Project Report

Facial key point detection and recognition.

May 2025

CS 419/619 Computer Vision

Course Instructor: Dr. Surya Prakash

Indian Institute of Technology Indore

Submitted By:

Gorantla Anuksha (210001018)

Jilagam Dinesh (210001026)

Contents

1 Introduction

Facial keypoint detection is a foundational task in the field of computer vision, enabling a wide range of applications such as facial recognition, emotion detection, face alignment, augmented reality, and medical diagnostics. The objective of this project is to predict specific landmarks on human faces — referred to as key points — using machine learning techniques. Each facial key point is represented as a pair of (x, y) coordinates in the pixel space of a grayscale image.

In this project, we focus on detecting 15 predefined facial landmarks, namely:
`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`.

Accurate detection of these features is critical for enabling downstream applications and improving the interpretability of facial image data.

The dataset used for this task is sourced from a Kaggle competition and comprises 96×96 pixel grayscale images of human faces. Training set contains 7089 images, each row includes up to 15 annotated key points along with image data encoded as a flattened sequence of pixel intensities. However, some training samples include missing keypoint annotations, which introduces an additional layer of complexity. The test set contains 1783 facial images without any keypoint annotations and requires model-based predictions for submission.

The scope of this project encompasses pre-processing the image data, addressing missing keypoint values, designing and training a predictive model, and evaluating its performance. Ultimately, the goal is to develop a robust model capable of accurately predicting keypoints on unseen test images and generating a submission file in the required format.

This report outlines the methodology followed to achieve these objectives, including data pre-processing techniques, model architecture and training strategy, evaluation metrics, visual results, and the final model's performance on the test set.

2 Github Repository

Find the repository link at Facial-keypoint-detection

3 Techniques Used

3.1 Data Preprocessing

Before training the model for facial keypoint detection, the dataset was subjected to several preprocessing steps to ensure clean and consistent inputs:

- **Normalization**: Pixel values of grayscale facial images (ranging from 0 to 255) were scaled to the [0, 1] range to improve training stability and convergence.
- **Missing Value Handling**: Some keypoint coordinates were missing in the dataset. These missing values were handled using forward-fill imputation (`train_data = train_data.ffill()`), where missing values are replaced with the last available non-missing value in the column.
- **Reshaping**: Image data was originally stored as flattened vectors. Each image was reshaped to a 96×96 matrix with a single channel to be compatible with convolutional neural networks.
- **Coordinate Scaling**: In certain cases, keypoint coordinates were normalized (e.g., scaled to the [-1, 1] range or divided by 96) to improve training efficiency and handle varying image resolutions.

3.2 Data Augmentation

To enhance model generalization and increase training data diversity, various data augmentation techniques were applied only on samples that had no missing (null) keypoint values to ensure label consistency and avoid propagation of incorrect annotations:

- **Horizontal Flipping**: Each image was flipped along the vertical axis, and the corresponding x-coordinates of facial keypoints were updated using $x' = 96 - x$.
- **Rotation**: Images were rotated clockwise and counterclockwise by $\pm 12°$, and keypoints were rotated accordingly using affine transformations around the image center.
- **Brightness Adjustment**: Brightness was altered by scaling pixel values by 1.2 (increase) and 0.6 (decrease), simulating varying lighting conditions. Pixel values were clipped to remain within [0, 1].
- **Translation (Shifting)**: Images were shifted by ± 12 pixels in both directions, and keypoints were adjusted to match. Only samples with all keypoints inside the image bounds after shifting were retained.
- **Gaussian Noise Addition**: Random noise sampled from a normal distribution was added to the images to simulate sensor noise, making the model more robust to distortions.

These augmentation methods significantly increased the size and variability of the training dataset, helping the model learn robust features invariant to orientation, lighting, and noise.

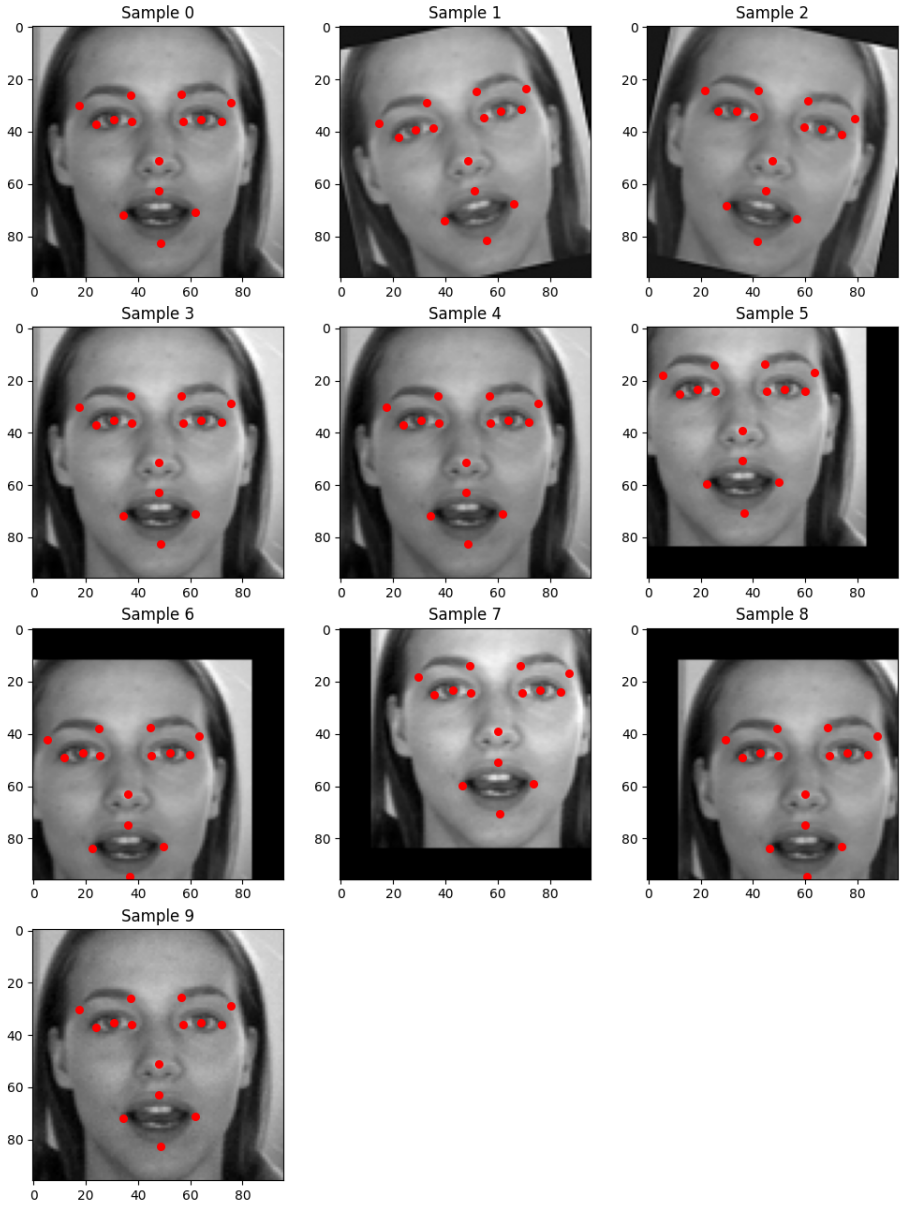

Figure 1: Augmented Images for a sample image in the train dataset

3.3 Model Architecture

The proposed convolutional neural network (CNN) for facial keypoint detection is composed of multiple convolutional blocks, each followed by nonlinear activation, batch normalization, and pooling layers. The architecture is as follows.

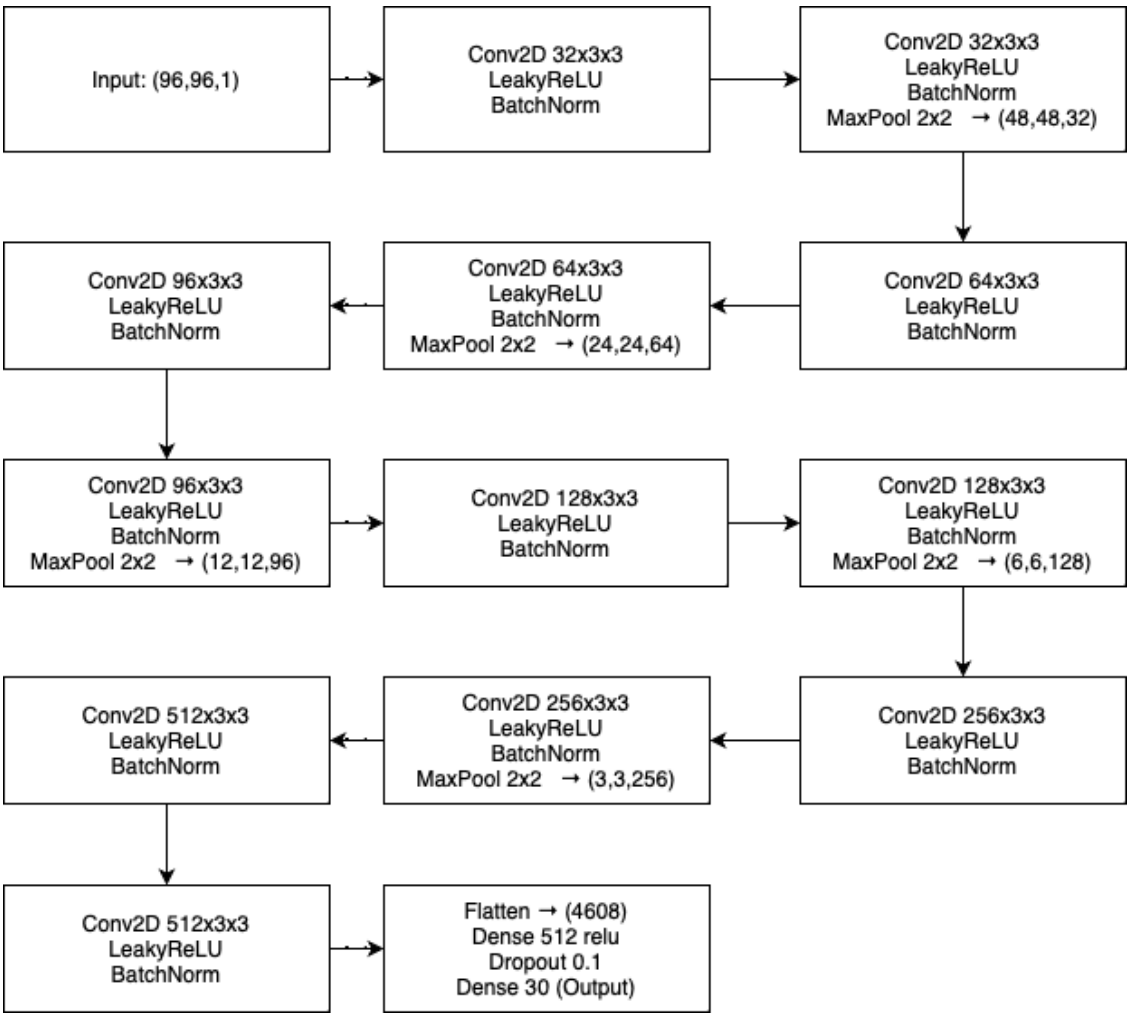

Figure 2: Model Architecture

- **Input:** Grayscale facial image of size $96 \times 96 \times 1$.
- **Convolutional Layers:** The network consists of stacked convolutional layers with increasing filter counts: 32, 64, 96, 128, 256, and 512. Each convolution uses a 3×3 kernel with 'same' padding and no bias term. Leaky ReLU [3] with $\alpha = 0.1$ is used for non-linearity.
- **Batch Normalization:** Applied after each convolutional layer to stabilize and accelerate training.
- **Max Pooling:** Performed after every two convolutional layers using 2×2 pooling to reduce spatial dimensions.

- **Fully Connected Layers:** After flattening the output, a dense layer with 512 units and ReLU [4] activation is used, followed by a dropout of 0.1 to prevent overfitting.
- **Output Layer:** A dense layer with 30 output units representing the (x, y) coordinates for 15 keypoints.

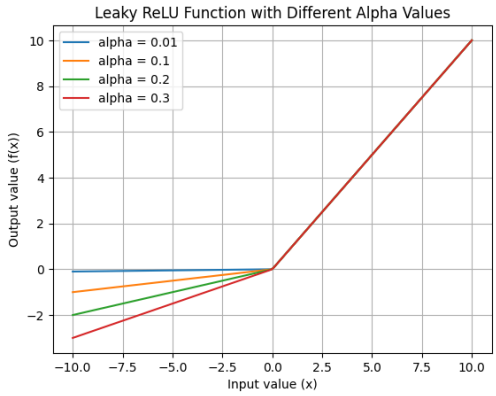

Figure 3: Leaky ReLU Activation Function

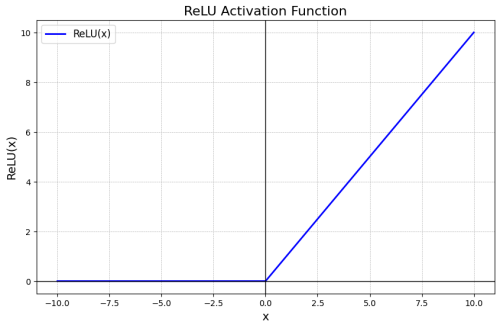

Figure 4: ReLU Activation Function

3.4 Performance Metrics Used

- **Accuracy**: can be defined as a performance metric by setting a threshold for acceptable error in key point predictions. One approach is **Key Point Accuracy** (The percentage of key points predicted within a certain error threshold (e.g., 5 pixels) from the actual location.)

$$Accuracy = \frac{\text{Number of key points with error} \leq \delta}{\text{Total number of key points}} \times 100$$

where:

  - δ is the predefined threshold (e.g., 5 pixels),

  - A key point is considered correct if $|y_i - \hat{y}_i| \leq \delta$ for both x and y coordinates.

- **Mean Absolute Error (MAE)**: Evaluates the average absolute differences between predictions and ground truth.

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|$$

- **Mean Squared Error (MSE) as Loss Function**: This metric evaluates the average of the squared differences between the predicted and actual key point coordinates.

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

3.5 Webcam-based Inference

To validate the practical applicability of our trained model, we implemented a real-time facial keypoint detection system using a webcam. This involved capturing live frames from the webcam, preprocessing them (grayscale conversion, resizing to 96×96, and normalization), and passing them through the trained model for inference.

The predicted keypoints were then rescaled to the original frame size and overlaid on the video stream for visualization. This allowed real-time tracking of facial landmarks such as eyes, eyebrows, nose, and mouth, even under varying lighting conditions and facial expressions.

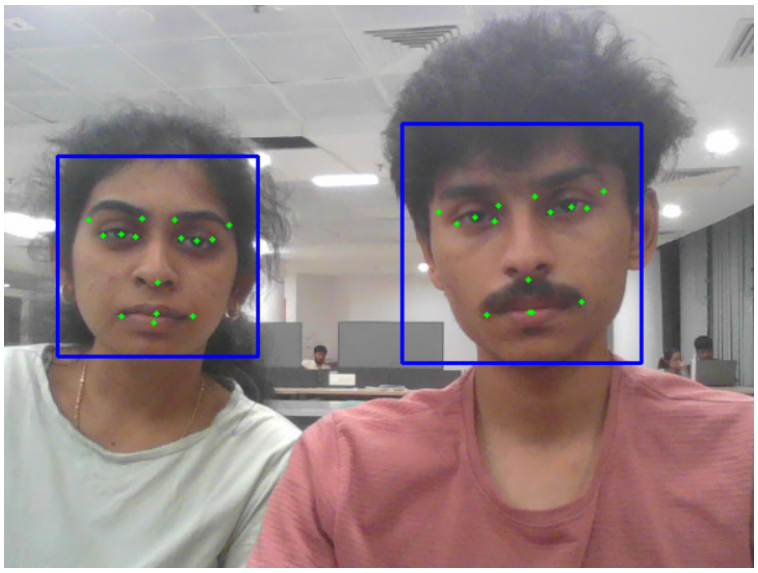

Figure 5: Live webcam inference showing predicted facial keypoints.

4 Results

The trained facial keypoint detection model was evaluated over 100 epochs using accuracy and Mean Absolute Error (MAE) as the primary performance metrics. The model's performance was tracked on both the training and validation datasets.

- **Final Training Accuracy:** 90.42%
- **Final Validation Accuracy:** 91.84%
- **Final Training MAE:** 1.1907
- **Final Validation MAE:** 0.5294
- **Final Training Loss:** 2.4840
- **Final Validation Loss:** 0.4836

The model was submitted to a Kaggle competition, and the scores were as follows:

- **Kaggle Private Score:** 1.77190
- **Kaggle Public Score:** 2.00421

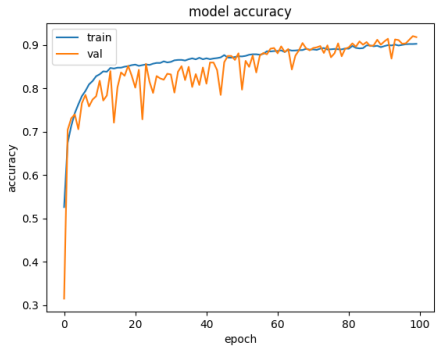

Figure 6: Training and validation accuracy over epochs.

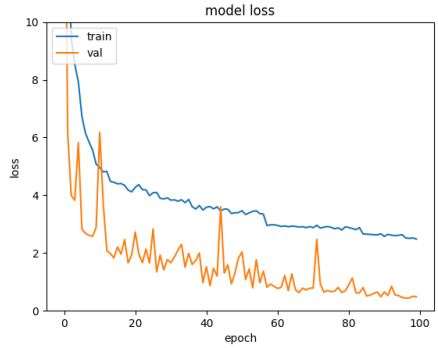

Figure 7: Training and validation loss over epochs.

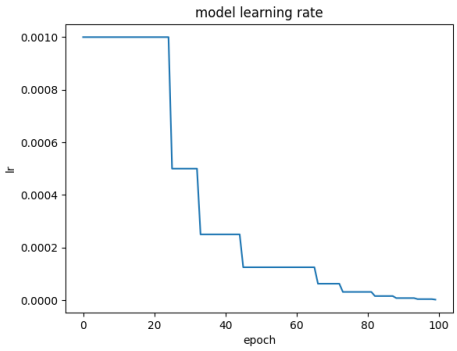

Figure 8: Learning Rate over epochs.

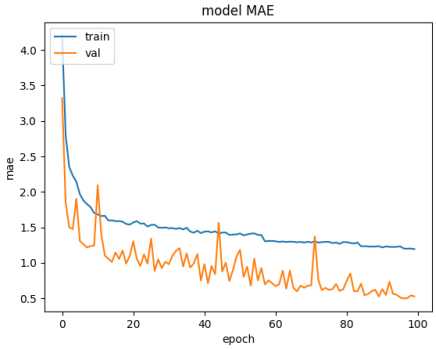

Figure 9: Training and validation MAE over epochs.

Submission and Description	Private Score	Public Score
submission.csv Complete · 6d ago	**1.77190**	**2.00421**

Figure 10: Score of our model in the Kaggle competition

Additionally, qualitative results of keypoint detection were visualized on test samples. As seen in Figure 11, the model accurately localized facial landmarks such as the eyes, nose, and mouth across various subjects.

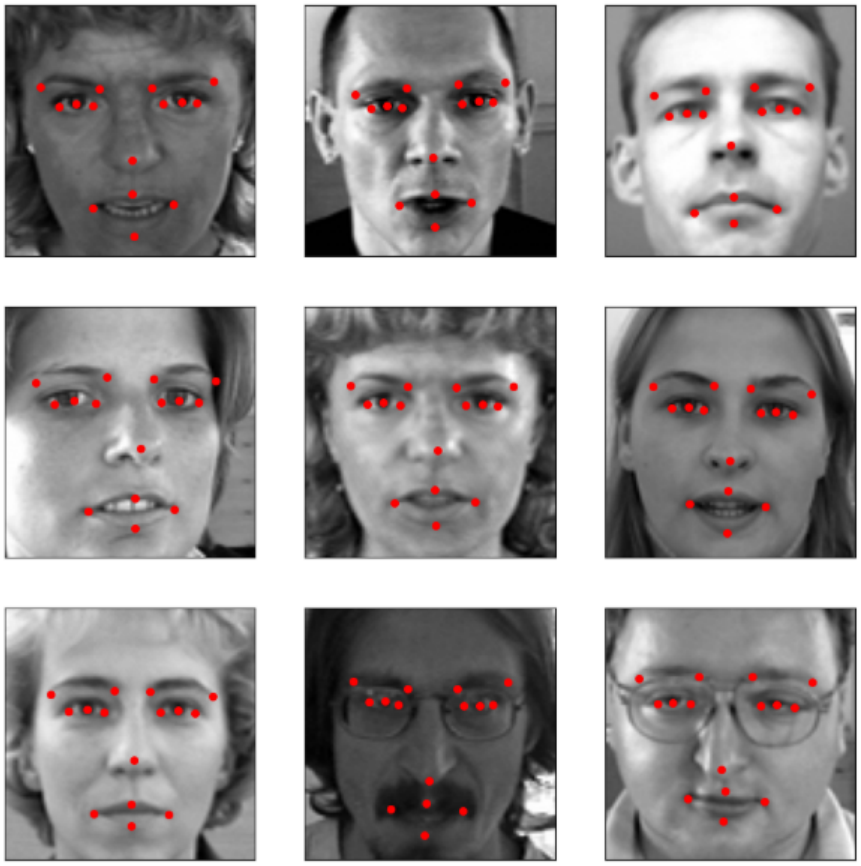

Figure 11: Predicted keypoints on sample test images.

5 Discussion on the Results

The facial keypoint detection model demonstrated stable and effective learning throughout training, as evidenced by the plots in Figures 6 and 7. The model achieved a final training accuracy of 90.42% and validation accuracy of 91.84%, showing strong generalization to unseen data.

The Mean Squared Error (MSE) steadily decreased across epochs, with only mild fluctuations in validation loss beyond epoch 30, possibly due to challenging samples or minor overfitting. However, the narrow gap between training and validation curves and the consistently declining trend confirm that the model maintained generalization and robustness.

Kaggle submission results 10 further validate the model's competitiveness:

- The **Private Score** (1.77190) reflects the model's performance on a hidden portion of the test set used for final rankings.
- The **Public Score** (2.00421) represents performance on a visible subset of the test set.

These scores align closely with validation MAE, confirming that the model generalized well to external test data.

Qualitatively, Figure 11 shows the model successfully localizing keypoints like eyes, nose, and mouth under diverse conditions — including different facial expressions, slight head tilts, and occlusions such as glasses. This indicates that the model captured spatial dependencies among facial features effectively, aided by data augmentation techniques.

6 Conclusion

This project highlights the critical role of data augmentation in enhancing the performance and generalizability of facial keypoint detection systems. Techniques such as horizontal flipping, brightness variation, and normalization were instrumental in addressing dataset limitations and improving the model's robustness to real-world variations in facial orientation and lighting conditions.

Building upon this enriched dataset, a carefully designed convolutional architecture was trained to accurately predict 15 facial keypoints from grayscale images. The model demonstrated strong performance, achieving a validation accuracy of 91.84% and a mean absolute error of 0.5294. Its reliability was further affirmed through a private Kaggle leaderboard score of 1.77190 and consistent results in real-time webcam-based inference.

Overall, this work demonstrates that systematic data augmentation, coupled with an effective convolutional pipeline, can significantly enhance facial keypoint detection. The approach offers a practical and scalable foundation for downstream tasks in face analysis, such as alignment, gesture tracking, and human-computer interaction.