Facial Expression Detection using ResNet-50

Abstract

Facial expression detection is an essential task in computer vision aimed at identifying human emotions based on facial features. Understanding emotions is crucial for improving human-computer interaction, sentiment analysis, and automatic affective computing. This project leverages deep learning techniques, specifically a modified ResNet-50 architecture, to classify images into six distinct facial expressions: *anger*, *disgust*, *fear*, *happy*, *pain*, and *sad*. The model employs transfer learning by utilizing pre-trained weights from ImageNet and fine-tuning the network for the emotion recognition task. The trained model processes images, extracts facial features, and maps them to one of the six emotion classes with high accuracy. The system is designed to be efficient for real-time emotion recognition applications, such as healthcare, surveillance, and interactive AI systems.

1. Introduction

Facial expressions are one of the most significant ways humans communicate emotions. Detecting and interpreting these expressions is important for improving user experience in various AI applications, such as virtual assistants, healthcare diagnostics, and human-computer interaction (HCI). Traditional emotion recognition systems are often limited by complex rule-based systems, but deep learning methods, especially convolutional neural networks (CNNs), have proven to be highly effective in such tasks.

2. Model Architecture

The model used for facial expression detection is based on **ResNet-50**, a deep convolutional neural network with residual connections. Residual networks (ResNets) mitigate the vanishing gradient problem by introducing skip connections between layers. These networks are known for their deep structure, allowing them to learn more complex representations, making them highly effective for image classification tasks.

2.1 Base Model: ResNet-50

- **ResNet-50** is a variant of ResNet with 50 layers, known for its ability to extract features from images effectively, even in deeper layers.

- **Pretrained Weights**: The model was initialized with weights pretrained on the **ImageNet** dataset, which is a large-scale image dataset with over 14 million labeled images across 1000 categories. This helps in transfer learning, allowing the model to learn general features and then fine-tune them for the facial expression task.

2.2 Customization for Emotion Classification

The final fully connected layer in the ResNet-50 model was modified to predict six emotion classes, i.e., *anger*, *disgust*, *fear*, *happy*, *pain*, and *sad*.

- **Dropout Layer**: A dropout layer (with a 50% probability) was added to the fully connected layer to prevent overfitting and improve the model's generalization.
- **Softmax Activation**: After the dense layer, a softmax activation function is applied to output probabilities for each class, ensuring that the predicted label corresponds to the highest probability.

2.3 Architecture Overview

- **Input**: 224x224 RGB image.
- **Feature Extraction Layers**: A sequence of convolutional layers that learn the hierarchical representation of the input image.
- **Fully Connected Layers**: Custom dense layer with dropout and a softmax activation to output one of the six possible emotions.

2.4 Training Process

- **Dataset**: A custom dataset consisting of facial expression images from six classes: *anger*, *disgust*, *fear*, *happy*, *pain*, and *sad*.
- **Data Augmentation**: To increase the diversity of the training data and prevent overfitting, techniques like random cropping, rotation, and color jittering were applied to the training images.
- **Loss Function**: **Cross-Entropy Loss**, a common loss function for multi-class classification problems, was used.
- **Optimizer**: **Adam Optimizer** was used for its ability to adapt the learning rate and improve convergence.
- **Learning rate scheduler: ReduceLROnPlateau** was also implemented to adjust the learning rate during training.

---

3. Application Flow

3.1 Data Preprocessing

Data preprocessing is crucial for the effective training of deep learning models. The following steps were applied to the input data:

1. **Resizing**: All images were resized to 224x224 pixels to match the input size expected by ResNet-50.
2. **Normalization**: The pixel values of images were normalized using the mean and standard deviation of the ImageNet dataset. This helps the model learn more effectively.
3. **Augmentation**: Random transformations such as cropping, rotating, and jittering were applied to improve the model's robustness and generalization to unseen data.

3.2 Model Inference

Once the model is trained, the inference process begins when an image is uploaded through the user interface.

- **Input**: The user uploads an image of a face, which is then preprocessed (resized and normalized).
- **Processing**: The preprocessed image is passed through the trained ResNet-50 model.
- **Output**: The model predicts an emotion label based on the highest output probability.

This prediction is mapped to one of the six expressions using a dictionary:

```
expression_mapping = {

    0: "anger",

    1: "disgust",

    2: "fear",

    3: "happy",

    4: "pain",

    5: "sad"

}
```

3.3 Backend (Flask)

The Flask backend provides an API endpoint **`/predict`**, which accepts an image file via HTTP POST:

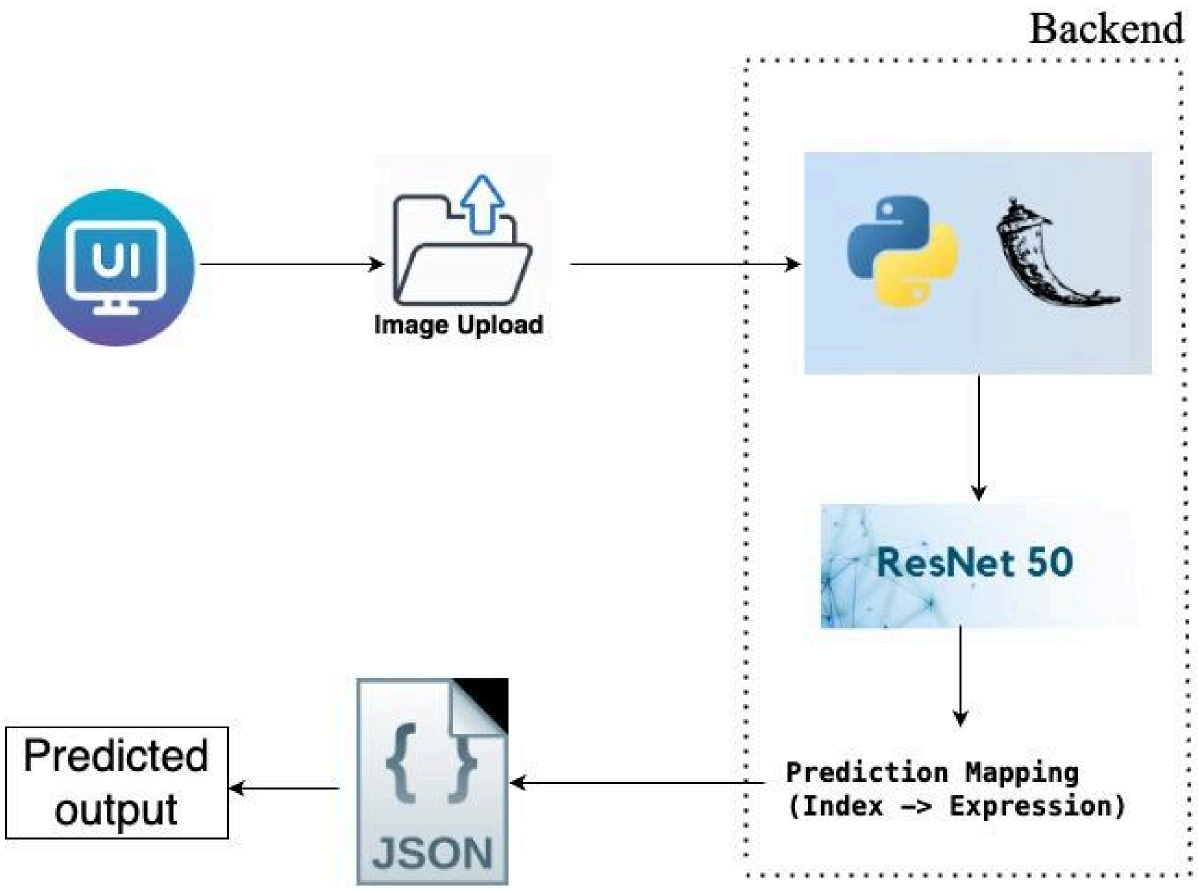
1. **Image Upload**: The user uploads an image.
2. **Preprocessing**: The image is resized and normalized to the expected format.
3. **Prediction**: The image is passed through the ResNet-50 model, and the result is returned as a JSON response with the predicted emotion.

3.4 Frontend (HTML/JavaScript)

The user interface is a simple web page that allows users to upload an image and receive a prediction. The frontend consists of:

1. **File Input**: The user uploads an image through an input field.
2. **Prediction Button**: Once the image is uploaded, the user clicks the "Predict" button to trigger the prediction process.
3. **Prediction Display**: The backend response is displayed on the web page, showing the predicted expression.

System Diagram

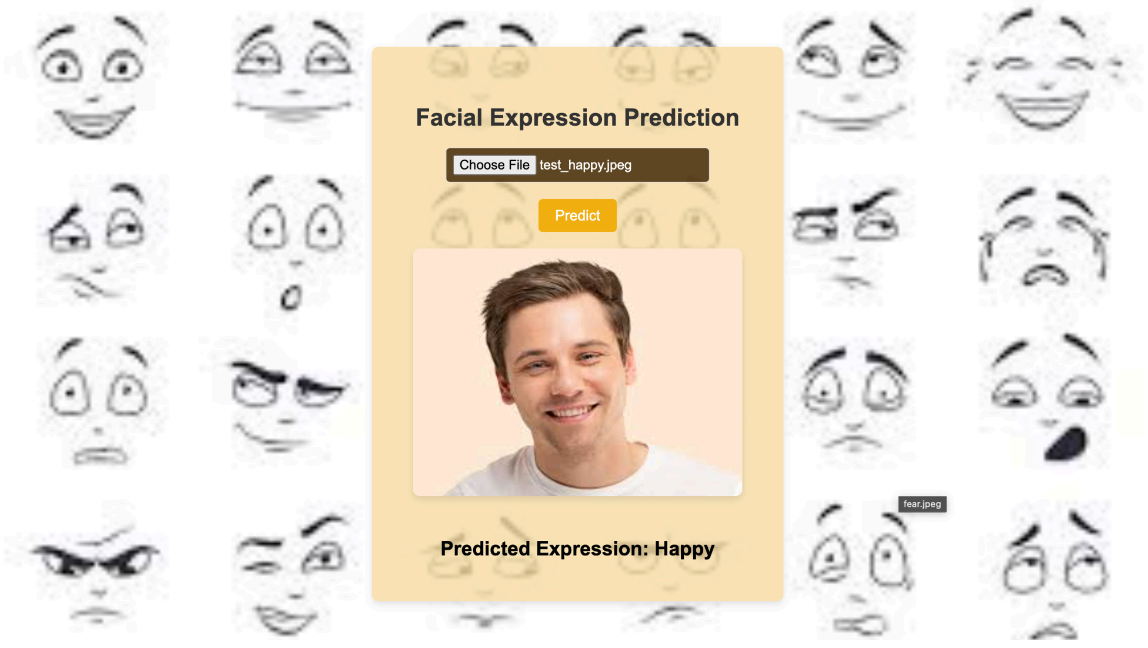


4. Results

4.1 Accuracy and Performance

The model was evaluated on a separate test dataset. Key performance metrics include:

- **Validation Accuracy**: The validation accuracy was 72%.
- **Test Accuracy**: The model performed with 74% accuracy on the test set.
- The model consistently classified the six emotion categories with a good degree of accuracy.
- UI (Output)



---

5. Key Applications

1. **Healthcare**: Real-time emotion analysis for assessing patient mental health and emotional states.
2. **Customer Interaction**: Enhancing user experience in AI-driven customer service, chatbots, and virtual assistants.
3. **Surveillance**: Monitoring emotional states of individuals in public spaces for safety and behavior analysis.
4. **Education**: Monitoring students' engagement and emotional responses during online learning or exams.
5. **Gaming**: Integrating emotion recognition into gaming systems to adjust gameplay based on players’ emotions.

---

6. Challenges

1. **Data Quality**: The performance of the model depends on the quality and diversity of the dataset.
2. **Class Imbalance**: Some emotions may be underrepresented in the dataset, which could lead to biased predictions.
3. **Generalization**: The model may struggle to generalize well to unseen facial expressions or images from different lighting conditions.

---

7. Future Scope

1. **Expand Dataset**: Increase the diversity of facial expressions and emotions for better generalization.
2. **Multimodal Emotion Recognition**: Combine facial expression analysis with voice and text analysis to create a multimodal emotion recognition system.
3. **Edge Deployment**: Optimize the model for deployment on mobile or edge devices for real-time predictions in low-resource environments.

---

8. Conclusion

This project successfully demonstrates the power of deep learning in emotion recognition using a ResNet-50 model. By leveraging transfer learning, data augmentation, and robust training techniques, we were able to create a system capable of classifying facial expressions with high accuracy. The integration of a user-friendly web interface and Flask backend makes this system suitable for real-world applications such as healthcare, education, and customer service.

Code_Drive Link :
https://colab.research.google.com/drive/1GqRcESfGSXuiTn-dN-adVpZ7qedOwfYb#scrollTo=l_wiCxMqDZ_2