Artificial Intelligence

Student: Lopez Alam - 菲格罗亚
Student ID: LJ2406203

December 31, 2024

Facial Expression Recognition

The Facial Expression Recognition project implements a deep learning-based pipeline for classifying emotions from images using the FER2013 dataset. It leverages transfer learning with the VGG16 architecture, data augmentation techniques, and a structured approach for training and evaluation.

Task

Facial Expression Recognition (FER) involves identifying emotions from human facial images. This task is crucial for various applications, including:

- Human-computer interaction: Enabling systems to adapt their behavior based on user emotions.
- Healthcare: Assisting in mental health diagnosis and therapy.
- Security: Enhancing surveillance systems with emotional cues.
- Entertainment: Personalizing content in gaming and media.

The task here is to classify grayscale images into one of seven emotions: angry, disgust, fear, happy, neutral, sad, and surprise, using a machine learning model. The implementation is structured to preprocess the data, train a convolutional neural network, and fine-tune the model for optimal performance.

Dataset

The FER2013 dataset is organized into distinct training and test folders, where each folder contains subdirectories named after the seven primary emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral. All images in these subfolders are grayscale and resized to 48x48 pixels, ensuring consistency in input dimensions throughout the training process. The training set supports the model in learning the various expressions, whereas the test set serves as a validation mechanism to gauge how well the model generalizes to unseen data. This setup allows for both an effective learning phase and a reliable assessment of final performance.

Model

The chosen model is the well-known deep learning architecture VGG16, a convolutional neural network pre-trained on ImageNet. The VGG16 architecture lies at the core of the model design due to its proven success in large-scale visual recognition tasks. Originally trained on ImageNet, VGG16 comes equipped with feature extraction capabilities that can be highly beneficial when transferred to a facial expression recognition context. This practice of transfer learning significantly reduces the amount of time needed for model training because it initiates the learning process with weights that have already been adjusted from millions of images. By removing the top classification layers, the convolutional layers of VGG16 can be used as a generalized feature extractor. On top of these layers, the model architecture incorporates a Flatten operation to convert the extracted feature maps into a one-dimensional vector, followed by a Dense layer with 256 units and a ReLU activation. To prevent overfitting, L2 regularization is applied to this layer, and a Dropout layer with a rate of 0.5 is introduced. These elements help the model learn more robust patterns without becoming too specialized to the training data. Finally, a Dense output layer with softmax activation maps the extracted features into the seven emotion classes, completing the classification pipeline.

Code Structure

The code is structured in 2 main files, first the main.py where all the flow is maintained and the main deep learning functions and processes are called. Second, the utils.py where some complementary functions such as importing the dataset, plotting, etc. are created and then imported to the main.py file. This, to ensure the readability of the code.

The preprocessing or preparation of the dataset involves loading images into TensorFlow through the image_dataset_from_directory function. This approach simplifies the process of generating training and validation batches while automatically labeling images based on their folder names. During this stage, class names are extracted so that they can be referenced in later evaluations and visualizations, such as confusion matrices and sample prediction plots. In addition to straightforward loading and splitting, data augmentation significantly diversifies the training set. Random transformations—including horizontal flipping, rotation, zooming, and changes in brightness and contrast—are applied to each batch of images. These operations work to simulate different lighting conditions, viewpoints, and potential image distortions that the model may encounter in the real world. Rescaling the pixel values to the [0, 1] range also ensures numerical stability during the training process and supports faster convergence of the optimization algorithm.

Next, dataset loading and augmentation are conducted, clearly organizing the data into training and validation sets. The model is defined by initializing a VGG16 base with pre-trained ImageNet weights, removing the top layers, and adding custom layers tailored for FER2013. Training is then divided into two phases: Phase 1 keeps the base

VGG16 layers frozen and only trains the newly added layers at a learning rate of 0.0001, allowing the new layers to adapt to the FER2013-specific emotions without disrupting the valuable general features that VGG16 already captures. Phase 2 involves unfreezing the entire network to fine-tune all layers at a reduced learning rate (0.00001). This step refines both the base VGG16 representations and the new layers, often leading to significant improvements in accuracy.

To safeguard the training process and maintain model quality, two key callback functions are integrated: ModelCheckpoint and EarlyStopping. ModelCheckpoint automatically saves the model weights to a file named vgg_facial_expression.keras whenever a new best validation accuracy is reached. This ensures that, even if training halts prematurely or if performance degrades later, the best model is preserved for evaluation or deployment. EarlyStopping tracks validation accuracy over the epochs and stops training once it detects no improvement for five consecutive epochs. This prevents excessive overfitting, reduces computational overhead, and ensures that the final model is captured at its optimal point rather than continuing to train while performance plateaus or worsens.

A further mechanism in place is the pre-trained weights check. At the beginning of the program, it verifies whether a previously saved set of weights exists. If such weights are found, the model is loaded directly from those weights, bypassing the entire training process and moving on to the evaluation stage. Conversely, if no pre-trained weights are available, the model must undergo the two-phase training procedure described above, after which the best weights are saved. This structure not only accelerates repeated experiments but also underscores reproducibility and ease of model sharing.

Experimental Setup

The experiments with this setup employ an image size of 48x48 pixels and a batch size of 32. The initial learning rate is set to 0.0001, with a further reduction by a factor of 10 (to 0.00001) during the fine-tuning phase. A total of 25 epochs provides enough iterations for the model to learn complex features, although the EarlyStopping callback can end the run early if improvements stagnate. Data augmentation, including the flipping, rotation, zoom, brightness, and contrast adjustments, significantly bolsters the model's ability to cope with real-world image variations, minimizing overfitting and improving generalization performance.

Comprehensive evaluation strategies ensure thorough insights into model behavior. During and after training, accuracy and loss metrics are plotted for both the training and validation sets, allowing real-time visualization of the learning progression and identification of potential overfitting. The confusion matrix serves as a powerful tool to reveal which emotion categories the model struggles with the most. Misclassifications in

Parameter	Value/Details
Image Size	48x48 pixels
Batch Size	32
Learning Rate	0.0001 (reduced to 0.00001 during fine-tuning)
Epochs	25
Data Augmentation	Random flipping, rotation, zoom, brightness, and contrast adjustments
Rescaling	Pixel values are normalized (rescaled) to ensure consistent input to the model
Evaluation	Validation accuracy is monitored to gauge model performance; early stopping is used to halt training when improvements stagnate, preventing overfitting and ensuring efficiency

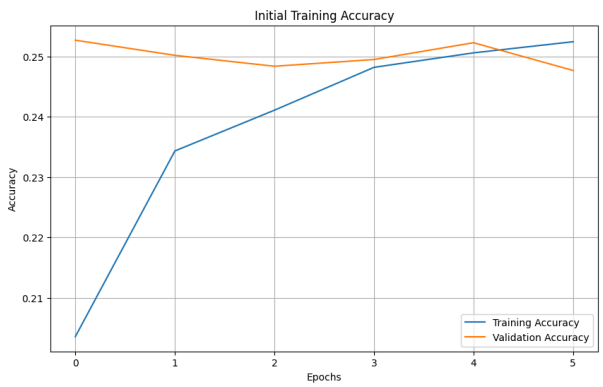
the confusion matrix often hint at certain facial expressions that share visual similarities, such as fear and surprise or disgust and anger.

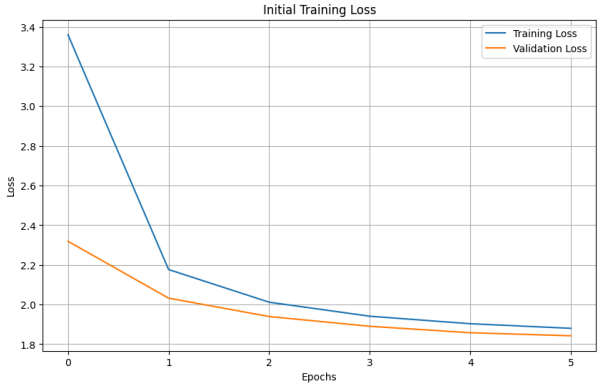
These insights not only highlight areas for potential improvement but also guide data augmentation strategies for future iterations of the project. Additionally, sample predictions on ten random test images illustrate the model's capability in practice, highlighting both correct and incorrect classifications and providing a tangible sense of how the system performs on real examples.

1. **Accuracy and Loss Over Epochs:** Plots showing training and validation accuracy/loss during both initial training and fine-tuning phases.
2. **Confusion Matrix:** Visual representation of model predictions vs. true labels to evaluate class-wise performance.

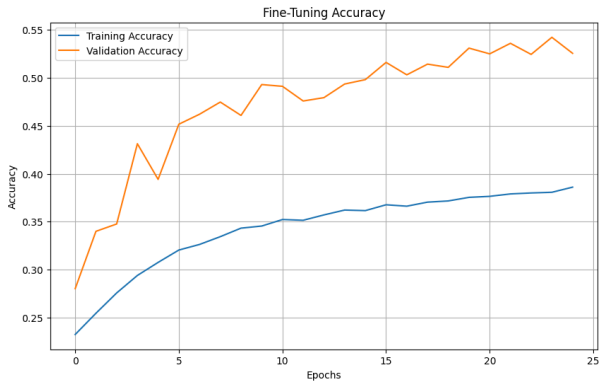
Results and Analysis

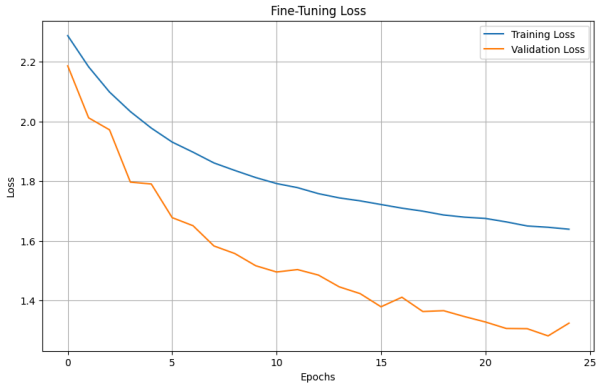
Training involves two phases: first, freezing the base VGG16 layers and training the custom layers with a learning rate of 0.0001; second, fine-tuning the entire model by unfreezing all layers and reducing the learning rate by a factor of 10.



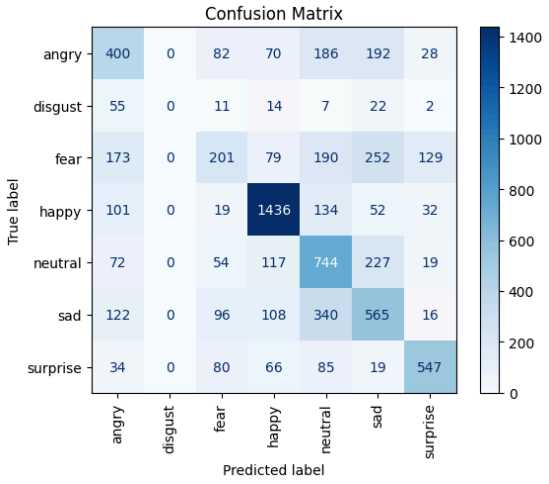


The initial training phase shows steady improvement in training accuracy as the custom dense layers learn features from the frozen VGG16 base. Validation accuracy, however, fluctuates, reflecting the limitations of training only the custom layers. Early divergence between training and validation accuracy indicates the need for further fine-tuning. The training loss decreases rapidly, showing that the custom layers are learning effectively. Validation loss aligns closely with training loss, suggesting good generalization at this stage. However, the frozen VGG16 base restricts the model's ability to adapt fully to the dataset-specific features.





Fine-tuning improves accuracy significantly. The training accuracy steadily increases as the base VGG16 layers are unfrozen, allowing the model to adapt more effectively to the dataset. Validation accuracy stabilizes around 50-55%, reflecting better generalization. The training loss continues to decrease during fine-tuning, and validation loss shows a similar trend. While there is a slight gap between training and validation loss, it remains controlled, suggesting that overfitting is minimal. Fine-tuning demonstrates the effectiveness of leveraging the pre-trained VGG16 model to enhance performance.



The confusion matrix provides detailed insights into the performance of the model for each emotion class. For the "Angry" class, 400 samples were correctly classified, but many were misclassified as "Neutral" (186) and "Sad" (192), likely due to overlapping visual features where angry expressions might appear neutral or sad depending on intensity. The "Disgust" class performed poorly, with only 11 samples correctly classified, and significant misclassifications as "Angry" (55) or "Fear" (14). This is attributed to the small number of samples in the "Disgust" class, which hampers the model's ability to learn its features effectively. Similarly, the "Fear" class saw only 201 correct classifications, with a notable number misclassified as "Sad" (252) and "Neutral" (190). This reflects the subtle nature of these emotions, which often overlap visually.

The "Happy" class was the most accurately classified, with 1,436 samples correctly identified. Misclassifications as "Neutral" (134) suggest that low-intensity smiles may be perceived as neutral by the model. For the "Neutral" class, 744 samples were correctly classified, but misclassifications into "Sad" (227) and "Fear" (117) indicate challenges in distinguishing low-intensity emotional expressions. The "Sad" class exhibited a similar issue, with 565 correct classifications but a significant number misclassified as "Fear" (340) and "Neutral" (108). Finally, the "Surprise" class had 547 correct classifications, with some misclassified as "Fear" (80) and "Neutral" (85), likely due to variations in facial intensity. Overall, "Happy" showed the highest classification accuracy due to its distinct features, while emotions like "Fear," "Sad," and "Neutral" were often confused due to subtle differences.

The visualization tools provide further insights into the model's performance. The confusion matrix highlights the importance of addressing class imbalance, as underrepresented classes like "Disgust" suffer from poor classification accuracy. The misclassification of overlapping classes such as "Fear," "Sad," and "Neutral" suggests a need for more discriminative feature learning. Accuracy and loss plots illustrate the learning progression, with significant improvements during

the fine-tuning phase. The close alignment of validation and training metrics during both initial training and fine-tuning indicates that the model is learning effectively without severe overfitting.

To further improve the model, addressing class imbalance is essential. Techniques such as oversampling, data augmentation, or class weighting can improve performance on underrepresented classes like "Disgust." Additionally, experimenting with advanced architectures like ResNet50 or EfficientNet could provide better feature extraction for overlapping classes. Including contextual features, such as head pose or environmental information, might help distinguish between subtle emotions. Hyperparameter optimization through grid search for learning rates, batch size, and dropout rates can further enhance performance. Finally, increasing the dataset size through transfer learning from similar datasets or generating synthetic samples for underrepresented classes can boost the model's ability to generalize.

Conclusion

This project comprehensively leverages the VGG16 architecture for facial expression recognition using the FER2013 dataset. By applying transfer learning from ImageNet, the training time is significantly reduced while maintaining strong baseline feature representations. Data augmentation—comprising flipping, rotation, zoom, and adjustments to brightness and contrast—expands the effective training set, aiding in better generalization. The two-phase training scheme, wherein the base layers are initially frozen and later unfrozen for fine-tuning at a reduced learning rate, successfully adapts the pre-trained features to the FER2013 domain without sacrificing the robustness gained from large-scale ImageNet training. Features such as dropout and L2 regularization further help control overfitting, allowing the model to learn useful patterns while maintaining reliable performance on unseen data.

Nevertheless, some limitations persist, including class imbalance and subtle misclassifications that may occur for certain expressions. Techniques such as class weighting, oversampling, or synthetic data generation could address the imbalance, while more advanced data augmentation and hyperparameter tuning may further refine model accuracy. Experimenting with newer architectures like ResNet50 or EfficientNet can also offer improvements in both speed and performance. Visualization tools—accuracy and loss curves, confusion matrices, and sample predictions—prove essential in diagnosing potential issues, informing model updates, and guiding future experimentation. Overall, this project highlights a scalable, reproducible framework for facial expression recognition, providing a solid foundation for continued enhancement through targeted solutions that address current challenges.