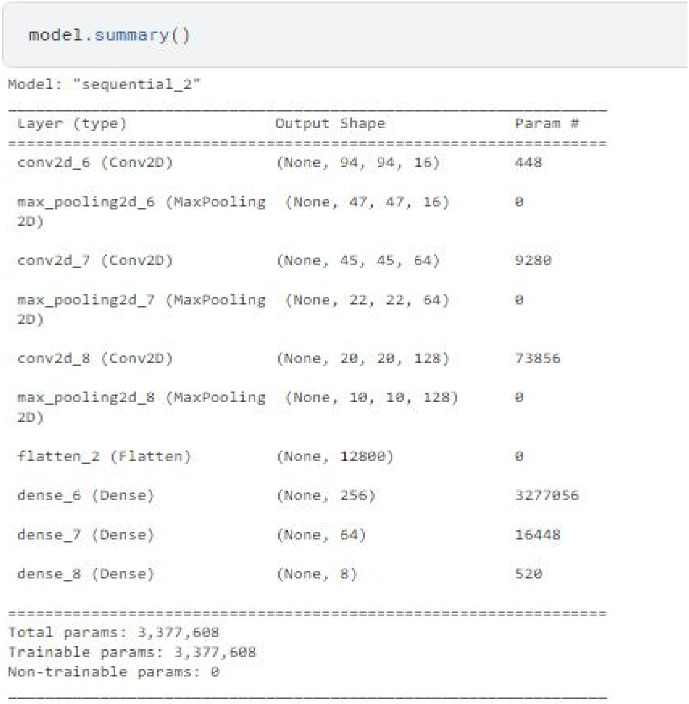
Figure 9: Splitting of train and validation set

3) Test Set:

The trained model's ultimate performance is assessed using the test set. It acts as a separate standard by which to judge how effectively the model extrapolates to unknown data. The test set is only utilised once the model has finished training, and it is entirely withheld throughout the validation and training stages. It usually makes up between 10 and 20 percent of the dataset, much as the validation set. For this dataset, we have separate data for testing. We have total 1783 images.

C. Model Architecture

Creating the CNN's architecture, which for classification tasks usually includes fully connected layers after alternating convolutional and pooling layers. Depending on the task's difficulty and the data's properties, the number of layers, size of the convolutional filters, number of filters per layer, stride, and padding configurations are chosen. Enhancing model performance, stability, and generalization may be achieved by using optional layers such as batch normalization, dropout, and skip connections.

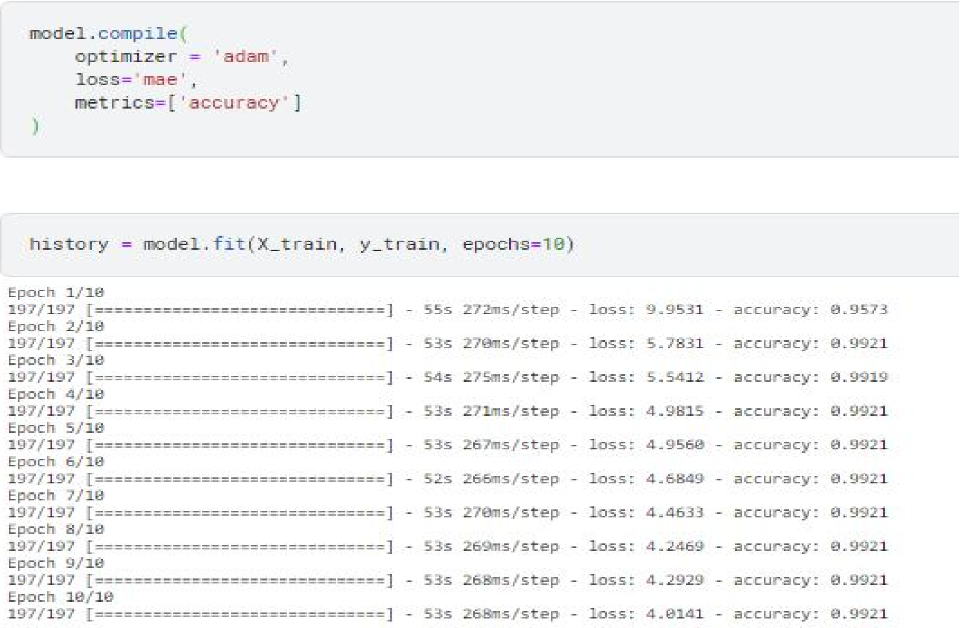


```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_6 (Conv2D)           (None, 94, 94, 16)        448

 max_pooling2d_6 (MaxPooling  (None, 47, 47, 16)       0
 2D)

 conv2d_7 (Conv2D)           (None, 45, 45, 64)        9280

 max_pooling2d_7 (MaxPooling  (None, 22, 22, 64)       0
 2D)

 conv2d_8 (Conv2D)           (None, 20, 20, 128)       73856

 max_pooling2d_8 (MaxPooling  (None, 10, 10, 128)      0
 2D)

 flatten_2 (Flatten)         (None, 12800)             0

 dense_6 (Dense)             (None, 256)               3277056

 dense_7 (Dense)             (None, 64)                16448

 dense_8 (Dense)             (None, 8)                 520

=================================================================
Total params: 3,377,608
Trainable params: 3,377,608
Non-trainable params: 0
_________________________________________________________________
```

Figure 10: CNN Model Architecture.

The next step of this implementation process is training the model. For training, we use the fit() method. For this implementation process, we choose the 10 epochs to train the model. The training is shown below.



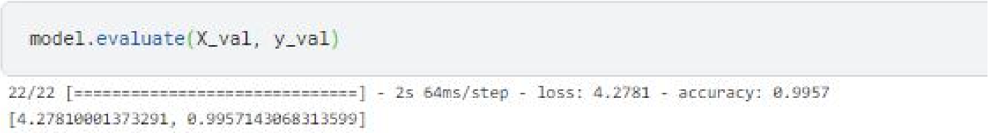
```
model.compile(
    optimizer = 'adam',
    loss='mae',
    metrics=['accuracy']
)
```

```
history = model.fit(X_train, y_train, epochs=10)
```

```
Epoch 1/10
197/197 [==============================] - 55s 272ms/step - loss: 9.9531 - accuracy: 0.9573
Epoch 2/10
197/197 [==============================] - 53s 270ms/step - loss: 5.7831 - accuracy: 0.9921
Epoch 3/10
197/197 [==============================] - 54s 275ms/step - loss: 5.5412 - accuracy: 0.9919
Epoch 4/10
197/197 [==============================] - 53s 271ms/step - loss: 4.9815 - accuracy: 0.9921
Epoch 5/10
197/197 [==============================] - 53s 267ms/step - loss: 4.9560 - accuracy: 0.9921
Epoch 6/10
197/197 [==============================] - 52s 266ms/step - loss: 4.6849 - accuracy: 0.9921
Epoch 7/10
197/197 [==============================] - 53s 270ms/step - loss: 4.4633 - accuracy: 0.9921
Epoch 8/10
197/197 [==============================] - 53s 269ms/step - loss: 4.2469 - accuracy: 0.9921
Epoch 9/10
197/197 [==============================] - 53s 268ms/step - loss: 4.2929 - accuracy: 0.9921
Epoch 10/10
197/197 [==============================] - 53s 268ms/step - loss: 4.0141 - accuracy: 0.9921
```

Figure 11: Training of the dataset.

The next step is model evaluation, Model evaluation is the process of determining how well the trained machine learning or deep learning model performs and how well it can generalise. It entails evaluating the model's performance on hypothetical data using a different dataset known as the validation set or test set.



```
model.evaluate(X_val, y_val)
```

```
22/22 [==============================] - 2s 64ms/step - loss: 4.2781 - accuracy: 0.9957
[4.278100013732 91, 0.9957143068313599]
```

Figure 12: validation of the dataset.

The next step is to evaluate the performance of the model. The efficiency with which a machine learning or deep learning model completes the job for which it was created is referred to as model performance. Its capacity to generate desired results or make precise predictions on data that hasn't been seen is how it is judged. To determine how effectively a model generalizes to new, unknown data and to inform model selection, hyperparameter tweaking, and optimization efforts, it is essential to evaluate model performance. Through the use of suitable metrics to assess model performance, practitioners may make well-informed choices aimed at enhancing the efficiency and dependability of their machine-learning models. We have so many model performance metrics, but for this implementation process, we choose the accuracy and loss of the models to evaluate the model performance.

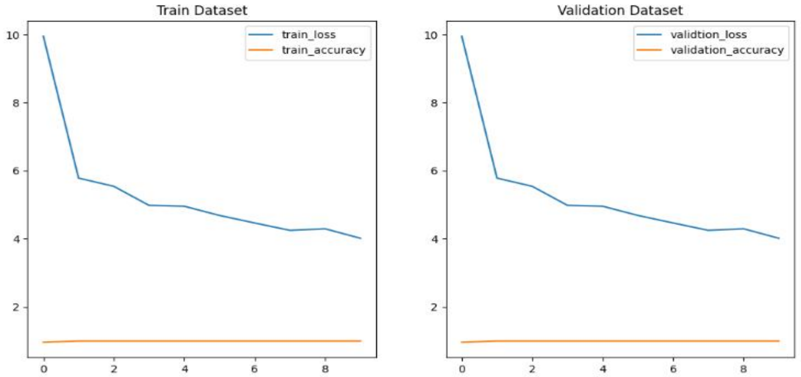


Figure 13: Performance metrics of the model.

The last step of this facial key point detection system is, to test the model performance on the unseen data. Practitioners may gain trust in the model's capacity for generalization and make well-informed judgments on its use in practical applications by thoroughly testing the model with untested data. Additionally, testing offers insightful input for continuously enhancing the performance and dependability of the model.

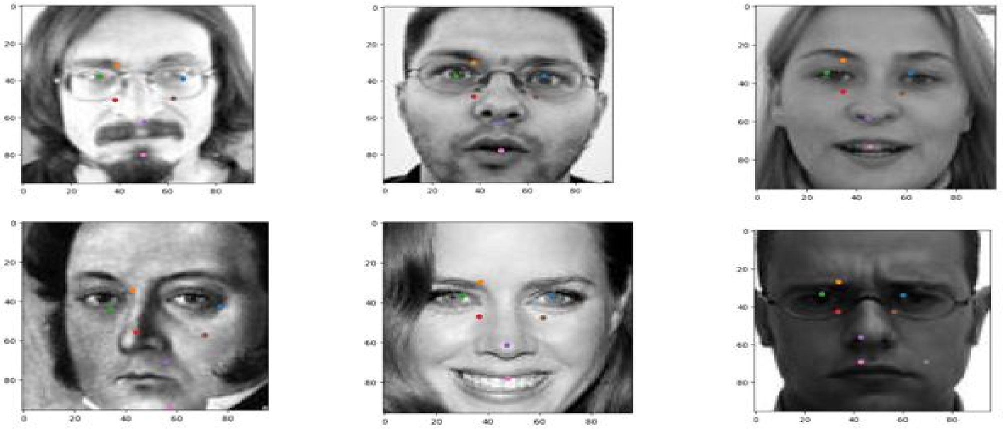


Figure 14: Testing of the model performance on unseen data.

V. CONCLUSION

In conclusion, considerable progress in the area of computer vision has been shown by the creation and assessment of a face key point identification system using convolutional neural networks (CNNs). Using CNNs and other deep learning approaches, the facial key point detection model has shown impressive performance in correctly identifying facial important points in a range of photos.

The CNN-based system's performance assessment has shown its resilience and effectiveness in identifying facial features with high accuracy and precision, even under difficult circumstances such as changing illumination, facial emotions, and postures. This demonstrates how CNNs may be used to efficiently extract and process intricate spatial connections from face pictures, which can result in more accurate and adaptable facial key point recognition systems. Even though the findings of our research are encouraging, but there are still certain limits to be aware of and opportunities for future development. Problems including biases in the datasets, the demand for computing resources, and the sporadic inability to identify important sites in harsh environments draw attention to the need for continuous study and improvement.

Overall, in our research the 99.2% accuracy achieved highlights the efficacy of the selected methodology, which most likely included rigorous data preparation, the creation of an intricate Convolutional Neural Network (CNN) architecture, cautious hyperparameter tweaking, and reliable training protocols.

REFERENCES

[1] Levi, G. and Hassncer, T. (2015) ‘Age and gender classification using Convolutional Neural Networks’, 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) [Preprint]. doi:10.1109/cvprw.2015.7301352.

[2] Harvey, A. (2014) Adience benchmark, Exposing.ai. Available at: https://exposing.ai/adience/ (Accessed: 26 February 2024).

[3] Chatfield, K. et al. (2014) Return of the devil in the details: Delving deep into convolutional nets, arXiv.org. Available at: https://arxiv.org/abs/1405.3531 (Accessed: 26 February 2024).

[4] Ghaffar, F. (2022) Facial emotions recognition using the convolutional neural net, arXiv.org. Available at: https://arxiv.org/abs/2001.01456 (Accessed: 27 February 2024).

[5] Lyons, M., Kamachi, M. and Gyoba, J. (2023) The Japanese Female Facial Expression (Jaffe) dataset, Zenodo. Available at: https://zenodo.org/records/3451524 (Accessed: 27 February 2024).

[6] Lundqvist, D., flykt, A., & hman, A. (1998). the Karolinska directed emotional faces-KDEF (CD ROM). Stockholm Karolinska Institute, Department of Clinical Neuroscience, Psychology Section. - references - scientific research publishing. Available at: https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?ReferenceID=1567781 (Accessed: 27 February 2024).

[7] Vakili, M., Ghamsari, M. and Rezaei, M. (2020) Performance analysis and comparison of machine and deep learning algorithms for IOT Data Classification, arXiv.org. Available at: https://arxiv.org/abs/2001.09636 (Accessed: 06 March 2024).

[8] Sambare, M. (2020) Fer-2013, Kaggle. Available at: https://www.kaggle.com/datasets/msambare/fer2013 (Accessed: 26 February 2024).