**Facial Emotion Recognition**

**Cody Mott Gavin Jones**

mottc@msoe.edu jonesg@msoe.edu

Milwaukee School of Engineering   
University

**TECHNICAL ABSTRACT**

This project aimed to build a Convolutional Neural Network model with the ability to predict human emotions based off facial expressions. We found a dataset containing 48x48x1 images of human faces that were labelled with one of seven different emotions. We built three total CNN models over the course of the project, progressively improving each model. We began by running a simple CNN model on just the raw data split into 70% training, 20% validation, and 10% testing splits. When this model resulted in major overfitting problems, we used data augmentation to get a more even distribution of data. We then built an improved model with more convolutional layers and better hyperparameters, which seemed to get rid of our overfitting problem. Finally, we built a third model which made slight improvements to our second model, and altered the data to remove outlier categories so we would have a better dataset. This model provided the best results with a higher testing accuracy and better loss values, and a better confusion matrix. We found that our big drawback in achieving great results was the dataset itself as there was corrupted or bad data that we were unable to remove within the timeframe of the project. The project highlighted how complex human emotions could be, and how similar some emotion could look to other emotions without the proper context.

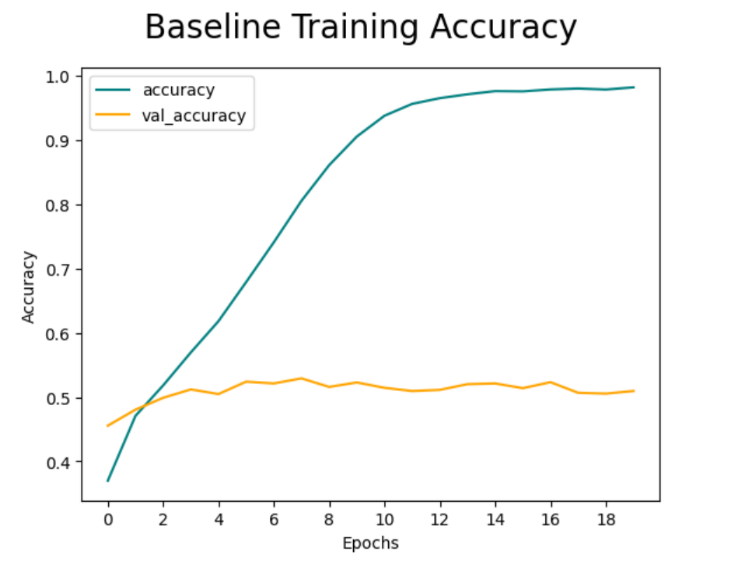
**INTRODUCTION**

We wanted to address the problem of recognizing emotion based off a persons facial expression, specifically to see if we could get an AI to recognize the emotion. Our objectives included finding a dataset to train on, manipulating the raw data to fit in a model, creating a CNN model as they are good for categorizing images into multiple different categories, training our model on the data, running tests and augmenting data to improve the model performance, and collecting results. For our dataset, we chose to use the FER-2013 facial emotion recognition dataset. This data was used as part of a Kaggle challenge back in 2013 to see who could train the best model to predict human emotions. The dataset included multiple files such as the train csv, the test csv, and a csv of all the data put together. In total the dataset contained 35,887 images of faces, each being 48x48x1. The image labels ranged from 0 to 6 to represent 7 different human emotions. The emotions included were angry, disgust, fear, happy, sad, surprise, and neutral. The dataset was created by Pierre-Luc Carrier and Aaron Courville and provided to Kaggle for the competition.

**BASELINE MODEL**

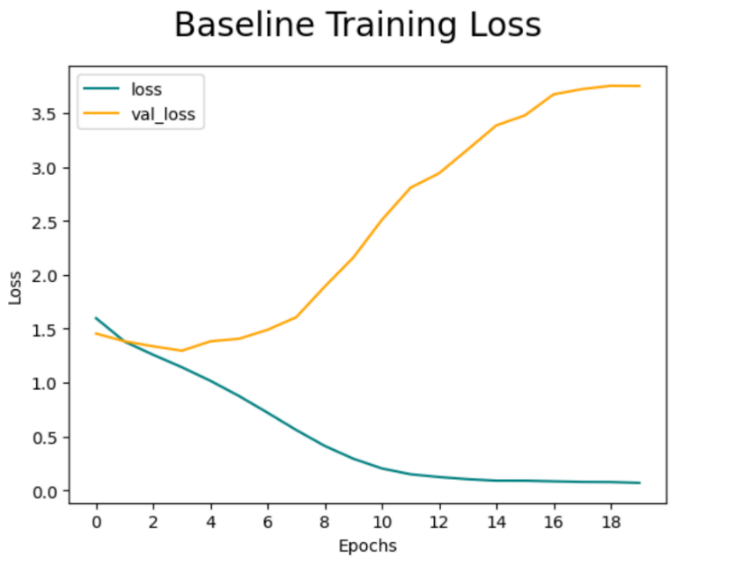
In developing a foundational approach for emotion recognition via facial expressions, we employed a basic Convolutional Neural Network (CNN) structure as our baseline model. This model's initial layer is a 2D convolutional layer, equipped with 32 filters and a kernel size of 3x3. It utilizes the ReLU activation function to introduce non-linearity, crucial for learning complex patterns in facial expressions. The layer's input shape is set to 48x48x1, aligning with the grayscale images in our dataset.

Following this, we implemented a max pooling layer to reduce spatial dimensions and to help in minimizing overfitting by providing an abstracted form of the representations. This process was replicated, adding another convolution and max pooling pair, this time with an increased capacity of 64 filters. Each convolutional layer is designed to learn the specified number of filters, while the subsequent max pooling layer reduces the spatial size of the representations by half, focusing on the most relevant features.

Post these layers, a Flatten layer was introduced. Its role is to transform the 2D feature maps into a 1-dimensional array, preparing the data for the fully connected dense layers. The network then proceeds through two dense layers; the first contains 256 neurons, offering a substantial level of learning capacity, and the second serves as the output layer with 7 neurons corresponding to the number of emotion categories.

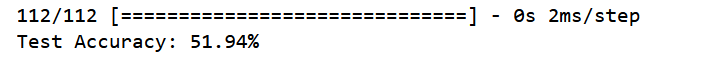
For the compilation of this model, the Adam optimizer was chosen, appreciated for its efficiency and effectiveness, with a learning rate of 0.001. The model employs a categorical cross-entropy loss function, suitable for multi-class classification tasks like ours, where categories are one-hot encoded.

Figure 1: Training and Validation Loss of Baseline Model

During training, however, we observed signs of overfitting as early as the third epoch. This was evident from Figure 1 and Figure 2, where the validation accuracy plateaued around 50%, while the validation loss began to increase, indicating the model's growing discrepancy in performance on training versus unseen data. The final testing accuracy, shown in Figure 3, stood at 51.94%, reinforcing the need for model improvements and adjustments.

This baseline model, while fundamental, offers critical insights into the challenges of emotion recognition using CNNs. The modest performance underscores the need for more sophisticated architectures or strategies to improve generalization and reduce overfitting.

Figure 2: Training and Validation Accuracy of Baseline Model

  
Figure 3: Baseline Model Testing Accuracy

**MODEL IMPROVEMENT APPROACHES**

**Improved Model v1: Architecture and Parameter Tuning**

***Model Architecture Enhancements:***

In our quest to refine the baseline model and address the overfitting issue, we expanded the architecture in Improved Model v1. This revised model includes four convolutional layers, each with progressively increasing filter counts: 16, 32, 64, and 128. A max pooling layer follows each convolutional layer, although the pool sizes remain unspecified. A significant addition to each layer is a dropout mechanism with a rate of 0.25, which helped in enhancing the model's performance by preventing overfitting. Another critical modification is the transition from the 'relu' to 'leaky\_relu' activation function. This change was prompted by the specific challenges presented by our dataset, which 'leaky\_relu' addresses more effectively.

Post the convolutional layers, the architecture incorporates a Flatten layer and a Dense layer. Notably, we reduced the neuron count in the Dense layer from 256 to 128, which proved more efficient for training. Similar to the convolutional layers, the Dense layer also includes a dropout layer. We continued using the Adam optimizer for this iteration.

***Parameter Tuning & Data Manipulation:***

In our parameter tuning experiments, we observed that batch normalization reduced performance. After testing various batch sizes (16, 32, 64, 128, 256, 512, and 1024), both with and without batch normalization, the optimal performance was achieved with a batch size of 64 without batch normalization. Larger batch sizes hindered the model's learning capability, while smaller ones resulted in poorer validation and test performance.

Regarding activation functions (elu, relu, and leaky relu), leaky relu slightly outperformed the others. We also compared Adam and Nadam optimizers but found no significant difference, leading us to retain Adam.

Our experiments with learning rates (0.1, 0.01, 0.001, 0.0001) highlighted that 0.1 and 0.01 led to significant overfitting, while 0.0001 was too low for effective learning. A rate of 0.001 emerged as the most balanced choice.

In exploring the number of filters for each convolution layer, starting with 16 filters and doubling them in each subsequent layer yielded better results for validation data than beginning with higher filter counts.

We also experimented with various dropout rates (20-50%) and configurations, including removing some dropout layers or assigning different rates to each. A consistent dropout rate of 25% showed promise, but comprehensive testing remains to be done.

***Addressing Data Imbalance:***

We identified a significant imbalance in our dataset, with some emotion categories being underrepresented compared to others. To address this, we developed a function to quantify the disparity and used an ImageDataGenerator (datagen) to augment underrepresented categories. The selected parameters for datagen included a rotation range of 10, width and height shift ranges of 0.05, a shear range of 0.1, a zoom range of 0.05, and horizontal flipping with 'nearest' as the fill mode. This approach effectively balanced our dataset, allowing us to have an equal number of images (8,989) for each emotion category.

*Performance Analysis of Improved Model v1:*

A graph of a graph

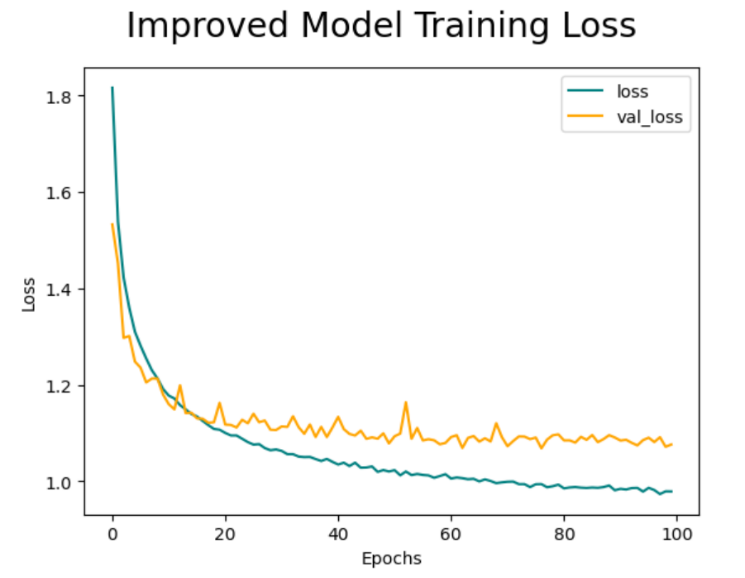
Description automatically generated with medium confidence

Figure 4: Training and Validation Accuracy of Improved Model v1

Figure 5: Training and Validation Loss of Improved Model v1

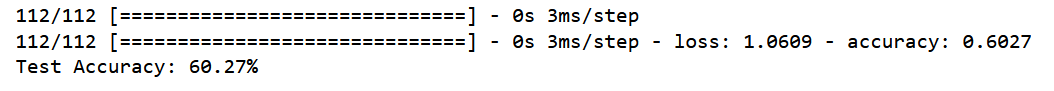


Figure 6: Improved Model v1 Test Accuracy

The Improved Model v1 demonstrated a notable advancement over the baseline model, yet it encountered limitations in learning complexity from the data. Over the course of 100 epochs, as A chart with different colored squares

Description automatically generateddepicted in *Figure 4*, the training accuracy plateaued at 64%. This outcome suggests that the model's architecture, despite improvements, may still be too simplistic to capture the nuances in the data fully. A more intricate model structure might be necessary to achieve higher learning efficacy.

Interestingly, the validation loss did not show a significant increase, indicating some level of model stability. However, it appears to have started leveling off between 30 to 40 epochs. This observation points towards a need for further adjustments in the model's learning process or architecture to enhance its ability to generalize better.

Figure 7: Improved Model v1 Confusion Matrix

The confusion matrix in *Figure 7* offers valuable insights into the model's performance across different emotions. The model showed the highest accuracy in predicting the 'happy' emotion, achieving 80.88%. 'Surprise' followed with a respectable 75.44% accuracy. The accuracies for 'disgust' and 'neutral' were somewhat moderate at 63.27% and 63.25%, respectively. However, the model struggled significantly with 'sad' and 'fear', attaining only 55.52% and 27.94% accuracy, respectively. Notably, the model had difficulty distinguishing between 'fear' and 'sad', as well as 'sad' and 'neutral'.

This differential performance across emotions highlights specific areas where the model requires refinement. The difficulty in discerning closely related emotions like 'fear', 'sad', and 'neutral' suggests that the model may benefit from further tuning of its feature detection capabilities or even an expansion of the dataset with more varied examples of these particular emotions.

**Improved Model v2: Enhancements and Evaluation**

***Model Architecture Enhancements:***

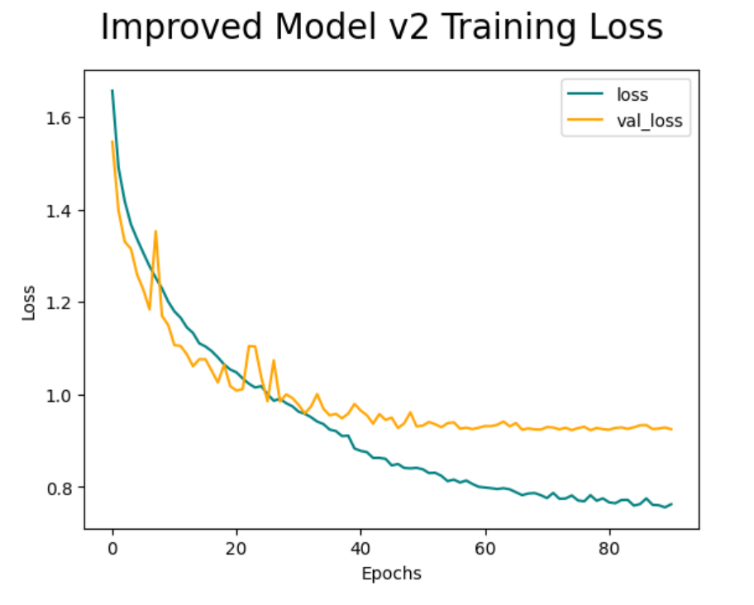
Improved Model v2 builds upon the previous iteration with four convolutional blocks, each comprising a 2D Convolution layer, Batch-normalization layer, Activation layer, 2D Max Pooling layer, and a Dropout layer. This sequence culminates in a Flatten layer, followed by a Dense layer, another Dropout layer, and a final Dense layer. Key enhancements in this model include the integration of batch normalization, weight normalization, a batch size of 128, and callback functions such as early stopping and a learning rate scheduler. These modifications, combined with data manipulation strategies, have notably improved the model's performance.

*Parameter Tuning & Data Manipulation:*

We continued to experiment with various parameters, such as filter numbers, dropout rates, and learning rates. Notably, implementing batch normalization and increasing the batch size to 128 yielded better results. In exploring weight normalization, we compared 'he\_normal' and 'glorot\_normal', finding that 'glorot\_normal' performed marginally better.

The primary challenge with the dataset was its quality, including mislabeled images, watermarks, obstructed faces, and ambiguous expressions. Additionally, the dataset's uneven distribution prompted us to down-sample the largest category (happy), exclude the smallest (disgust), and remove the most ambiguous category (neutral). This restructuring aimed to create a more balanced dataset for effective model training.

*Model Evaluation & Results*

**

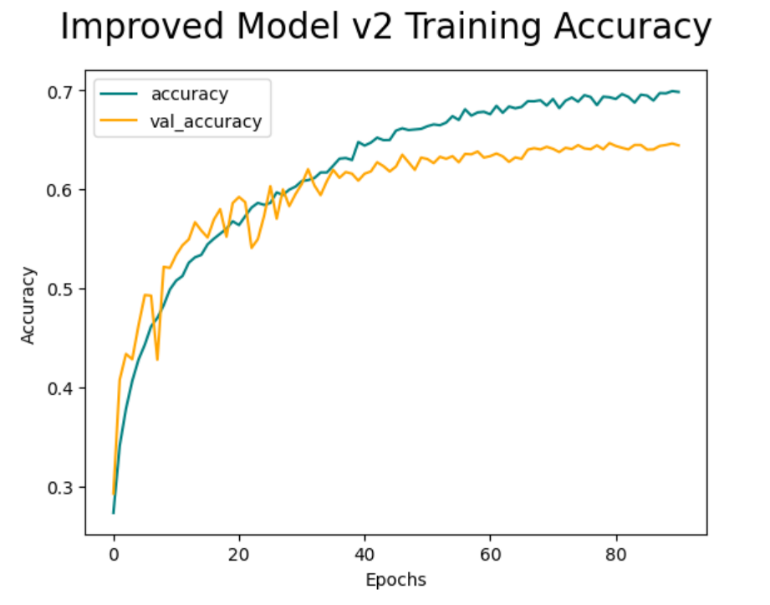
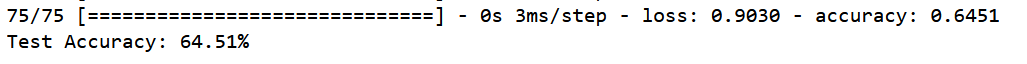
**

Figure 8: Training and Validation Accuracy of Improved Model v2

Figure 9: Training and Validation loss of Improved Model v2

*A chart of different colored squares

Description automatically generated*

Figure 10: Test Accuracy of Improved Model v2

As illustrated in *Figure 8*, Improved Model v2 achieved a test accuracy of 64.51%, marking it as our best-performing model. The model was particularly adept at predicting 'happy' (81.98% accuracy), 'sad' (71% accuracy), and 'surprised' (77.97% accuracy). 'Angry' achieved a lower accuracy of 61.80%, often being confused with 'sad'. The most significant challenge was with 'afraid', which only had a 34.62% accuracy and was   
frequently misclassified as 'sad'.

Figure 11: Improved Model v2 Confusion Matrix

**Insights:**

The improved models consistently struggled to differentiate between negative emotions like fear, anger, and sadness. While a more complex architecture may be beneficial, the core issue lies in the dataset's quality and distribution. With significant disparities in category sizes and numerous low-quality images, future work would ideally focus on acquiring a more reliable dataset or extensively cleaning and augmenting the current one to balance the underrepresented categories.

**CONCLUSION**

For our final project, we sought to build a CNN model that could accurately predict human emotion when given a facial expression. In all we were successful in doing so despite the challenges that came along with our dataset. Our model was able to predict most emotions very accurately, with a highlight of the happy emotion which reached an 81.98% testing accuracy. We conducted many experiments involving altering our data and models so that we could achieve the best possible results. A big highlight from our project was eliminating the big overfitting problem. In our baseline model results there is obvious overfitting issues which would not have allowed us to create a reliable model. The one thing holding us back was the data that we chose. If we had more time, we would have gotten new data, or removed all the bad data before training. This was a nagging issue throughout the entire process and caused issues with our overall results. The data led to issues differentiating between certain negative emotions such as afraid and sad as shown above. Moving forward, we would want to fix the data so that we would not get the same issues. It would also help us to conduct more testing to ensure that we have the best parameters to train the model. Lastly, we would want to build a DNN model for comparison purposes. In All, our project was a success and taught us a lot about building a model and using data to create an accurate prediction system.

**NON-TECHNICAL ABSTRACT**

In this project, we had the goal of creating an Artificial Intelligence model that could predict human emotions when it was given facial expressions. While doing the project we created and tested multiple ideas for models. Our data was fed into these models to help them recognize differences in the facial expressions between different emotions. What resulted from our experiments was a model that could predict the emotion with 64% accuracy. We could have improved the accuracy by getting new data as we had some of our data corrupted with watermarks on the images, as well as blank images, or images that didn’t show enough of the face. This means that the model would be learning on images that didn’t show the emotion very good and would therefor lead to worse accuracy scores when tested on other images.

**REFLECTION**

Over the course of this project, we ran into many issues that we had to solve. Most of these issues were unexpected and things that we had never dealt with before. This forced us to come up with solutions that we never even knew existed until beginning work on our model. An example of this would be when we ran into an issue getting our model to stay consistent throughout the entire training process. This led us to finding callback functions that we could add to the model, which we had never known about until researching ways to fix the problem. This let us introduce early stopping and a learning rate scheduler so that we could have the best model possible. Not only did this project teach us problem solving techniques and more about CNN model creation, but we also saw how our model could be used in the real world. The model is trained on data from so many places that it became good at detecting certain emotions. One such way this could be implemented in the real world would be during usability testing. When conducting usability testing, the testers want to be able to know as much as possible from the people being tested on, but they have limitations. During the test the user says all their thoughts on the product out loud, and on occasion there is an eye tracker used to see what catches the users eye. Adding an emotion model on top of this could help to advance the test by seeing more than just what you are being told, which could help to eliminate bias in the test, as well as get a deeper understanding for how the user really feels about certain aspects of a product. All of this can help companies create even better products for its users and could be a very valuable tool when used correctly. In all, this project taught us a lot about how we can use data and data alteration techniques to create better models, which can be used in the real world.