Final Project Report

Deep Learning

Code: 2598038

Group: 01

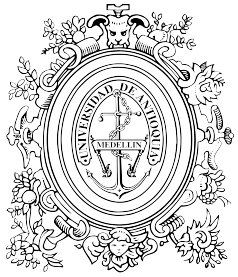
Date: 20/ 11/ 2024

By

Melissa Galeano Ruiz, CC: 1000416463

Professor

Raúl Ramos Pollán



UNIVERSIDAD DE ANTIOQUIA

1 8 0 3

Faculty of Engineering

Mechanical Engineering Department

University of Antioquia

Contents

1 Introduction

The notebook developed for this project is divided into four main sections:

1. Data Preprocessing
2. Convolutional Neural Network Models
3. Model Evaluation
4. Different Approach

Each section provides a detailed explanation of the analysis and procedures performed and the reasoning behind each decision, presented in both the report and the notebook. Additionally, the subsections within the model section cover aspects such as model selection, hyperparameter tuning, and performance metrics, ensuring a comprehensive understanding of the model development process.

1.1 Application context

In mechanical engineering, the integration of artificial intelligence and computer vision has the potential to revolutionize the approach to complex challenges. This project focuses on developing an emotion recognition model using convolutional neural networks to improve safety and efficiency in industrial environments through real-time implementation. By identifying emotional states such as happiness, sadness, anger, fear, disgust, neutrality, and surprise, the model provides valuable insights into workforce well-being. Recognizing emotions like surprise can signal emerging issues, enabling timely interventions and facilitating data-driven decisions on task assignments, workload distribution, and safety measures. Ultimately, this fosters a safer, more responsive workplace culture, enhancing productivity, satisfaction, and safety.

1.2 Dataset

The dataset used in this project is FER-2013 from *Kaggle* [1]. It is downloaded directly in the notebook with the command: `!kaggle datasets download -d msambare/fer2013 -p /content/dataset` and then unzipped. The data is split into training and test sets, each organized into folders by class. It contains 48x48 pixel grayscale images of faces, with 28,709 images in the training set and 3,589 in the test set, totaling 32,298 images in *JPG* format and occupying approximately 140 MB. Users can download the dataset as shown and proceed with preprocessing steps in the same notebook. For optimal performance, running the notebook on a GPU is recommended.

2 Methodology

The methodology was developed as follows:

2.1 Data Preprocessing

This process is crucial because it directly impacts the performance of the emotion recognition model. Each preprocessing step, as detailed in the subsections below, was carefully carried out to specifically address the characteristics of the ***FER-2013*** dataset and ensure it's well-prepared for efficient learning.

2.1.1 Data exploration

In the data exploration phase, the data was imported and divided into training and testing datasets. Upon graphing the distribution of classes, it became evident that the dataset was highly unbalanced. For instance, the "disgust" class contained only 436 images, while the "happy" class had 7,215 images. This imbalance presented a challenge in training the model, as it could lead to biased predictions towards the more frequent classes.

2.1.2 Data normalization

After exploring the dataset, the next step was to normalize the data. This process ensured that the pixel values of the images were scaled to a consistent range, between 0 and 1, by dividing each pixel or image by 255.

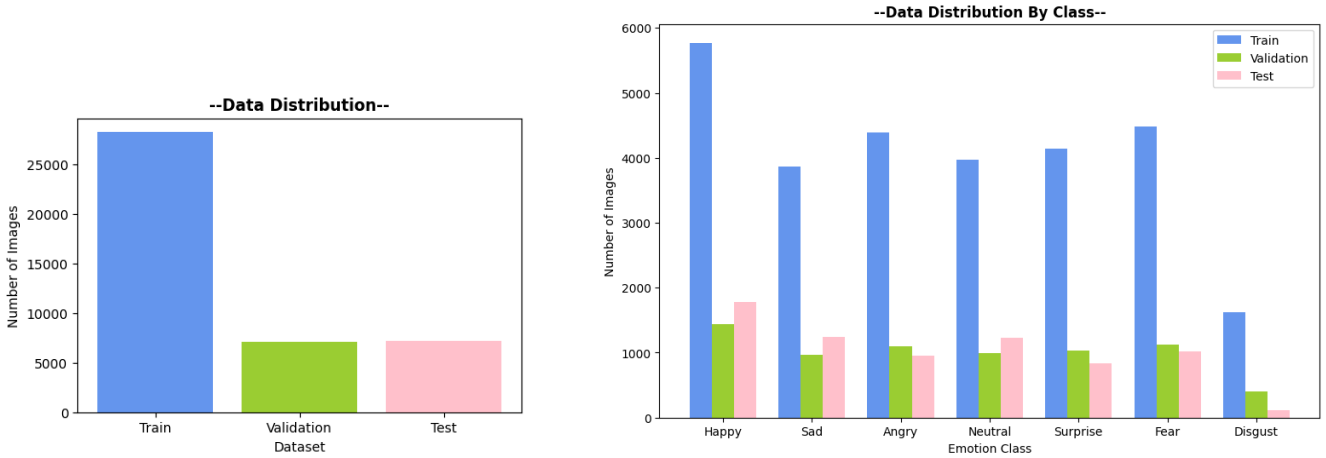
2.1.3 Data augmentation

Due to the significant class imbalance, data augmentation was applied to the classes with fewer samples, such as *disgust*, *surprise*, *angry*, and *fear*, in order to enhance their representation and reduce the overall data imbalance. Augmentation techniques such as rotation, horizontal flipping, zooming, brightness and contrast adjustments, cropping, and slight translations were applied to the images. The augmented sample sizes for these target emotions were as follows: 1,600 images for disgust, 2,000 for surprise, 1,500 for anger, and 1,500 for fear.

It is important to note that while the number of augmented images increased, it did not match the size of the majority class like *happy*. Generating a similar number could lead to redundancy, causing the model to overfit on repetitive patterns rather than learning generalizable features. The total number of data samples after augmentation increased from 28,709 to **35,309**.

2.1.4 Data shuffling and splitting

Since the data was initially sorted by class, shuffling was necessary to reduce bias and ensure proper model training and generalization. It is important to note that the test set was left unshuffled to maintain reproducibility in the final evaluation, allowing for accurate performance tracking and model comparison. Additionally, the *X_train* dataset was further divided into training and validation sets, with stratification applied to ensure that both sets maintained a similar class distribution. Figure 1 illustrates the overall data distribution as well as the class-wise distribution.



(a) General data distribution.

(b) Data distribution by class.

Figure 1: Data distribution.

2.2 Model Architectures

Before delving into the model architecture, it's important to note that several models were tested, and various hyperparameters were experimented with during the process. However, for clarity and ease of presentation, only the most relevant models and configurations are included in the notebook, ensuring better organization and visualization.

The architecture section is divided into two main parts. The first iteration consists of a simple CNN model with two convolutional layers, each having a kernel size of 3x3, followed by two max pooling layers and one dense layer with 128 neurons. This initial architecture served as a baseline to understand how basic CNN components interact for emotion recognition.

The second section, titled *Custom CNN Model Architectures*, introduces a function designed to streamline the process of testing different CNN architectures. This function was developed after experimenting with various architectures, enabling me to efficiently test and compare different models. It builds on a flexible yet optimized base structure, allowing easy modifications for diverse configurations. This section includes several more complex architectures that incorporate additional techniques such as **dropout** and **batch normalization**. Dropout was included to prevent overfitting by randomly setting a fraction of the input units to zero during training, while batch normalization was applied to normalize the activations of each layer, stabilizing the learning process and improving convergence.

In addition to architecture modifications, hyperparameters were fine-tuned to further enhance model performance. **Early stopping** was employed to halt training when the model stopped improving, effectively preventing overfitting. A patience value of 12 was set, and the option `restore_best_weights=True` was enabled to ensure that the model reverted to its best-performing state once training concluded.

Moreover, **ReduceLROnPlateau** was implemented to some of the models to dynamically adjust the learning rate during training. This method reduces the learning rate when the validation performance plateaus, helping the model converge more effectively by avoiding overshooting the optimal solution. This was necessary, as it was observed during testing that the model tended to overfit, which could hinder its ability to generalize well to unseen data. By lowering the learning

rate in response to stagnation, the model is better able to refine its weights, improving both training stability and generalization performance.

2.3 Iterations and Refinements

It is important to mention that different variations of the architecture were tested, including adjustments in the number of layers, kernel sizes and channels, dropout layers, and the inclusion or exclusion of batch normalization layers. These iterations were performed to assess which configuration would yield the best results in terms of model performance.

The number of layers and kernel sizes were modified in each iteration to explore the optimal configuration for the model. The input for all models consisted of 48x48x1 grayscale images, and the output layer used a softmax activation function to classify the data into seven emotion categories.

The models tested are referred to as Model B, Model C, Model D, Model E, and Model F, each representing a different configuration. For optimization, both the Adam optimizer and the Stochastic Gradient Descent (SGD) optimizer were used to train the models. Finally, for each of these models, accuracy and loss graphs were generated to visualize their performance and track improvements throughout the training process.

2.4 Different approach

2.4.1 Feature extraction + ML algorithms

In addition to the custom CNN models, an alternative approach was explored by utilizing the layers and weights of these models as feature extractors. The extracted features were then used as input to machine learning algorithms, such as **Support Vector Machine (SVM)** and **Random Forest**, to evaluate their performance in classifying the features. This approach aimed to leverage the power of traditional machine learning methods while benefiting from the feature extraction capabilities of CNN models.

2.5 Evaluation metrics

As mentioned in the previous report, all models were initially evaluated based on their accuracy. For the best-performing models, additional evaluation metrics, such as *precision*, *recall*, and the *F1-score*, were calculated to better assess the models' ability to distinguish between different emotions, as well as their confusion matrices.

3 Results

After training and testing the different models, their performance accuracy is shown in Figure 2.

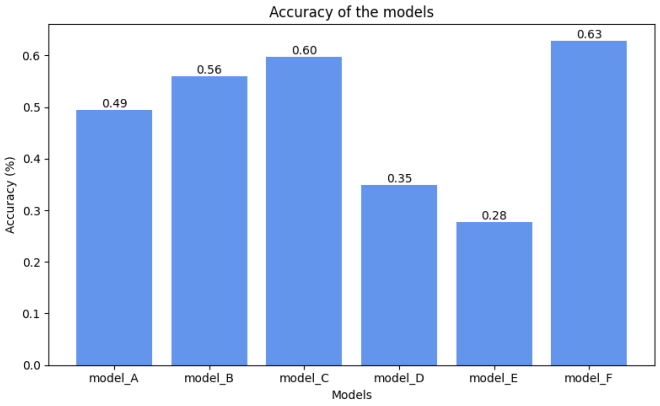


Figure 2: Evaluation of the different models.

As shown, the top-performing models are **Model F** and **Model C**, with accuracies of 63% and 60%, respectively. The architecture of the best model, **Model F**, is summarized in Table 1.

Table 1: Compact Summary of Model F Architecture

Layer	**Details**	**Activation/Other**
Input	(48, 48, 1)	-
Conv2D 1	64 filters, (3, 3), 'same'	BatchNorm, ReLU, MaxPool, Dropout (0.25)
Conv2D 2	128 filters, (5, 5), 'same'	BatchNorm, ReLU, MaxPool, Dropout (0.25)
Conv2D 3	512 filters, (3, 3), 'same'	BatchNorm, ReLU, MaxPool, Dropout (0.25)
Conv2D 4	512 filters, (3, 3), 'same'	BatchNorm, ReLU, MaxPool, Dropout (0.25)
Flatten	-	-
Dense 1	256 units	BatchNorm, ReLU, Dropout (0.25)
Dense 2	512 units	BatchNorm, ReLU, Dropout (0.25)
Output	num_classes	Softmax

3.1 Model Performance Metrics

The confusion matrix and performance metrics of the top two models were analyzed to gain deeper insights into their effectiveness. However, for clarity and relevance, only the best-performing model will be presented below in Table 2 and Figure 3.

Table 2: Performance Metrics for Model F

Class	Precision	Recall	F1-score	Support
Happy	0.81	0.84	0.82	1774
Sad	0.48	0.55	0.51	1247
Angry	0.59	0.52	0.56	958
Neutral	0.55	0.63	0.58	1233
Surprise	0.74	0.79	0.77	831
Fear	0.53	0.34	0.41	1024
Disgust	0.52	0.53	0.53	111
Accuracy		0.63		7178
Macro avg	0.60	0.60	0.60	7178
Weighted avg	0.63	0.63	0.62	7178

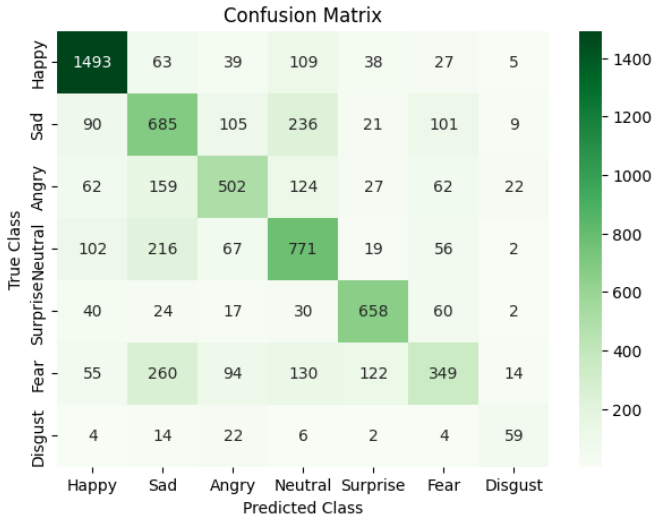


Figure 3: Confusion matrix of model F.

As shown, the confusion matrix and evaluation metrics highlight that the model struggles to accurately classify certain emotions. Notably, the F1-score, which combines precision and recall, is lowest for emotions like *fear*, *anger*, and *sadness*, suggesting these emotions are more challenging for the model to identifym as evidenced by the confusion matrix. It is important to note that this trend of misclassification is observed across all models tested.

3.2 Feature Extraction + ML Algorithms Evaluation

Applying the SVM model (`SVM_model = SVC(kernel='linear')`) and the Random Forest model (`RF_model = RandomForestClassifier(n_estimators=100)`) to the top three models (Model F, Model C, and Model B) resulted in a clear performance improvement, as shown in the table below.

As can be seen, implementing a CNN as a feature extractor and applying an ML algorithm can lead to better performance. When comparing the results, it is evident that the models show some improvement, indicating that the feature extraction process is effectively enhancing the learning process. Among the ML algorithms tested, Random Forest demonstrated the best performance,

Table 3: Performance of feature extractors with SVM and Random Forest

Feature Extractor	SVM Accuracy	RF Accuracy	Initial Accuracy
model_F	0.6110	0.6344	0.6293
model_C	0.5933	0.6183	0.5972
model_B	0.5644	0.5759	0.5602

outperforming SVM in terms of accuracy.

4 Discussion and Conclusions

From the previous results, Model F demonstrates clear proficiency in detecting happy expressions but struggles to accurately differentiate certain negative emotions. The challenges can be attributed to the following factors:

1. An imbalance in the training set, with a higher proportion of positive images compared to negative ones.

2. The more distinct and exaggerated features of "happy" and "surprised" expressions, in contrast to the more varied and subtle characteristics of negative emotions.

3. Some images and emotion labels in the dataset are ambiguous, even for humans, which makes it difficult for the model to learn and classify these images accurately. This happens especially between negative emotions.

4.1 Comparison with the literature

As shown in Table 4, the proposed model outperformed Ketan Sarvakar et al.'s CNN (55.61% accuracy), while FERNet (69.57%) and Fine-tuned VGGNet (73.28%) achieved higher accuracies. FERNet's success stemmed from optimized architecture, while VGGNet benefited from data augmentation, transfer learning, and advanced optimization. All models were tested on the FER2013 dataset, which several authors agree is one of the most challenging for emotion recognition, due to the heterogeneity of human faces, and significant variations in angles, lighting, and factors like age, gender, and ethnicity, increasing the complexity of facial expression recognition.

Model	Accuracy
CNN (K. Sarvakar)	55.61%
Model F + SVM	61.10%
Model F	62.93%
Model F + Random Forest	63.44%
FERNet	69.57%
Fine-tuned VGGNet	73.28%

Table 4: Model Performance Comparison

Despite the challenges posed by ambiguous images and emotion labels, which even humans find difficult to classify, the model demonstrates strong performance. Its ability to achieve good accuracy under such conditions underscores its effectiveness for engineering applications. This level of performance is particularly significant, as it highlights the model's potential utility in real-world engineering scenarios, where achieving perfect accuracy may not be critical.

References

[1] *FER-2013 | Kaggle.* (n.d.). Retrieved Sept. 16, 2024, from `https://www.kaggle.com/datasets/msambare/fer2013`

[2] K. Sarvakar *et al.*, "Facial emotion recognition using convolutional neural networks," *Materials Today: Proceedings*, vol. 80, pp. 3560–3564, Jan. 2023, doi: `https://doi.org/m79z`.

[3] J. D. Bodapati *et al.*, "FERNet: A Deep CNN Architecture for Facial Expression Recognition in the Wild," *J. Inst. Eng. India Ser. B*, vol. 103, no. 2, pp. 439–448, Apr. 2022, doi: `https://doi.org/m792`.

[4] Y. Khaireddin *et al.*, "Facial Emotion Recognition: State of the Art Performance on FER2013," May 08, 2021, arXiv:2105.03588, doi: `https://doi.org/m793`.