Spring 5-2024

Developing a Convolutional Neural Network (CNN) Model for Facial Expression Recognition (FER)

Danrlei Martins
CCT College Dublin

Leonardo Diesel
CCT College Dublin

Developing a Convolutional Neural Network (CNN) Model for Facial Expression Recognition (FER)

Danrlei Martins & Leonardo Diesel

A Report Submitted in Partial Fulfilment

of the requirements for the

Degree of

BSc in Computing in IT (4th year)

May 2024

Supervisor: Dr Muhammad Iqbal

Abstract

This Capstone Project focused on developing an accurate Facial Expression Recognition (FER) model by leveraging deep learning techniques, specifically Convolutional Neural Networks (CNNs). The objective was to explore, design, and implement custom architectures and evaluate their performance against existing work. The process involved several stages, such as data preprocessing, data augmentation, architecture design, hyperparameter tuning, and performance assessment using metrics like accuracy and F1-score while utilizing the FER-2013 dataset for training. The resulting FER model exhibited competitive accuracy levels and generalization capabilities, opening up opportunities for real-time implementation and application across various domains.

Table of Contents

Table of Figures

Terminologies & Definitions

FER = Facial Expression Recognition

FER-2013 = Facial Expression Recognition 2013 Dataset

CNN = Convolutional Neural Network

DCNN = Deep Convolutional Neural Network

CRISP-DM = Cross-Industry Standard Process for Data Mining

EDA = Exploratory Data Analysis

HPO = Hyperparameter Optimization

TP = True Positive

TN = True Negative

FP = False Positive

FN = False Negative

1. Introduction

1.1 Background

Facial expressions and body language are crucial to human communication, expressing various emotions and unspoken messages, which are fundamental to our daily social interactions and nonverbal aptitude (Barsoum et al., 2016). The ability to accurately recognize and interpret these expressions has numerous applications across multiple domains, including healthcare, education, customer behaviour analysis, and advertising (Vemou, Horvath and Zerdick, 2021).

FER refers to the process of automatically detecting and interpreting human facial expressions using computer algorithms and systems (Tian, Kanade and Cohn, 2011). It involves analyzing facial features such as movements of the eyebrows, eyes, nose, mouth, and overall facial muscle activity to perceive an individual's emotional state or intention (Bettadapura, 2012).

The rapid growth of computer vision techniques and deep learning algorithms has led to considerable FER and image classification breakthroughs in recent years (Bansal et al., 2021). Among the most prominent factors contributing to this boost are the appearance of large, high-quality, publicly available labelled datasets and the empowerment of parallel GPU computing, which enabled the transition from CPU-based to GPU-based training, thus allowing for significant acceleration in deep models' training (Voulodimos et al., 2018).

This capstone project aims to contribute to the field of FER by developing a reliable and accurate model. In order to implement a system that can identify and categorize facial expressions into seven different categories—angry, disgusted, fearful, happy, sad, surprised, and neutral—the project heavily focuses on deep learning techniques.

1.2 Project Concept

The concept of our project is to develop and evaluate a CNN model for FER. CNNs are a type of deep neural network architecture that is particularly effective for processing data with a grid-like topology, such as images, speech, or video (Goodfellow, Bengio and Courville, 2016).

By methodically preparing and preprocessing the dataset, exploring different model architectures, and fine-tuning hyperparameters, we aim to develop a model that can compete with or ideally outperform existing solutions regarding accuracy and generalization capabilities.

1.3 Roles & Responsibilities

The responsibilities were divided across the CRISP-DM phases. However, extensive collaboration and knowledge sharing happened throughout the project's lifecycle.

Danrlei Martins was primarily responsible for the initial phases, including Business Understanding (Phase 1), Data Understanding (Phase 2), and Data Preparation (Phase 3). Responsibilities included defining the project objectives, exploring the dataset, performing EDA, and implementing data preprocessing techniques such as augmentation.

Leonardo Diesel took the lead in the final phases, including Modelling (Phase 4), Evaluation (Phase 5), and Deployment (Phase 6). He started exploring and implementing different model architectures, conducted hyperparameter tuning, evaluated the model's performance using appropriate metrics, and deployed the trained model in a real-time facial expression recognition system.

Despite the division of responsibilities, both team members actively contributed to research, experimentation, and decision-making processes across all aspects of the project. Regular meetings, discussions, and collaborative coding sessions ensured effective knowledge transfer and a joint approach to overcoming challenges and achieving the project's objectives.

2. Methodology

We followed the CRISP-DM framework to ensure a structured and systematic approach to our FER project. CRISP-DM is a well-known framework for planning, creating, and implementing predictive machine learning models, and it provides businesses with the structure they need to get better and faster results (Singh and Joshi, 2022; Shearer, 2000). As illustrated in Figure 1, the CRISP-DM framework covers six phases (Chapman et al., 2000).

Here is an overview of what each phase consists of in the context of this capstone project:

1. **Business Understanding (Phase 1)**: We determined our project's criteria and business goals at this first stage. We explored possible use cases for a reliable and accurate FER system. We also established the success criteria and the metrics that will be used to assess the project's business results.
2. **Data Understanding (Phase 2)**: We carefully reviewed the FER2013 dataset, which includes labelled photos of faces with various expressions. This step identified potential problems or obstacles related to the dataset, along with EDA and quality evaluation.

We learned more about the dataset's properties, including expression class distribution, image resolutions, and any imbalances or discrepancies.

3. **Data Preparation (Phase 3)**: We performed the necessary data preprocessing steps based on the data understanding phase findings. Specifically, we applied data augmentation techniques to our image data. Finally, we split the dataset into training, validation, and testing sets to ensure proper model evaluation and prevent overfitting.
4. **Modelling (Phase 4)**: We explored various deep-learning architectures and techniques suitable for the FER task. Our primary focus was on pre-trained models and implementing hyperparameter tuning to optimize the performance of the custom model.
5. **Evaluation (Phase 5)**: We evaluated the trained models using appropriate performance metrics, such as accuracy, precision, recall, and F1-score. Additionally, we generated visualizations like confusion matrices to gain insights into the models' strengths and weaknesses in classifying different expressions.
6. **Deployment (Phase 6)**: We deployed the FER model as a real-time system using OpenCV, which aimed to detect live emotions through the webcam.

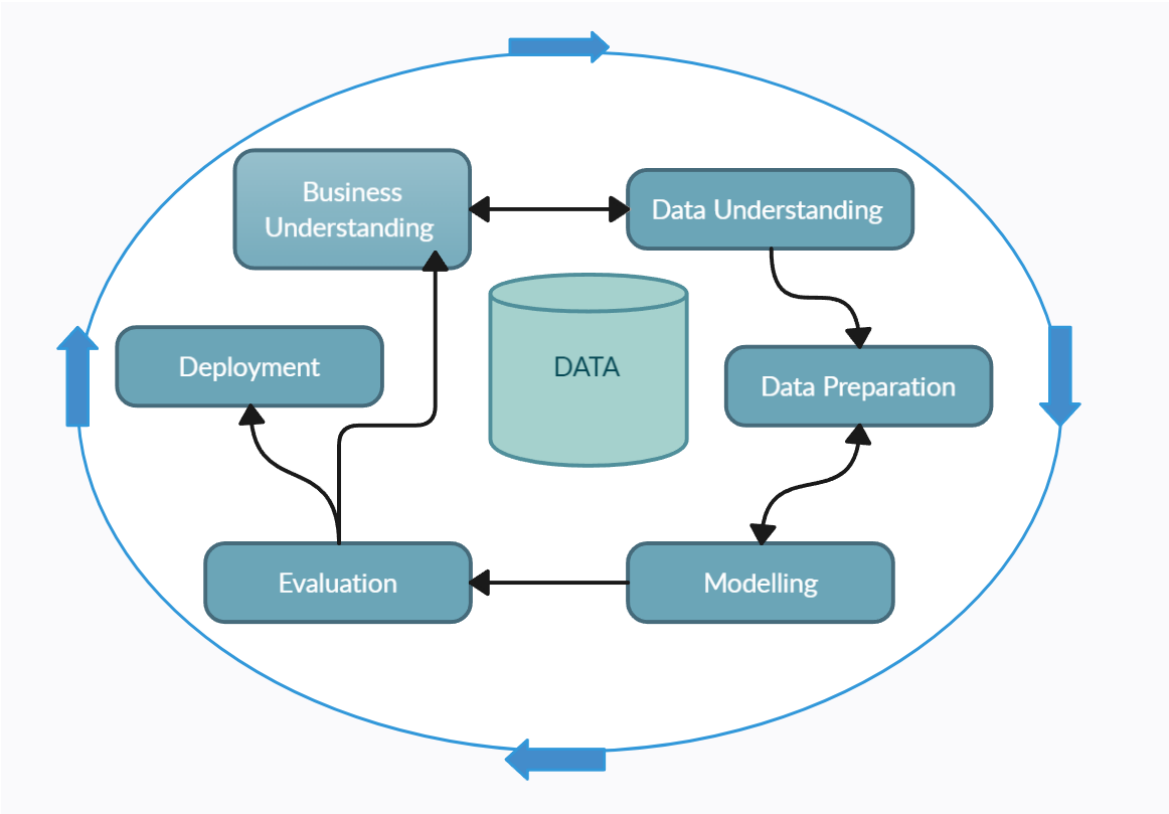

Figure 1 - CRISP-DM Framework Lifecycle

3. Business Understanding (CRISP-DM Phase 1)

The FER technology industry is expanding and has a lot of promise for companies in various sectors. According to Fortune Business Insights (2023), the market value is estimated to reach USD 74.80 billion by 2029 from an earlier value of USD 26.25 billion in 2022. Companies across sectors are investing in FER due to its diverse applications and potential for human-machine interaction.

In recent years, significant research has been done in FER, leveraging deep learning models to improve emotion analysis and prediction capabilities (Pise et al., 2022). The potential use cases continue to expand as it becomes more sophisticated and accurate. From enhancing customer service and user experience by measuring satisfaction and engagement levels to improving mental health support by detecting emotional distress to advancing human-robot interactions and emotional AI assistants – real-world applications of FER are vast (Samadiani et al., 2019).

3.1 Problem Definition

This project aims to develop an accurate FER model using deep learning techniques, specifically CNNs. FER is a challenging problem in computer vision and has numerous applications across various domains. However, existing models may still have room for improvement in accuracy and generalization capabilities.

3.2 Business Objectives

The primary objective is to train a CNN model for FER that can accurately recognize different facial expressions. The specific objectives are:

1. Explore and implement a custom CNN architecture for FER.
2. Evaluate the trained model's performance on the FER-2013 dataset.
3. Compare the developed model's accuracy with existing work and identify areas for potential improvement.
4. Deploy the trained model for real-time facial expression recognition on live webcam feeds.

3.3 Business Success Criteria

The following criteria will measure the success of this project:

- The trained CNN model achieves competitive or superior accuracy compared to existing work on the FER-2013 dataset.

- The model demonstrates good generalization capabilities across diverse facial expressions and demographics.
- The deployed model can accurately recognize facial expressions in real-time on live webcam feeds.
- The developed model can be further fine-tuned or integrated into broader applications or systems involving FER.

We aim to contribute to FER technology by developing an accurate and robust CNN model, which can be a foundation for future research or practical applications in this field, including real-time facial expression recognition.

4. Data Understanding (CRISP-DM Phase 2)

4.1 Data Collection

We trained our machine learning models using the FER-2013 dataset. The dataset was developed by Pierre Luc Carrier and Aaron Courville and was introduced at the International Conference on Machine Learning (ICML) in 2013 (Goodfellow et al., 2013).

The facial images were retrieved using Google Image Search API, corresponding to different emotion keywords. These keywords were combined with words about age, gender, or ethnicity to create around 600 strings utilized as search terms for facial images. OpenCV face recognition technology was used to create squared boxes surrounding each face in the gathered photos. Next, humans filtered out duplicate images, rejected improperly identified images, and adjusted cropping as needed (Goodfellow et al., 2013).

The final dataset contains 35,887 images of facial expressions, each labelled with one of seven emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral.

4.2 Exploratory Data Analysis (EDA)

We began by examining the distribution of expression categories in both the training and testing sets. The code revealed an imbalance in the number of images across different expression classes. For instance, in the training set, the 'happy' class had the highest number of images (7,215), while the 'disgust' class had the lowest (436). This imbalance could introduce bias during model training and affect the overall performance.

To visualize the distribution, we plotted bar charts representing the number of images per expression category for the training and testing sets. As seen in Figures 2 and 3, these charts

clearly illustrate the class imbalance, highlighting the need for potential data augmentation or resampling techniques during the data preparation phase.

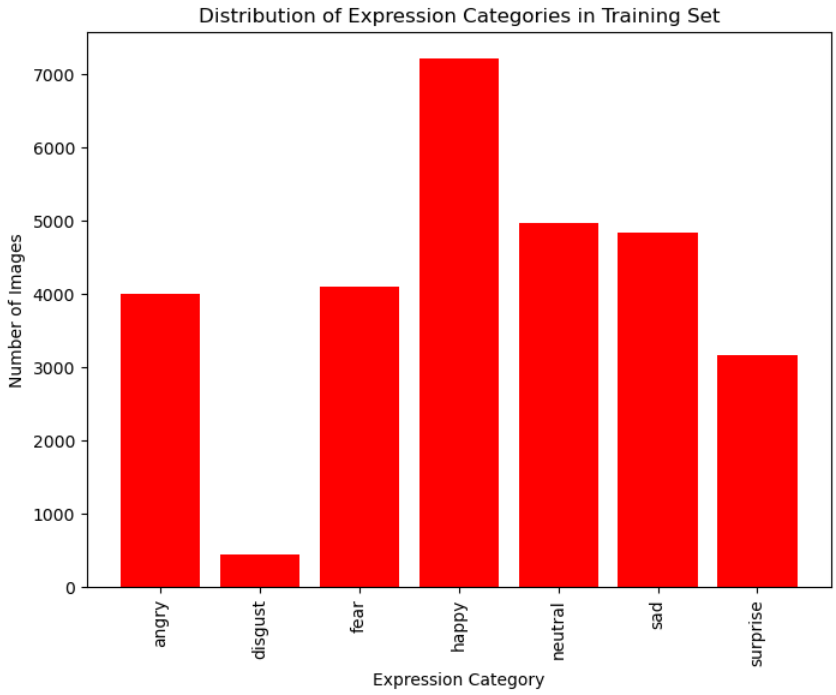

Figure 2 – Distribution of expression categories in the FER2013 training set

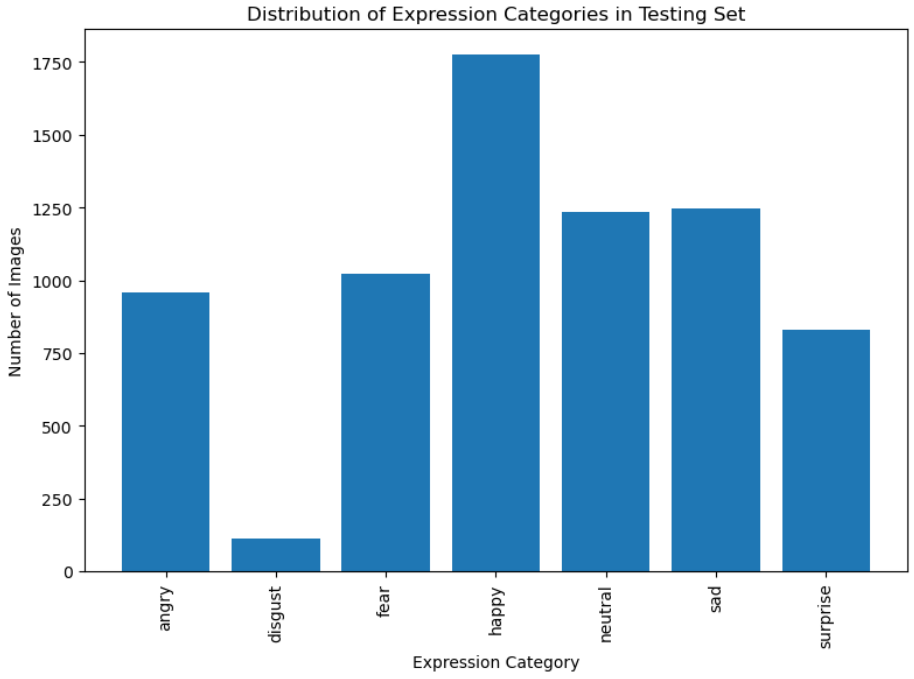

Figure 3 - Distribution of expression categories in the FER2013 testing set

Furthermore, we implemented a function to visualize a random sample of images from each expression category in the training set. As seen in Figure 4, this visual inspection allowed us to assess the quality and diversity of the data and identify any potential issues or inconsistencies within the dataset.

Figure 4 – FER-2013 Sample of Images

We observed that some images were not accurately classified into correct emotion categories. Certain instances revealed images categorized as 'happy' with neutral or non-smiling facial expressions. In contrast, others displayed pictures assigned to the 'fear' category that did not exhibit characteristics typically associated with that emotion.

These misclassifications could impact the model's overall accuracy when deployed, highlighting the need for further improvement and optimization of the classification algorithms or the training data. In summary, the key findings were:

- **Class Imbalance:** The dataset exhibited a significant imbalance in the number of images across different expression categories, with some classes being heavily underrepresented compared to others.
- **Distribution Consistency:** Despite the imbalance, the distribution of expression categories was consistent between the training and testing sets, ensuring that the model's performance could be evaluated on a representative test set.
- **Image Quality:** A visual inspection of the sample images revealed that the dataset contained images of varying quality, resolutions, and lighting conditions, which could pose challenges during model training and inference.

5. Data Preparation (CRISP-DM Phase 3)

We employed data augmentation techniques to address the class imbalance observed in the Data Understanding phase and enhance the model's performance. Data augmentation is a widely used approach in deep learning models, particularly for image data, as it helps increase the training dataset's diversity and size, reduce overfitting, and improve the model's generalization capabilities (Shorten and Khoshgoftaar, 2019).

5.1 Data Augmentation

We utilized the ImageDataGenerator class from the Keras library to apply various augmentation techniques to the training data (TensorFlow, 2024). Using augmentation techniques, the ImageDataGenerator class generates new variations of the original images, effectively creating a more extensive and diverse dataset (Rosebrock, 2019).

Table 1 lists the augmentation operations applied to the dataset, and Figure 5 illustrates the overall augmentation process. To visualize the effects of data augmentation, we implemented a function that displays a random sample image from each expression category alongside five augmented versions of the same image.

As seen in Figure 6, This visual inspection allowed us to ensure that the augmentation techniques were applied correctly and introduced the desired variations without compromising the integrity of the facial expressions.

Operation	Description	Parameters/Range
Rescaling	Normalizes pixel values to a range between 0 and 1.	Dividing by 255
Rotation	Randomly rotates images to introduce orientation variations.	Up to 15°
Zoom	Applies random zoom to simulate varying distances and scales.	Zoom range of 0.1
Horizontal Flipping	Randomly flips images horizontally to account for left-right variations.	Random
Height & Width Shifting	Randomly shifts the image along the height and width to introduce positional variations.	10%
Fill Mode	Fills in empty areas created by transformations.	'nearest'

Table 1 - Data Augmentation Operations Summary

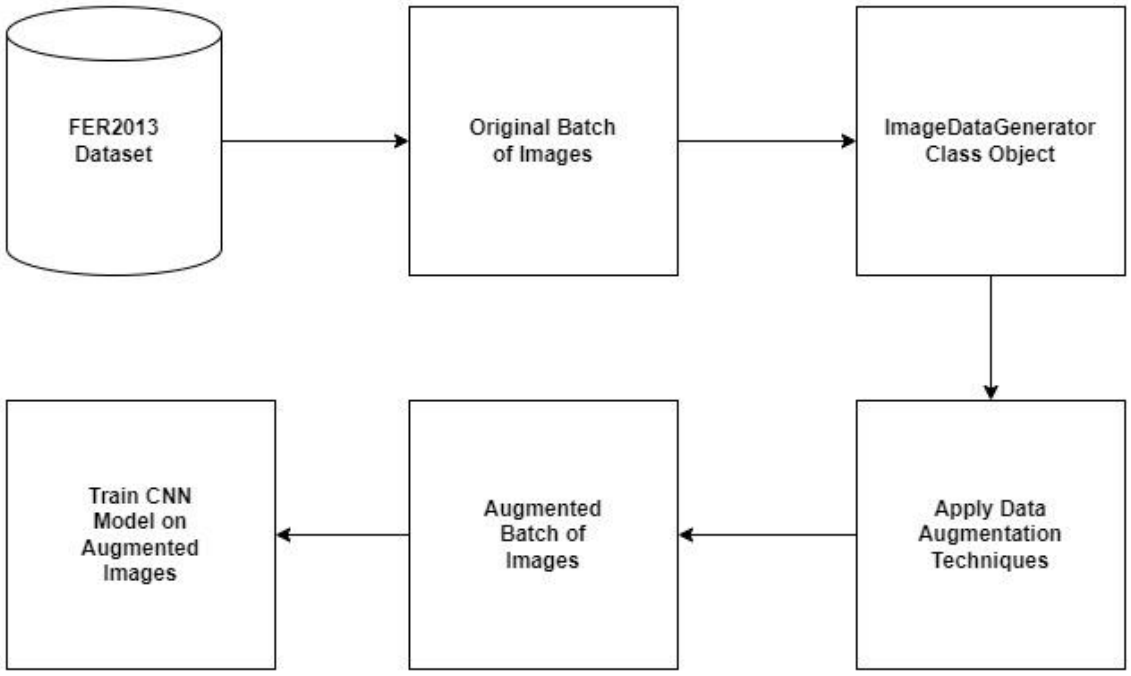

Figure 5 – Data Augmentation Process Diagram

Figure 6 – Data Augmentation Results

5.2 Preparing Training, Validation & Testing Sets

After applying data augmentation to the training set, we utilized the flow from directory function from the Keras library to load and preprocess the training, validation, and testing sets. This function automatically splits the data into batches, applies the specified augmentation techniques (for the training set), and performs one-hot encoding of the labels. (TensorFlow, 2024)

The data loading function also allowed us to specify several important parameters for efficient data loading and preprocessing:

- **Target Size**: All images were resized to a consistent target size of 48x48 pixels, matching our input data.
- **Batch Size**: We set a batch size of 128 images, determining the number of images loaded and preprocessed simultaneously from the directory.
- **Shuffle**: The training and testing sets were shuffled to ensure randomization and prevent any potential biases introduced by the order of the data.
- **Class Mode**: The labels were set to be one-hot encoded, representing the seven expression categories as binary vectors.

The training set was split into a training subset (80%) and a validation subset (20%), ensuring the model's performance could be evaluated on unseen data during the training process. This validation set acted as a crucial checkpoint, allowing us to monitor the model's performance and detect potential overfitting or underfitting issues.

As for the testing set, only the rescaling operation was applied to ensure consistent preprocessing with the training and validation sets. The testing set remained separate and untouched by any augmentation techniques, providing an unbiased evaluation of the model's generalization capabilities on unseen data.

The distribution of images across the seven expression categories (angry, disgust, fear, happy, neutral, sad, surprise) in each set was as follows:

- **Training Set**: 22,968 images
- **Validation Set**: 5,741 images
- **Testing Set**: 7,178 images

We ensured the training process had access to a diverse and balanced dataset while maintaining a separate, untouched testing set for reliable performance evaluation. These are the fundamental rationale for this phase:

- **Increased Data Diversity:** The data augmentation techniques introduced variations in orientation, scale, and position, effectively increasing the diversity of the training dataset.
- **Balanced Class Distributions:** We attempted to mitigate the class imbalance issue observed in the Data Understanding phase by applying augmentation to the classes.

- **Efficient Data Loading:** The image data generators allowed for efficient loading and preprocessing of the data, reducing memory consumption and enabling the training process to handle larger datasets.

The augmented and preprocessed datasets were then used in the subsequent modelling phase, where we trained and evaluated various deep-learning architectures.

6. Modelling (CRISP-DM Phase 4)

Building a model is a complex process that requires extensive research to identify the appropriate library and fine-tune the specific parameters to develop the most accurate solution. After exploring numerous libraries and testing various models, we determined that the pre-trained model, VGG16 from the VGGNet family, was suitable for our project's requirements.

6.1 Model Architecture

We chose to work with the VGG16 architecture, a popular DCNN model. It was developed by the Visual Geometry Group (VGG) at the University of Oxford and presented at the International Conference on Learning Representations (ICLR) 2015 conference (Simonyan and Zisserman, 2015).

The model was pre-trained on the ImageNet dataset, a large-scale database of annotated images, allowing it to learn rich visual representations that could be transferred and fine-tuned in different domains (Russakovsky et al., 2015). VGG16 earns its name from its architecture, consisting of 16 layers, including convolutional, max-pooling and fully connected layers. By increasing the depth of the network, the researchers aimed to achieve significant improvements in image classification tasks.

The VGG16 architecture, as illustrated in Figure 7, follows a simple and uniform design. Similar to many other architectures, it extracts features from input images. However, it employs small 3x3 filters and max-pooling to downsample the feature maps, contributing to its excellent performance in various computer vision tasks, such as image classification, segmentation, and object detection (Sunyoto et al., 2022; Simonyan and Zisserman, 2015).

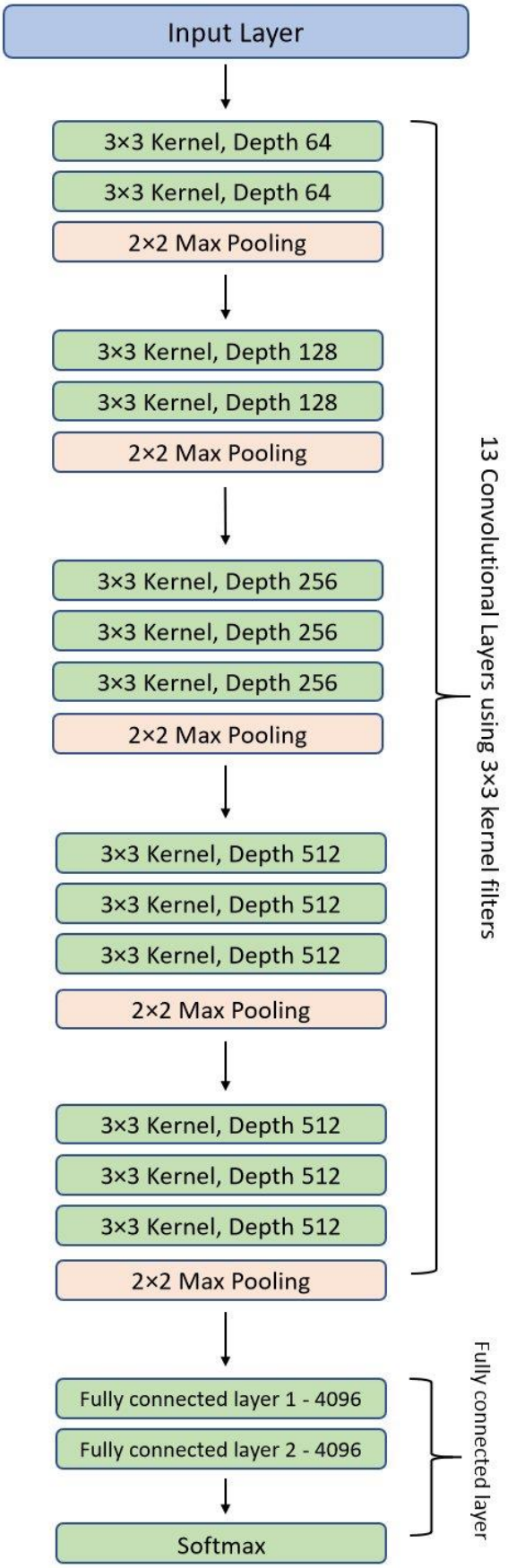

Figure 7 - VGG-16 Model Architecture (Tammina, 2019)

The initial tests with the architecture on default parameters showed promising results, achieving 61.02% validation accuracy. This performance was notably better than the previously tested models, providing a solid foundation for further model tuning and fine adjustments.

6.2 Transfer Learning & Fine-Tuning

We used a transfer learning technique to benefit from the knowledge of the pre-trained VGG16 model and modified it accordingly for our FER task. The process of applying previously learned model knowledge to a new task is known as transfer learning (Hosna et al., 2022; Tammina, 2019).

In our project's context, transfer learning involved using the VGG16's knowledge and fine-tuning it on the FER-2013 dataset. Specifically, we used the pre-trained model as a base and removed the top (classification) layer. We then added a new output, a fully connected layer with seven units corresponding to the seven expression categories.

Additionally, we employed the following callback techniques to mitigate overfitting and improve the model's generalization capabilities.

- **ReduceLROnPlateau**: It can be thought of as a learning rate scheduler. It monitors and helps the model to reduce the learning rate when the training stagnates (Keras, 2024b).
- **EarlyStopping**: This method monitors the training, and as soon as a target metric no longer improves, it stops the training (Keras, 2024a).

Finally, we trained the model, allowing it to learn the specific representations of our facial expression data.

6.3 Hyperparameter Optimization (HPO)

Training deep learning models such as CNN on image datasets is often computationally intensive (Forruque Ahmed et al., 2023), and finding the optimal hyperparameters through manual trial-and-error is considered impractical. HPO techniques can automate this process, reducing the human effort required while improving the models' performance by systematically searching for the best hyperparameter configurations.

To address this challenge, we utilized the Keras Tuner library, which provides a convenient and structured interface for performing hyperparameter tuning (O'Malley et al., 2019).

According to O'Malley (2020), the library allows for the automated exploration of various hyperparameter configurations, facilitating the identification of optimal settings that return the best model performance.

As for the search strategy, we used the built-in RandomSearch, which randomly samples hyperparameter configurations from the defined search space (Keras, 2019). The tuner evaluated multiple trial models with a different hyperparameter configuration and selected the configuration that maximized the validation accuracy.

The Keras Tuner function was designed to construct the model architecture based on the sampled hyperparameters and the base model of VGG16. We defined a comprehensive hyperparameter search space that included the following:

Hyperparameter	Search Space	Best Value Found
Dense Layer	2, 3	2
Units per Dense Layer	256, 512, 1024, 2048, 4096	1024
Dropout Rate	0.1, 0.2, 0.3, 0.4, 0.5	0.4, 0.3
Optimizer	‘Adam’, ‘sgd	‘sgd’
Learning Rate	0.001, 0.0001	0.0001
Batch Size	128, 256	256
Activation Function	‘relu’, ‘sigmoid’	‘relu’

Table 2 - Summary of Hyperparameter Tuning Search

6.4 Challenges Faced & Findings

The modelling phase involved an iterative process of experimenting with different architectures, transfer learning strategies and hyperparameter configurations.

However, we encountered several challenges while fine-tuning the base model. We started by freezing pre-trained layers, which we initially thought could help increase the accuracy as it would keep the pre-trained model weights. However, after multiple tries on our base model, there was no increase in accuracy, leading us to abandon this approach.

Another technique we explored was using Global Average Pooling (GAP) instead of flattening on the output layer. GAP computes the average value of all elements in the feature map, significantly reducing the number of parameters (Lin, Chen and Yan, 2014). This technique inherently reduces overfitting, an issue observed while testing models, where the training accuracy was usually notably higher than the validation accuracy. Later, we understood that it

did not resolve the overfitting problem, but as it reduced the number of parameters, we kept it in our model for quicker executions.

Including batch normalization layers after each dense layer was theoretically beneficial to improve our custom model's performance, but once more, it did not work as expected. In our understanding, this technique would help to decrease the overall loss, increasing the model's accuracy, which did not happen. The results were unsatisfactory when tested, and the idea was not considered.

During our research to improve accuracy, we reviewed our data preparation and discovered the Synthetic Minority Oversampling Technique (SMOTE). This method deals with imbalanced data by generating new images based on existing ones for better learning and validation (Chawla et al., 2002). However, after implementation, the results were unsatisfactory, and we dropped this idea due to the lack of accuracy improvement and continuous overfitting.

As previously mentioned, we chose a combination of Keras Tuner and RandomSearch for model tuning due to computational power limitations. Our initial plan was to use Grid Search. Still, after a test run, we realized that executing the desired search would take more than a year due to hyperparameter complexity.

Therefore, we defined a comprehensive hyperparameter search space for Keras Tuner and tested multiple executions on personal computers and through the Google Colab Platform, which helped with available computing power. However, it is essential to note that the best hyperparameters found through this search may not be optimal within the possible search space as it is a randomized search.

Finally, we combined the structure of found hyperparameters with the knowledge gathered throughout the project. In this project, we discovered that fine-tuning a pre-trained model combines applying existing knowledge and trial and error through multiple tests.

6.5 Proposed Custom CNN Model

We developed a custom model architecture to improve the default model's performance. The custom model was built upon the pre-trained VGG16 model as the base, but we made several modifications to the architecture. Firstly, we applied GAP to the output of the VGG16 base model, which helped reduce the number of parameters and prevent overfitting.

Additionally, we added two dense layers with 4096 and 1024 units, respectively, along with ReLU activation functions and dropout layers to introduce non-linearity and regularization.

The final layer was dense with seven units and a softmax activation function to perform the multi-class classification task.

Based on our research and understanding of our limitations, we were satisfied with our findings, which aligned with existing ones. Nevertheless, we believe we could achieve higher accuracy with more time and resources.

7. Evaluation (CRISP-DM Phase 5)

In the evaluation phase, we assessed the performance of our trained models using various metrics to gain insights into their strengths and weaknesses. The primary objective was to evaluate the model's ability to accurately classify facial expressions across the seven categories present in the dataset. As we dealt with a multiclass classification task on an imbalanced dataset, we carefully selected the evaluation metrics to ensure a comprehensive and unbiased assessment.

7.1 Evaluation Metrics & Scoring

We utilized confusion matrices and classification reports to evaluate the models' performance. These techniques helped us spot any possible problems and assess the true predictive capability of each model.

A confusion matrix is a table that summarizes the performance of a classification model by comparing the predicted classes with the actual (true) classes. It shows the number of correctly classified and misclassified instances for each class (Grandini, Bagli and Visani, 2020). As seen in Figure 8, predictions are broken down into four primary categories.

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Figure 8 - Confusion Matrix Labels (Draelos, MD, PhD, 2019)

We focused on the metrics seen in Table 3, which individually considered each class's metrics to address the class imbalance challenge (Müller and Guido, 2017; Kirk, 2017).

Metric	Definition	Equation
Accuracy	The proportion of correctly classified instances out of the total cases.	$\frac{TP + TN}{TP + TN + FP + FN}$
Precision	The proportion of positively predicted genuinely positive samples.	$\frac{TP}{TP + FP}$
Recall	Measures how many positive samples the positive predictions capture.	$\frac{TP}{TP + FN}$
F1-Score	The harmonic mean of precision and recall provides a balanced evaluation of both metrics.	$2 * (\frac{precision * recall}{precision + recall})$

Table 3 – Multi-Classification Model Metrics

7.2 Performance Evaluation Results

7.2.1 Default Model (VGG16)

The starting point was the pre-trained VGG16 model, which was used as a baseline model. The default model achieved a training accuracy of 66.54% and a validation accuracy of 61.02% on our dataset. While these results were reasonable, there was room for improvement.

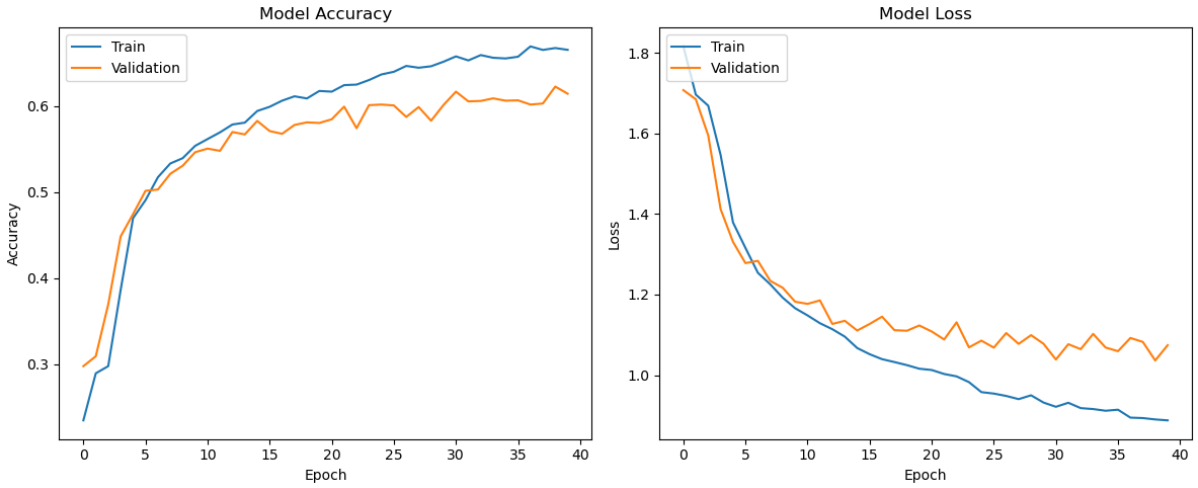

Figure 9 - Default Model Accuracy & Loss Plot

Classification Report:

	precision	recall	f1-score	support
Angry	0.40	0.50	0.44	799
Disgust	0.00	0.00	0.00	87
Fear	0.36	0.23	0.29	819
Happy	0.70	0.87	0.78	1443
Sad	0.51	0.57	0.54	993
Surprise	0.42	0.32	0.36	966
Neutral	0.69	0.64	0.66	634
accuracy			0.55	5741
macro avg	0.44	0.45	0.44	5741
weighted avg	0.52	0.55	0.53	5741

Figure 10 - Baseline Model Classification Report Results

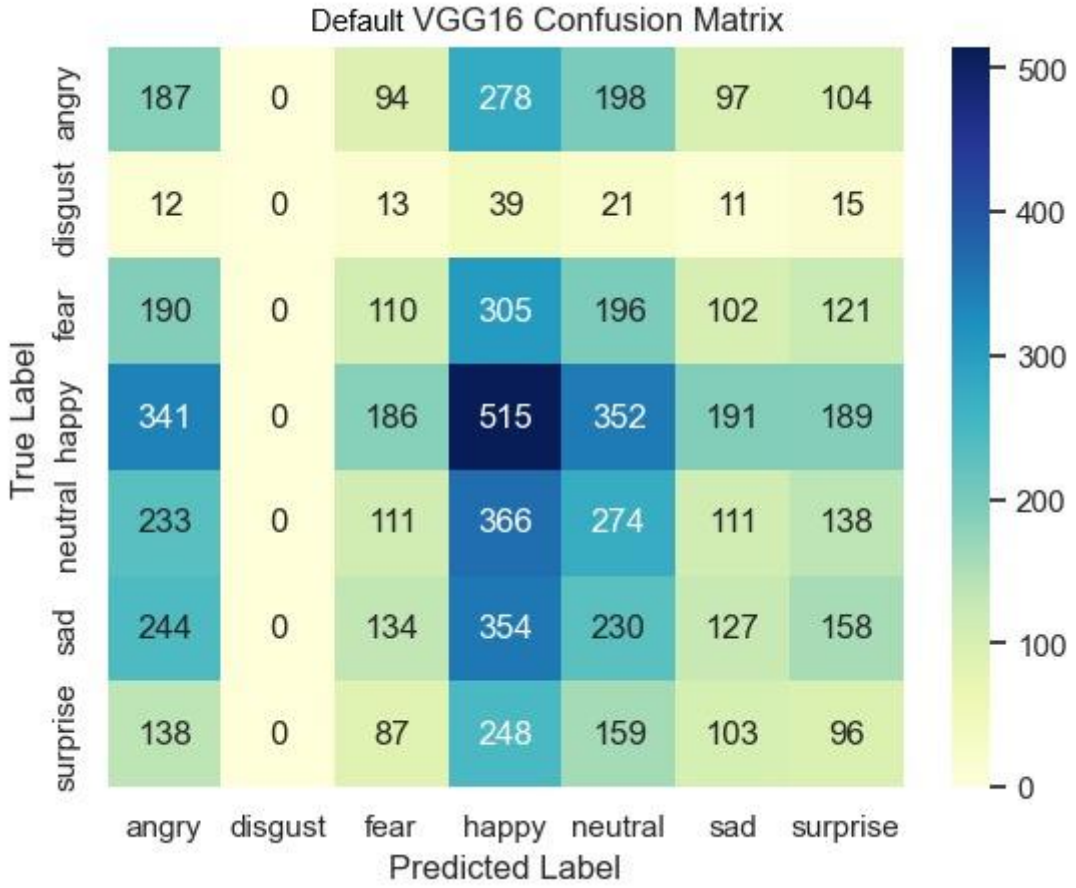

Figure 11 - Default Model Confusion Matrix

7.2.2 Final Custom Model

The custom model achieved significantly better results compared to the default model. It achieved a training accuracy of 95.37% and a validation accuracy of 66.57%, demonstrating

its improved ability to generalize to unseen data. On the test set, the custom model achieved an accuracy of 67.11%, outperforming the default model's test accuracy of 61.94%.

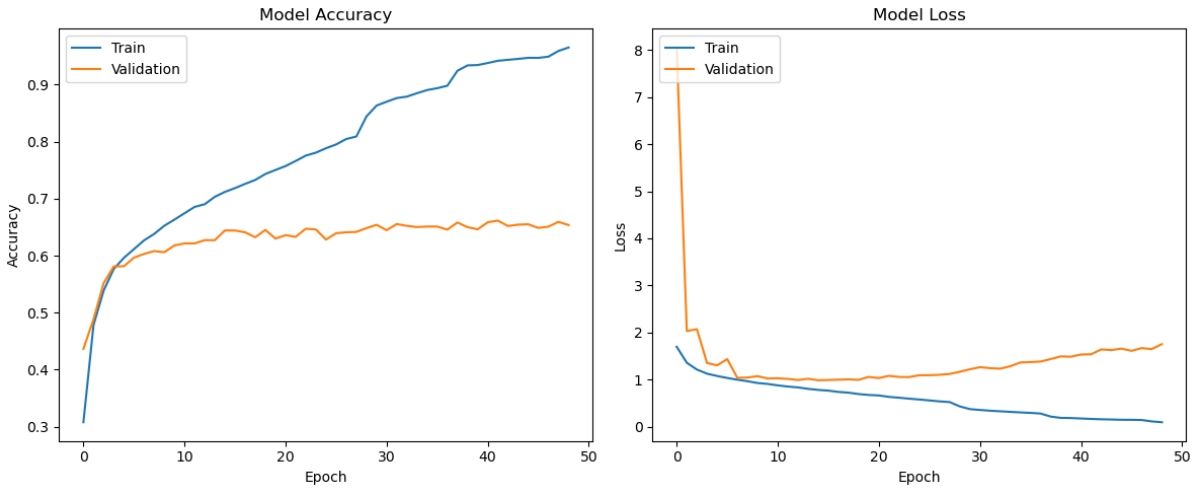

Figure 12 - Custom Model Accuracy & Loss Plot

Classification Report:

	precision	recall	f1-score	support
Angry	0.58	0.57	0.57	799
Disgust	0.76	0.64	0.70	87
Fear	0.56	0.48	0.52	819
Happy	0.86	0.83	0.85	1443
Sad	0.58	0.66	0.62	993
Surprise	0.53	0.55	0.54	966
Neutral	0.79	0.81	0.80	634
accuracy			0.66	5741
macro avg	0.67	0.65	0.66	5741
weighted avg	0.67	0.66	0.66	5741

Figure 13 - Custom Model Classification Report

As seen in Figure 13, the classification report for the custom model showed improved precision, recall, and F1-scores across most emotion classes, particularly for the "Disgust" and "Neutral" classes, which were challenging for the default model.

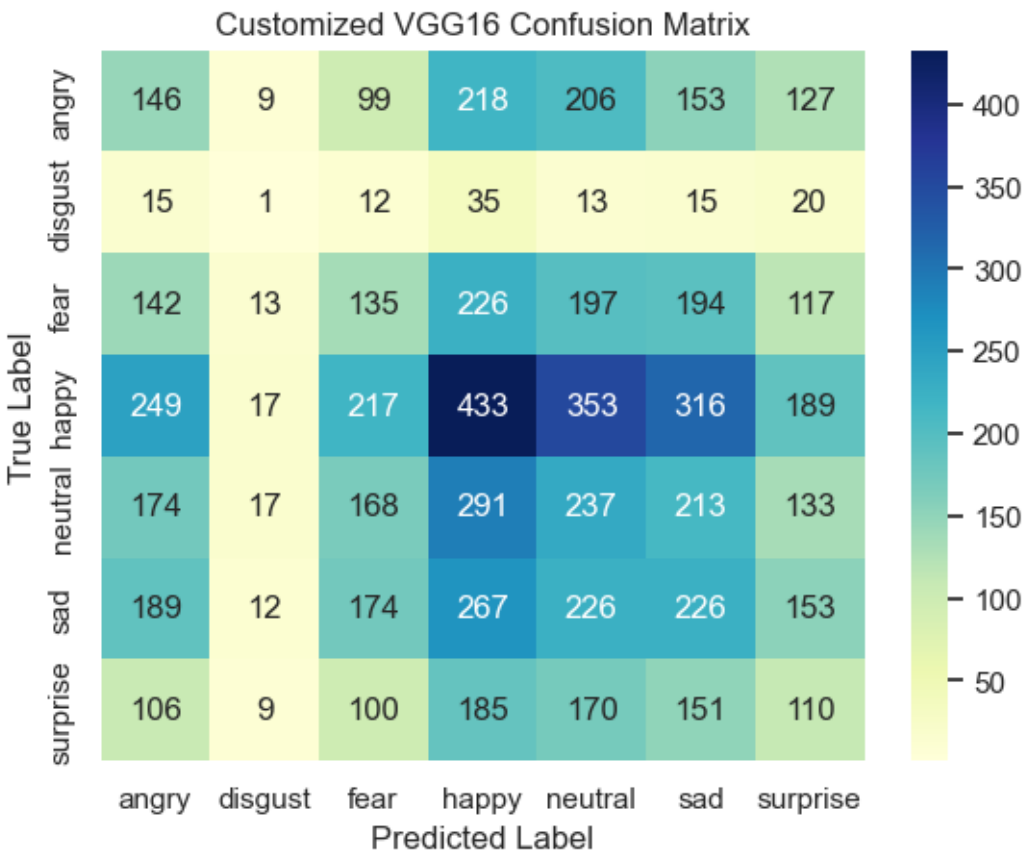

Figure 14 - Custom Model Confusion Matrix

7.3 Other Evaluation Visualizations

To conceptualize the model's predictions, having a visualization of how it works and not base our understanding only on the metrics, we decided to plot a few images, checking when the model predicted correctly according to the accurate labels or not. Figure 15 displays some randomized samples of when the model made correct emotion predictions.

Figure 15 - Correctly Classified Images

Also, we displayed some randomized samples again. Still, now, when the model made the wrong prediction according to the true labels, We were able to spot that the proper labels are not always correct, which can be a personal perspective, and in this case, the prediction could be accurate.

Figure 16 - Misclassified Images

The third visualization generated is a technique to understand better how the model works on the inside, a snapshot of the black box. This technique is called a saliency map, displayed as a heat map, highlighting the most critical pixels for the classification algorithm (Alqaraawi et al., 2020). Figure 17 shows that the model successfully highlights brighter parts directly over the face, leaving aside the parts the model considers unimportant for the FER task.

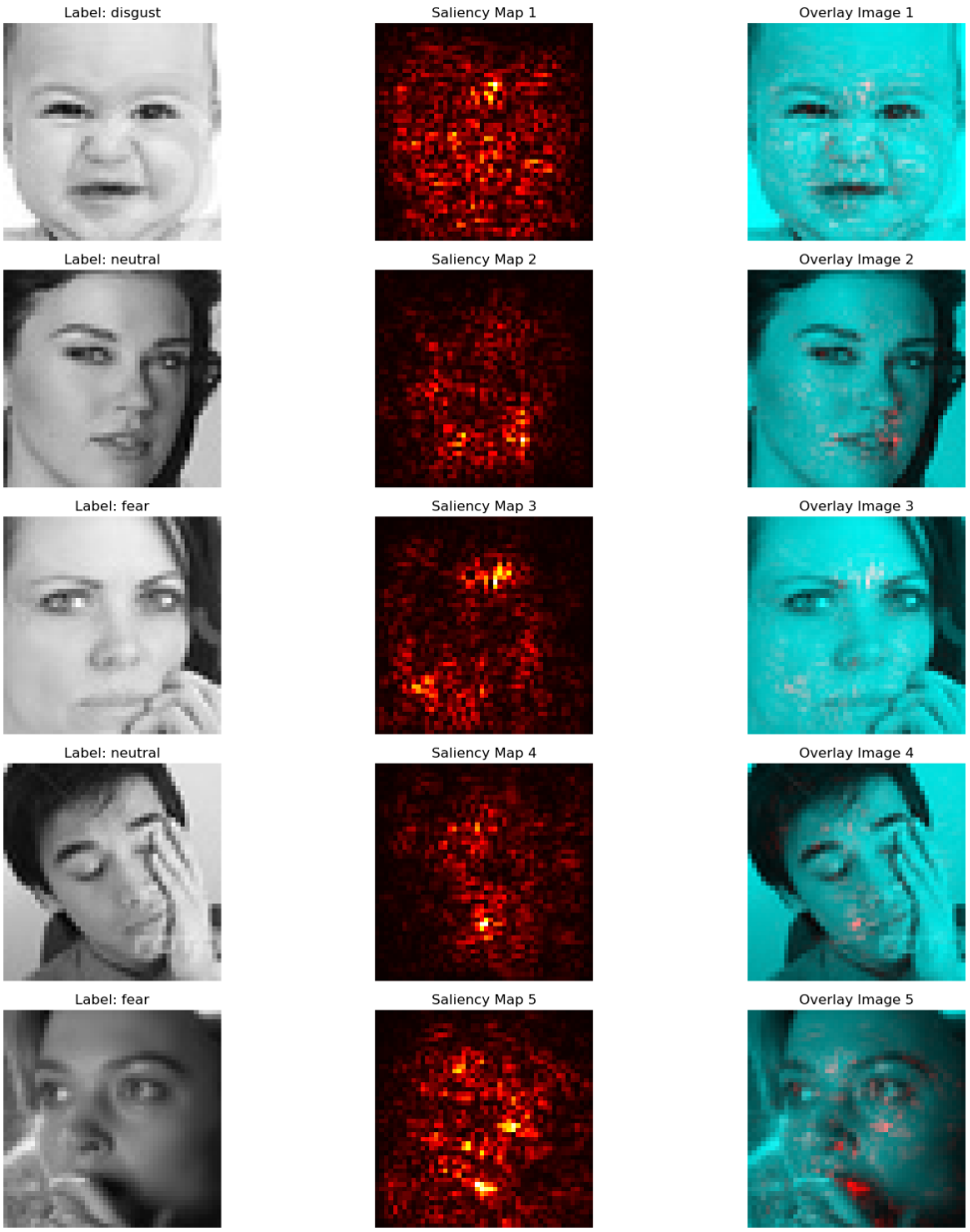

Figure 17 - Saliency Map of Random Sample of Images

7.4 Performance Comparison to Related Work

Model	Accuracy Rate
(Arriaga, Valdenegro-Toro and Plöger, 2017)	66%
Custom VGG16 (Our Model)	**67.11%**
(Ionescu, Popescu and Grozea, 2013)	67.48%
(Kusuma, Jonathan and Lim, 2020)	69.40%
(Cao et al., 2020)	71%
(Khaireddin and Chen, 2021)	73.28%

Table 4 - Performance Comparison with Existing Models

8. Deployment (CRISP-DM Phase 6)

After successfully training and evaluating our facial expression recognition model, we explored the deployment of our solution in a real-world scenario. We decided to showcase our model's practical application and potential impact in a simple system.

We developed a real-time FER system using OpenCV, a widely used computer vision library (OpenCV, 2024). The system used the custom model, which was loaded and prepared for inference. Additionally, we utilized the library's pre-trained Haar Cascade Classifier, an algorithm for object detection, to locate and extract facial regions from the video feed (Tan, 2024). As seen in Figure 18, we deployed the FER model and successfully detected all the emotion categories related to our dataset.

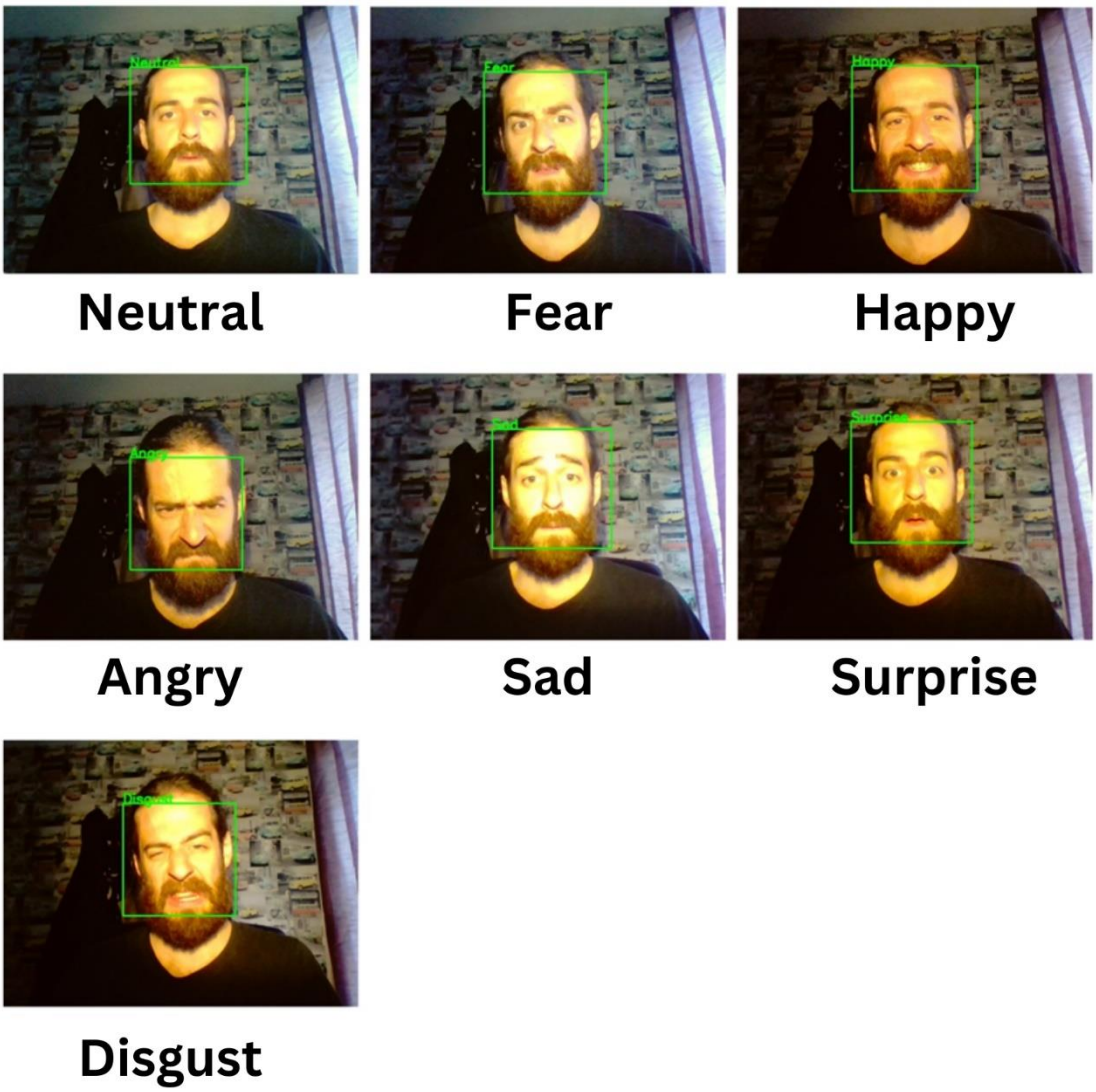

Figure 18 – Real-Time FER System Demonstration

9. Conclusion

This capstone project successfully developed a robust and accurate FER model using deep learning techniques, specifically CNN. Following the structured CRISP-DM framework, we systematically addressed each phase, from business understanding and data exploration to model development, evaluation, and deployment.

Through data preprocessing, including data augmentation techniques, we attempted to mitigate the class imbalance issue in the FER-2013 dataset and increase the diversity of the training data.

Many challenges were encountered while searching for the most accurate model possible, including computing and processing limitations, and lack of experience with deep learning models and neural networks.

While the proposed model showed promising performance, there is still room for improvement. Potential limitations include the inherent biases and poor data quality present in the dataset, the need for more extensive computational resources for exhaustive hyperparameter tuning, and the challenge of generalizing to diverse real-world scenarios.

Future work could explore additional data augmentation and preprocessing techniques, experiment with other state-of-the-art CNN architectures, and collect, train and merge larger, more diverse datasets.

Appendix

GitHub Repository Link

GitHub FER Capstone

Reflective Journals

Danrlei Martins

This capstone project was a challenging and rewarding experience. Initially, I was concerned about working with deep learning and computer vision, as it was a completely new domain for me. However, after much effort to understand its foundational principles, I can say that the field of deep learning intrigues me, and I am very curious about how it can impact my favourite domain, cybersecurity.

The data preparation phase was particularly insightful, as I learned the importance of data augmentation and preprocessing in achieving better model performance. Exploring different augmentation techniques and visualizing their effects on the dataset was fun and exciting.

The biggest challenge was dealing with the highly imbalanced dataset. I tried to apply some methods, such as SMOTE, to deal with that issue, but unfortunately, I was unsuccessful, and I understand more research has to be done regarding this aspect.

Managing other college modules and this project while working full-time was another obstacle, so having realistic expectations and setting task priorities was vital to have the project ready

on time. I could not achieve this work alone, and excellent collaboration with Leonardo was critical to succeed in this project.

Looking back, I am proud of the progress we made and the results we achieved. However, I also recognize the limitations of our approach and the potential for further improvement. Overall, this capstone project has been an excellent learning experience, and I am grateful for the opportunity to work on a real-world problem. The skills improved during this project will be very important to my future in the IT industry.

Leonardo Diesel

Working in this capstone project was something that challenged me since the beginning. Since the brainstorming for the project, me and my project partner tried to go beyond what we thought we could reach. It was concerning the challenges of working with neural networks but as the project went, I got to understand little by little how it works and its achievements.

There are many fields in the IT world and Machine Learning is a huge area that can be explored, and through the project they got even more evident. All the process of data preparation and augmentation was very interesting and to see how that can change in the learning/teaching of the model it is definitely a major change for the model.

Although the data part was very interesting, the challenges I face throughout the building of the model were the ones that most spoke to me. The research to find everything that could change the model's accuracy and then the testing of all of them were at times upsetting, but very satisfying. It is very hard though when theory does not apply. Of course, working as a team was essential to get to a final model, as many times Danrlei came up with very good ideas and peace of mind to help and enhance the project.

Even though our resources were not the best, I look back in a broad view to this extensive project and I can say that I am very happy and grateful for our achievements, the knowledge acquired and the partnership that I had with Danrlei. Working in a project like this makes me believe that we both could be able to work in the field as I feel prepared to face new challenges even more difficult.

References

Alqaraawi, A., Schuessler, M., Philipp Weiß, Costanza, E. and Berthouze, N. (2020). Evaluating Saliency Map Explanations for Convolutional Neural Networks: A User Study. *arXiv (Cornell University)*. doi:https://doi.org/10.48550/arxiv.2002.00772.

Arriaga, O., Valdenegro-Toro, M. and Plöger, P. (2017). Real-time Convolutional Neural Networks for Emotion and Gender Classification. [online] *arXiv (Cornell University)*. doi:https://doi.org/10.48550/arxiv.1710.07557.

Bansal, M., Kumar, M., Sachdeva, M. and Mittal, A. (2021). Transfer Learning for Image Classification Using VGG19: Caltech-101 Image Data Set. *Journal of Ambient Intelligence and Humanized Computing*, [online] 14, pp.3609–3620. doi:https://doi.org/10.1007/s12652-021-03488-z.

Barsoum, E., Zhang, C., Ferrer, C.C. and Zhang, Z. (2016). Training deep networks for facial expression recognition with crowd-sourced label distribution. *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. [online] doi:https://doi.org/10.1145/2993148.2993165.

Bartz, E., Bartz-Beielstein, T., Zaefferer, M. and Mersmann, O. (2023). *Hyperparameter Tuning for Machine and Deep Learning with R*. 1st ed. [online] Singapore: Springer. Available at: https://link.springer.com/book/10.1007/978-981-19-5170-1 [Accessed 12 May 2024].

Bettadapura, V. (2012). Face Expression Recognition and Analysis: The State of the Art. *arXiv (College of Computing, Georgia Institute of Technology)*. [online] doi:https://doi.org/10.48550/arxiv.1203.6722.

Cao, W., Feng, Z., Zhang, D. and Huang, Y. (2020). Facial Expression Recognition via a CBAM Embedded Network. *Procedia Computer Science*, [online] 174, pp.463–477. doi:https://doi.org/10.1016/j.procs.2020.06.115.

Chapman, P., Kerber, R., Clinton, J., Khabaza, T., Reinartz, T., Wirth, R. and Shearer, C. (2000). *CRISP-DM 1.0 step-by-step data mining guide*. [online] Available at: https://www.kde.cs.uni-kassel.de/wp-content/uploads/lehre/ws2012-13/kdd/files/CRISPWP-0800.pdf [Accessed 20 Apr. 2024]

Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16(16), pp.321–357. doi:https://doi.org/10.1613/jair.953.

Draelos, MD, PhD, R. (2019). Measuring Performance: The Confusion Matrix. [online] Glass Box. Available at: https://glassboxmedicine.com/2019/02/17/measuring-performance-the-confusion-matrix/ [Accessed 16 May 2024].

Feurer, M. and Hutter, F. (2019). Hyperparameter Optimization. *Automated Machine Learning*, [online] pp.3–33. doi:https://doi.org/10.1007/978-3-030-05318-5_1.

Fortune Business Insights (2023). *Global Emotion Detection and Recognition Market Size, Share & Growth Analysis By Type, By Application with Regional Forecast, 2022-2029*. [online] Available at: https://www.fortunebusinessinsights.com/industry-reports/emotiondetection-and-recognition-market-101326 [Accessed 5 Mar. 2024].

Forruque Ahmed, S., Bin, S., Hassan, M., Rodela Rozbu, M., Ishtiak, T., Rafa, N., Mofijur, M., Shawkat, A. and Gandomi, A.H. (2023). Deep learning modelling techniques: current progress, applications,

advantages, and challenges. *Artificial Intelligence Review*, [online] 56, pp.3521–13617. doi:https://doi.org/10.1007/s10462-023-10466-8.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*. [online] MIT Press. Available at: https://www.deeplearningbook.org [Accessed 30 Apr. 2024].

Goodfellow, I.J., Erhan, D., Luc Carrier, P., Courville, A., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D.-H., Zhou, Y., Ramaiah, C., Feng, F., Li, R., Wang, X., Athanasakis, D., Shawe-Taylor, J., Milakov, M., Park, J. and Ionescu, R. (2013). Challenges in representation learning: A report on three machine learning contests. *ICONIP 2013: Neural Information Processing*, [online] 64, pp.117–124. doi:https://doi.org/10.48550/arXiv.1307.0414.

Grandini, M., Bagli, E. and Visani, G. (2020). Metrics for Multi-Class Classification: an Overview. *arXiv (Cornell University)*. [online] doi:https://doi.org/10.48550/arxiv.2008.05756.

Hosna, A., Merry, E., Gyalmo, J., Alom, Z., Aung, Z. and Azim, M.A. (2022). Transfer learning: a friendly introduction. *Journal of Big Data*, [online] 9(1). doi:https://doi.org/10.1186/s40537-022-00652-w.

Ionescu, R., Popescu, M. and Grozea, C. (2013). Local Learning to Improve Bag of Visual Words Model for Facial Expression Recognition. In *Proceedings of ICML Workshop on Challenges in Representation Learning*. [online] Available at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=97088cbbac03bf8e9a209403f097bc9af46a4ebb [Accessed 17 May 2024].

Keras (n.d.). *Keras documentation: Optimizers*. [online] keras.io. Available at: https://keras.io/api/optimizers/ [Accessed 4 May 2024].

Keras (2019). *Keras documentation: RandomSearch Tuner*. [online] keras.io. Available at: https://keras.io/api/keras_tuner/tuners/random/ [Accessed 7 May 2024].

Keras (2024a). Keras documentation: EarlyStopping. [online] keras.io. Available at: https://keras.io/api/callbacks/early_stopping/ [Accessed 4 May 2024].

Keras (2024b). *Keras documentation: ReduceLROnPlateau*. [online] keras.io. Available at: https://keras.io/api/callbacks/reduce_lr_on_plateau/ [Accessed 4 May 2024].

Khaireddin, Y. and Chen, Z. (2021). Facial Emotion Recognition: State of the Art Performance on FER2013. *arXiv (Cornell University)*. doi:https://doi.org/10.48550/arxiv.2105.03588.

Kirk, M. (2017). *Thoughtful Machine Learning with Python: A Test-Driven Approach*. 1st ed. [online] Sebastopol: O'Reilly Media. Available at: https://moodle.cct.ie/mod/resource/view.php?id=37605 [Accessed 14 May 2024].

Kusuma, G.P., Jonathan, J. and Lim, A.P. (2020). Emotion Recognition on FER-2013 Face Images Using Fine-Tuned VGG-16. *Advances in Science, Technology and Engineering Systems Journal*, [online] 5(6), pp.315–322. doi:https://doi.org/10.25046/aj050638.

Lin, M., Chen, Q. and Yan, S. (2014). Network In Network. [online] *arXiv.org*. doi:https://doi.org/10.48550/arXiv.1312.4400.

Müller, A.C. and Guido, S. (2017). *Introduction to Machine Learning with Python: A Guide for Data Scientists*. 1st ed. [online] Beijing: O'Reilly Media. Available at: https://moodle.cct.ie/mod/resource/view.php?id=37688 [Accessed 14 May 2024].

O'Malley, T. (2020). *Hyperparameter Tuning with Keras Tuner*. [online] TensorFlow Blog. Available at: https://blog.tensorflow.org/2020/01/hyperparameter-tuning-with-keras-tuner.html [Accessed 11 May 2024].

O'Malley, T., Bursztein, E., Long, J., Chollet, F., Jin, H. and Invernizzi, L. (2019). *Keras documentation: KerasTuner*. [online] keras.io. Available at: https://keras.io/keras_tuner/. [Accessed 04 May 2024].

OpenCV (2024). OpenCV: Introduction. [online] Open Source Computer Vision. Available at: https://docs.opencv.org/4.x/d1/dfb/intro.html [Accessed 17 May 2024].

Pise, A.A., Alqahtani, M.A., Verma, P., K, P., Karras, D.A., S, P. and Halifa, A. (2022). Methods for Facial Expression Recognition with Applications in Challenging Situations. *Computational Intelligence & Neuroscience*, [online] pp.1–17. doi:https://doi.org/10.1155/2022/9261438.

Rosebrock, A. (2019). *Keras ImageDataGenerator and Data Augmentation*. [online] PyImageSearch. Available at: https://pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/ [Accessed 30 Apr. 2024].

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C. and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, [online] 115(3), pp.211–252. doi:https://doi.org/10.1007/s11263-015-0816-y.

Samadiani, Huang, Cai, Luo, Chi, Xiang and He (2019). A Review on Automatic Facial Expression Recognition Systems Assisted by Multimodal Sensor Data. *Sensors*, [online] 19(8), p.1863. doi:https://doi.org/10.3390/s19081863.

Shorten, C. and Khoshgoftaar, T.M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, [online] 6(1). doi:https://doi.org/10.1186/s40537-019-0197-0.

Simonyan, K. and Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *Computer Science: Computer Vision and Pattern Recognition*. [online] doi:https://doi.org/10.48550/arXiv.1409.1556.

Singh, V.K. and Joshi, K. (2022). Integrating Fairness in Machine Learning Development Life Cycle: Fair CRISP-DM. *e-Service Journal*, [online] 14(2), pp.1–24. doi:https://doi.org/10.2979/esj.2022.a886946

Shearer, C. (2000) The CRISP-DM Model: The New Blueprint for Data Mining. *Journal of Data Warehousing*, [online] 5(4), pp.13–22. Available at: https://mineracaodedados.wordpress.com/wp-content/uploads/2012/04/the-crisp-dm-model-the-new-blueprint-for-data-mining-shearer-colin.pdf [Accessed 20 Apr. 2024]

Sunyoto, A., Pristyanto, Y., Setyanto, A., Alarfaj, F., Almusallam, N. and Alreshoodi, M. (2022). The Performance Evaluation of Transfer Learning VGG16 Algorithm on Various Chest X-ray Imaging Datasets for COVID-19 Classification. *International Journal of Advanced Computer Science and Applications*, [online] 13(9), pp.196–203. doi:https://doi.org/10.14569/ijacsa.2022.0130923.

Tammina, S. (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. *International Journal of Scientific and Research Publications (IJSRP)*, [online] 9(10). doi:https://doi.org/10.29322/ijsrp.9.10.2019.p9420.

Tan, A. (2024). *Using Haar Cascade for Object Detection*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/using-haar-cascade-for-object-detection/ [Accessed 17 May 2024].

TensorFlow (2024). *ImageDataGenerator*. [online] TensorFlow. Available at: https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator [Accessed 30 Apr. 2024].

Tian, Y., Kanade, T. and Cohn, J.F. (2011). Facial Expression Recognition. *Handbook of Face Recognition*, [online] pp.487–519. doi:https://doi.org/10.1007/978-0-85729-932-1_19

Voulodimos, A., Doulamis, N., Doulamis, A. and Protopapadakis, E. (2018). Deep Learning for Computer Vision: a Brief Review. *Computational Intelligence and Neuroscience, 2018*, [online] pp.1–13. doi:https://doi.org/10.1155/2018/7068349.