Bérénice Le Glouanec 04/2023

Machine Learning for Statistical NLP: Advanced LT2326

Facial Expression Recognition

Abstract

The aim of this project is to recognize which kind of emotion is expressed by a subject among pictures. This code implements a facial expression recognition model using a convolutional neural network (CNN) based on LeNet-5 architecture model. The FER2013 dataset consists of over 35,000 grayscale images of faces labeled with one of seven emotions. The CNN model consists of three convolutional layers followed by three fully connected layers, with ReLU activations and dropouts. The model is trained using the Adam optimizer and cross-entropy loss function. The training process achieved a near-perfect accuracy of 99.3% on the training set but only 53.7% on the testing set, indicating possible overfitting. The model was able to classify happy and surprise expressions with high accuracy, but struggled with sad and fear expressions.

Introduction

The ability to recognize emotions in human faces is a crucial aspect of social interaction and communication. Traditional methods of emotion recognition have relied on subjective evaluations by human experts, which can be time-consuming and prone to errors. With the advent of machine learning and artificial intelligence, there has been a growing interest in using computational methods to automate emotion recognition. One popular approach involves training neural networks on large datasets of facial expressions to learn how to classify emotions.

In this context, we present a neural network model for facial emotion recognition based on the Facial Expression Recognition 2013 (FER2013) dataset. Our model uses a deep convolutional neural network architecture with three hidden layers to extract features from the images, followed by three fully connected layers to perform classification. We train the model using a cross-entropy loss function and the Adam optimizer, and evaluate its performance on separate training and testing datasets.

The results of our experiments show that our model achieves high accuracy on the training set, but lower accuracy on the testing set. We discuss possible solutions to this problem and suggest directions for future work to improve the model's performance.

Background

Initially, the aim was to classify emotions into binary categories: 0 for negative emotions such as sadness, anger, and fear, and 1 for positive emotions such as happiness, neutral, and surprise. However, it was found that isolating parameters unique to one class and not the other was not possible, which could have affected the overall results. Thus, the model is based on a multi-class classification approach instead of a binary one.

When selecting a suitable dataset for this machine learning project, several options were considered, such as the Google Facial Expression Comparison Dataset, AffectNet, the Extended Cogn-Kanade Dataset, and the FER2013 dataset, which was created for an ongoing Kaggle competition. The FER2013 dataset, which stands for "Facial Expression Recognition 2013," was chosen as it is simple, lightweight, and widely used, making it particularly suitable for this type of project.

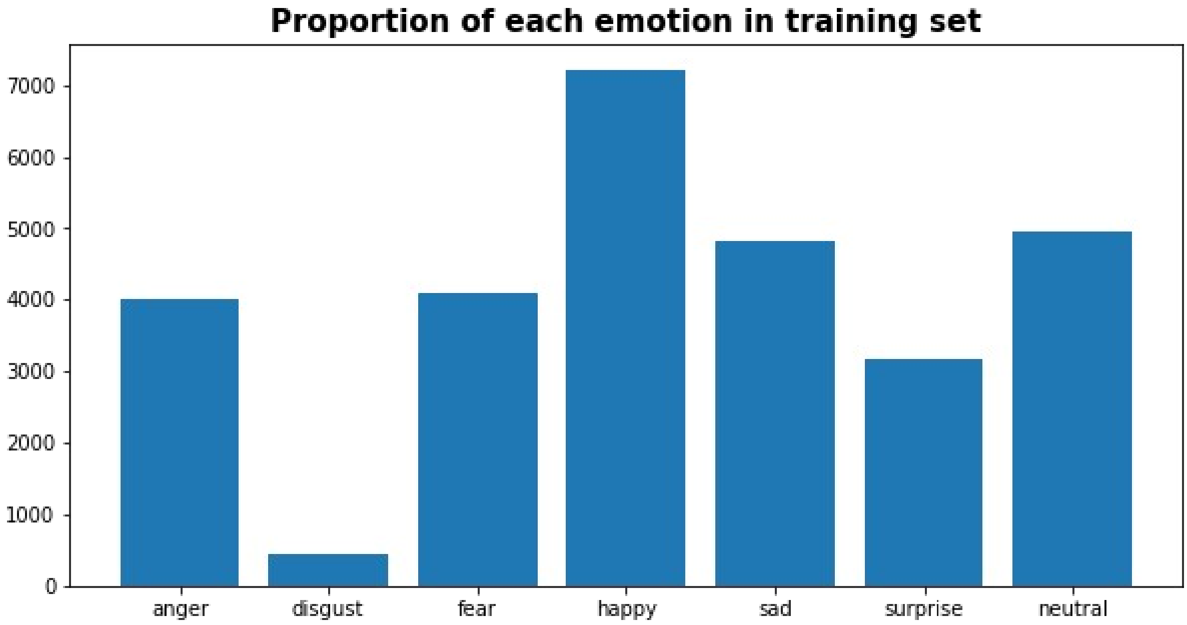
The goal of this project is to develop a deep learning model that can accurately classify facial expressions based on the images in the FER2013 dataset. Specifically, we aim to build a convolutional neural network (CNN) with multiple hidden layers that can learn to extract meaningful features from the images and use those features to predict the correct emotion label for each image. The performance of the model will be evaluated using accuracy scores on both the training set and the test set.

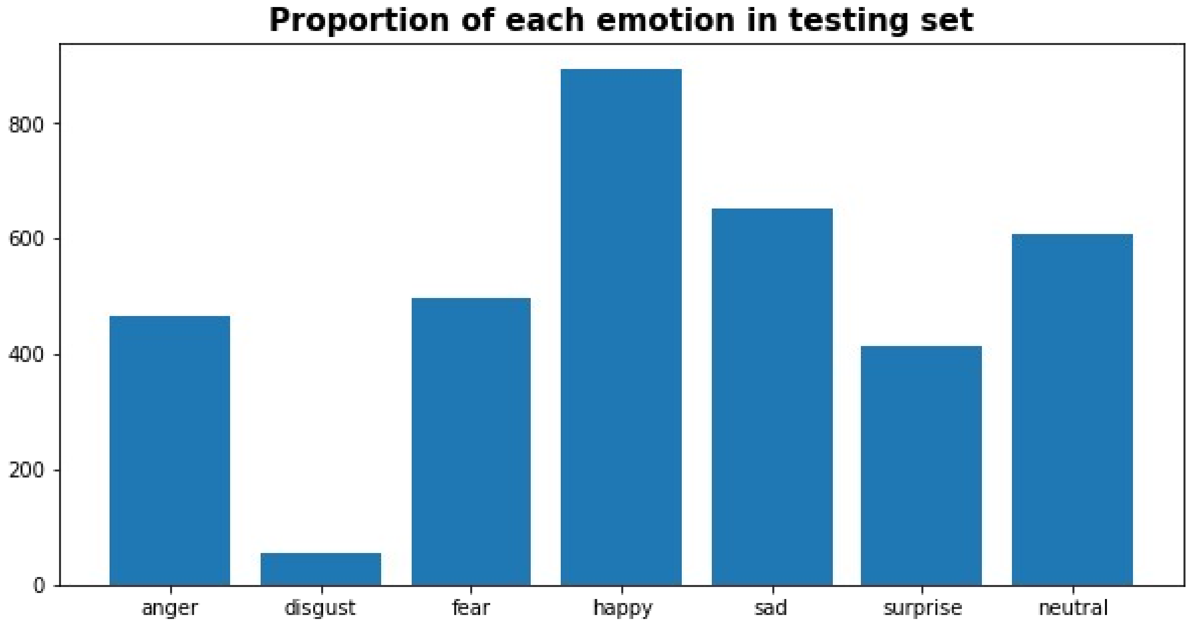
Methods

Dataset

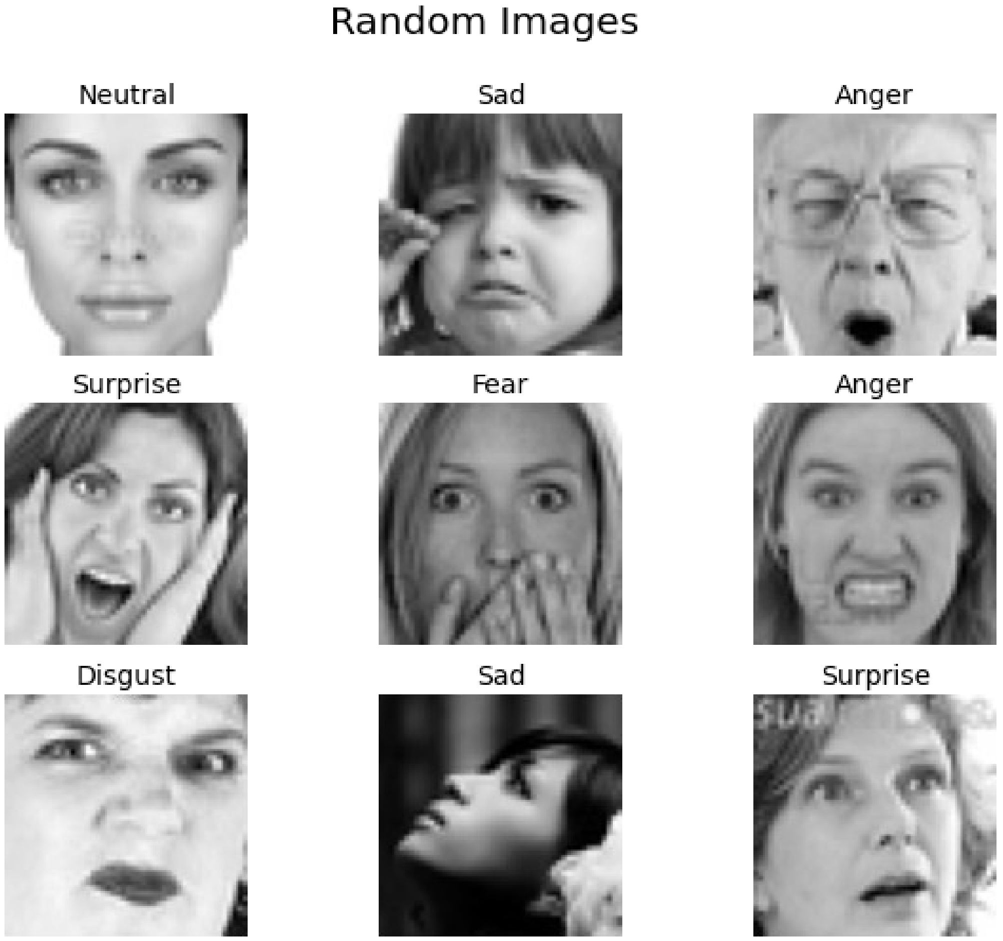
We used the "FER2013" dataset, which contains grayscale images of faces and their corresponding emotion labels. The dataset consists of 28,709 images for training, 3,589 for validation, and 3,589 for testing, each image having a size of 48x48 pixels. The emotion labels are classified into seven categories: anger, disgust, fear, happiness, sadness, surprise, and neutral.

Before processing any data, we checked the distribution of each label among the training and testing sets. A greater representation of one label compared to another one would lead to a better accuracy for the first label and worse for the second one.





According to our results, there is a bias in each set, some labels have more chances to be correctly identified than others. For example, the emotion “disgust” is more likely to be wrongly identified than the emotion “happy”. As it is important to understand what data we are working with, we reconstruct random images from the training set. This visualization helps to determine the visual quality of images (e.g. noisy images because of a bad quality that makes the face on an image impossible to be detected) and if there is any wrongly labeled instance (e.g. a picture of an angry face is labeled as a happy).



This small sample tends to indicate that the images are less likely to contain noise (neutral background, centered object, gray scaled …) or to be mislabeled. Additionally, as explained by the creators of FER2013, the images share a standard format, size and quality. In this situation, there is no need to determine a structured standard polygon to structure the dataset with a consistent and uniform format, neither to perform noise reduction.

Data Preprocessing

We first read the dataset as a CSV file using the pandas library. Then, we extracted the pixel values of the images and converted them into a numpy array. We split the dataset into a training set and a test set (80% for training and 20% for testing). The pixel values were normalized to a range of [0,1]. The images in the dataset were originally stored as a string of pixel values. We reshaped them into 48x48 matrices and stored them as numpy arrays. The emotion labels were originally stored as strings. We converted them to numerical values so that they could be used by the neural network (integer value between 0 and 6).

Model Architecture

We used a convolutional neural network (CNN) based on LeNet-5 architecture to classify the emotions in the FER2013 dataset. The model consists of three convolutional layers with ReLU activation functions, max-pooling and batch normalization layers. We then added three fully connected layers to the model, with ReLU activation functions and dropout layers to prevent overfitting. Finally, the output layer consists of seven neurons, each corresponding to one of the seven emotions in the dataset.

Training

We trained the model using the Adam optimizer with a learning rate of 0.0001 and a batch size of 128. We used the cross-entropy loss function as the objective function during training. The model was trained for 20 epochs.

Testing

We tested the model on the testing set and reported the accuracy, precision, recall and F1 score as the evaluation metric.

Evaluation

To evaluate the performance of our model, we used a standard method of splitting the dataset into training and testing sets. The training set was used to update the weights of the neural network during the learning process, while the testing set was used to evaluate the model's ability to generalize to new data.

We trained the model for 20 epochs, using the Adam optimizer with a learning rate of 0.0001 and a batch size of 128. During each epoch, we computed the training loss and accuracy and saved these values for later analysis. We also computed the test loss and accuracy for each epoch. After training the model, we evaluated its performance on the testing set. For the training set, the model achieved an accuracy of 99.3%, which suggests that it learned the features of the images and was able to accurately classify the emotions in the training data. However, the model's accuracy was only 53.7% on the testing set, which is significantly lower than the accuracy achieved on the training set. This indicates that the model may have overfit to the training data and is not able to generalize well to new data.

Table 1: Scores training set

Accuracy	99.3416698596259
F1-score	0.9325737689728546
Recall	0.9254241861667797
Precision	0.940534813887365

Table 2: Scores testing set

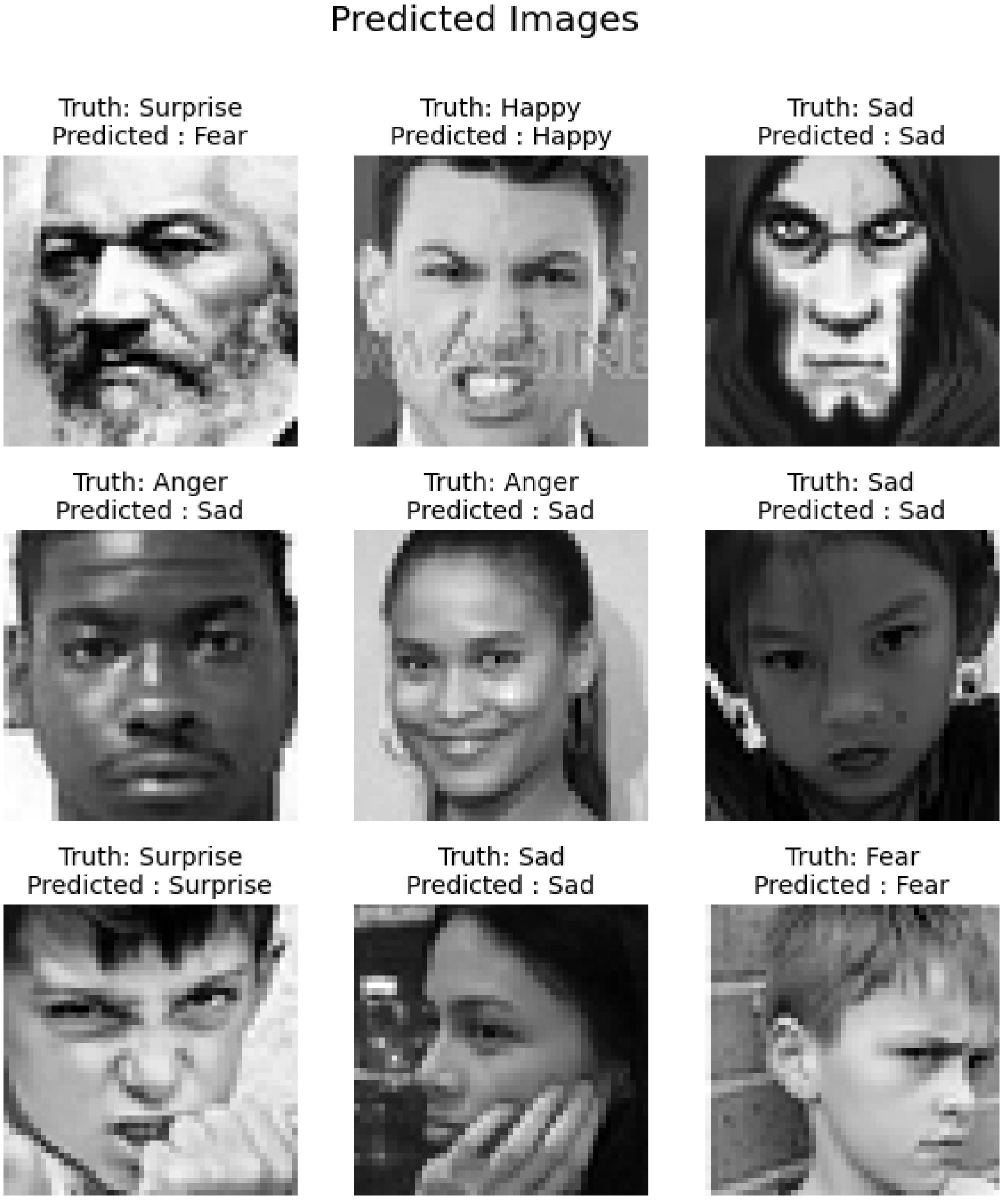
Accuracy	53.74756199498468
F1-score	0.5161748167263325
Recall	0.5002420723467257
Precision	0.5564931451326833

We also calculated the precision, recall, and F1-score for each class, which provide a more detailed view of the model's performance for each emotion. The evaluation of the model's performance using the testing set showed a precision score of 0.55, a recall score of 0.50, and an F1-score of 0.51. These results suggest that the model can correctly classify positive and negative emotions with an overall accuracy of 53.7%. The precision score indicates that the model have some false positives, while the recall score suggests that the model have missed some true positives.

Breaking down the scores per class, we see that the precision score is highest for the "disgust" class with a value of 0.76, and lowest for the "fear" class with a value of 0.45. The recall score is highest for the "happy" class with a value of 0.82, and lowest for the "sad" class with a value of 0.38. The F1-score is highest for the "surprise" class with a value of 0.71, and lowest for the "fear" class with a value of 0.39. These results suggest that the model struggle more with certain emotions than others.

Table 3: Scores per class in testing set

	Precision	**Recall**	**F1-score**	**Support**
Anger	0.45	0.43	0.44	9340
Disgust	0.76	0.39	0.52	1120
Fear	0.45	0.34	0.39	9920
Happy	0.61	0.82	0.70	17900
Sad	0.46	0.38	0.42	13060
Surprise	0.72	0.71	0.71	8300
Neutral	0.45	0.44	0.44	12140
Accuracy			0.54	71780
Macro avg	0.56	0.50	0.52	71780
Weighted avg	0.53	0.54	0.53	71780



Overall, the evaluation metrics suggest that our CNN model performed well on the training set, but did not generalize well to the test set. In the next section, we discuss some potential reasons for this and suggest future directions for improving the model's performance.

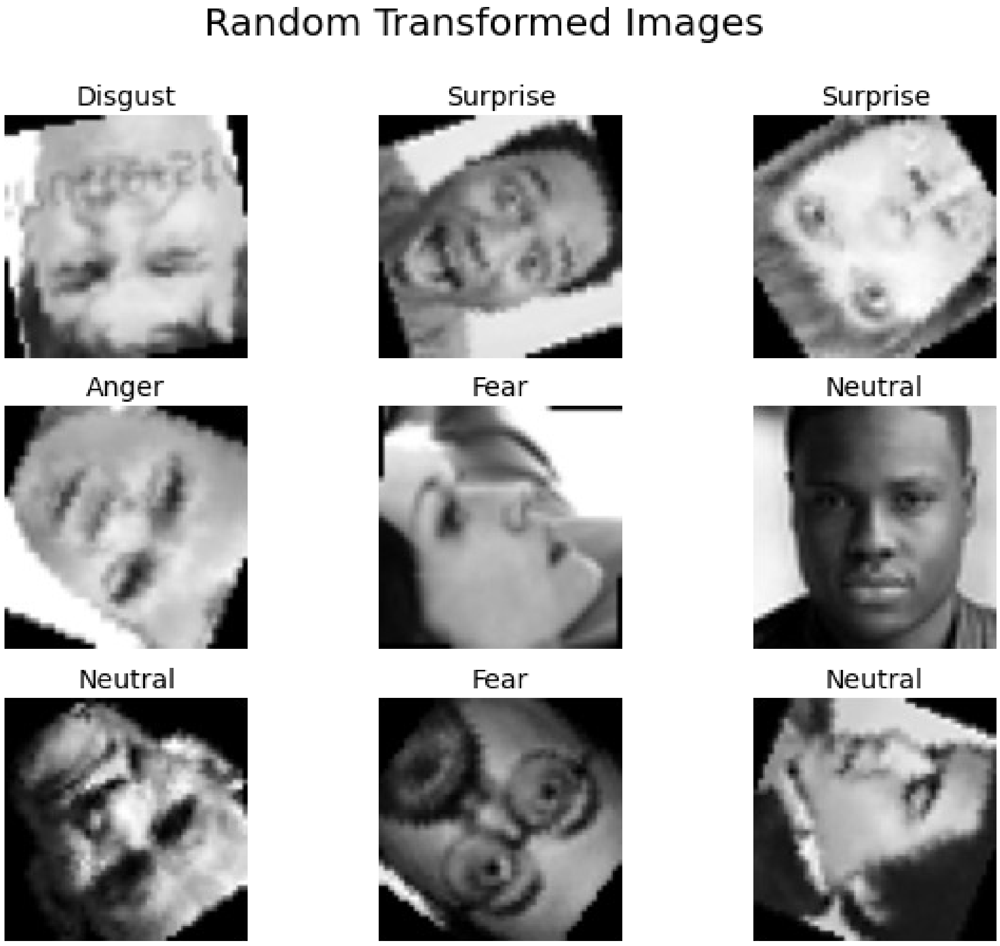
Discussion and Future Work

In this study, we proposed a convolutional neural network (CNN) model to classify facial expressions into seven categories: anger, disgust, fear, happy, sad, surprise, and neutral. Our model achieved an accuracy of 99.3% on the training set, but only 53.7% on the testing set, indicating that the model may be overfitting the training data.

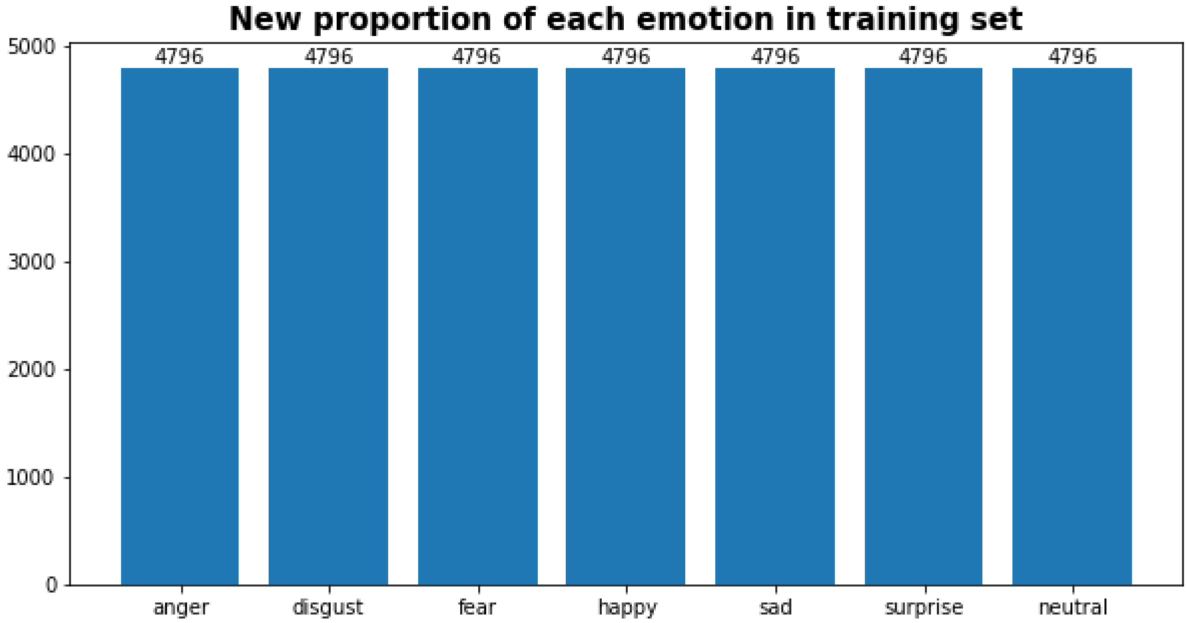
One possible reason for overfitting is the complexity of the model. Our CNN model has three hidden layers, each with a large number of neurons. While this allows the model to learn intricate features of the facial expressions, it also increases the risk of overfitting. To prevent overfitting and improve the model's performance on the testing set, the model's complexity could be reduced by decreasing the number of layers or neurons.

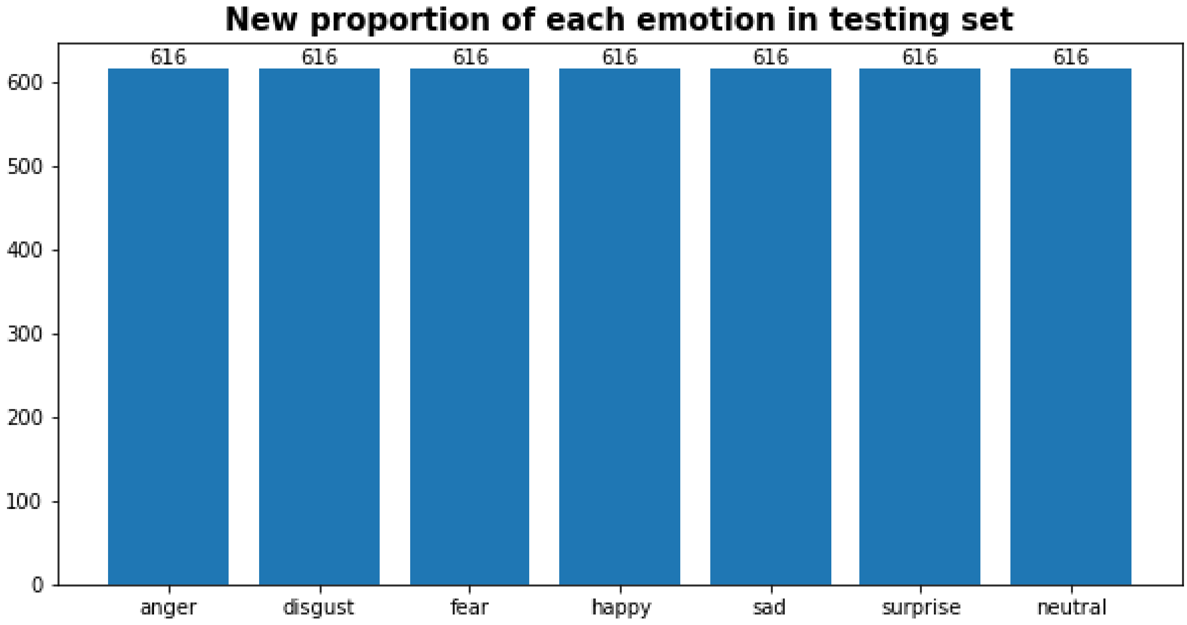
Furthermore, we observed that the model's performance varied across different emotions. For example, the model performed well in identifying happy and surprise expressions, but struggled with fear and sad expressions.

Another possible reason for the overfitting is the imbalanced distribution of images in the dataset. Some emotions are represented more frequently than others, which can lead the model to prioritize these emotions over others, resulting in lower accuracy for the less represented emotions. To address the issue of imbalanced data, we generated a new dataset with balanced class distributions using data augmentation techniques. Specifically, we created additional images for the underrepresented emotions by randomly rotating, flipping, scaling, adjusting brightness, and contrast, and applying other transformations to the original images.



In order to avoid any bias, we selected as much images as in the smaller class (i.e. 436 images for “disgust” class in training set), and created 10 transformed images for each of them (i.e. 4796 images in total per class in training set).





However, when we tested this new dataset on the same model with the same hyperparameters, we found that the performance was worse than the original dataset. The training accuracy was around 98.65% and the testing accuracy was only 28.64%.

Table 4: Scores training set with data augmentation

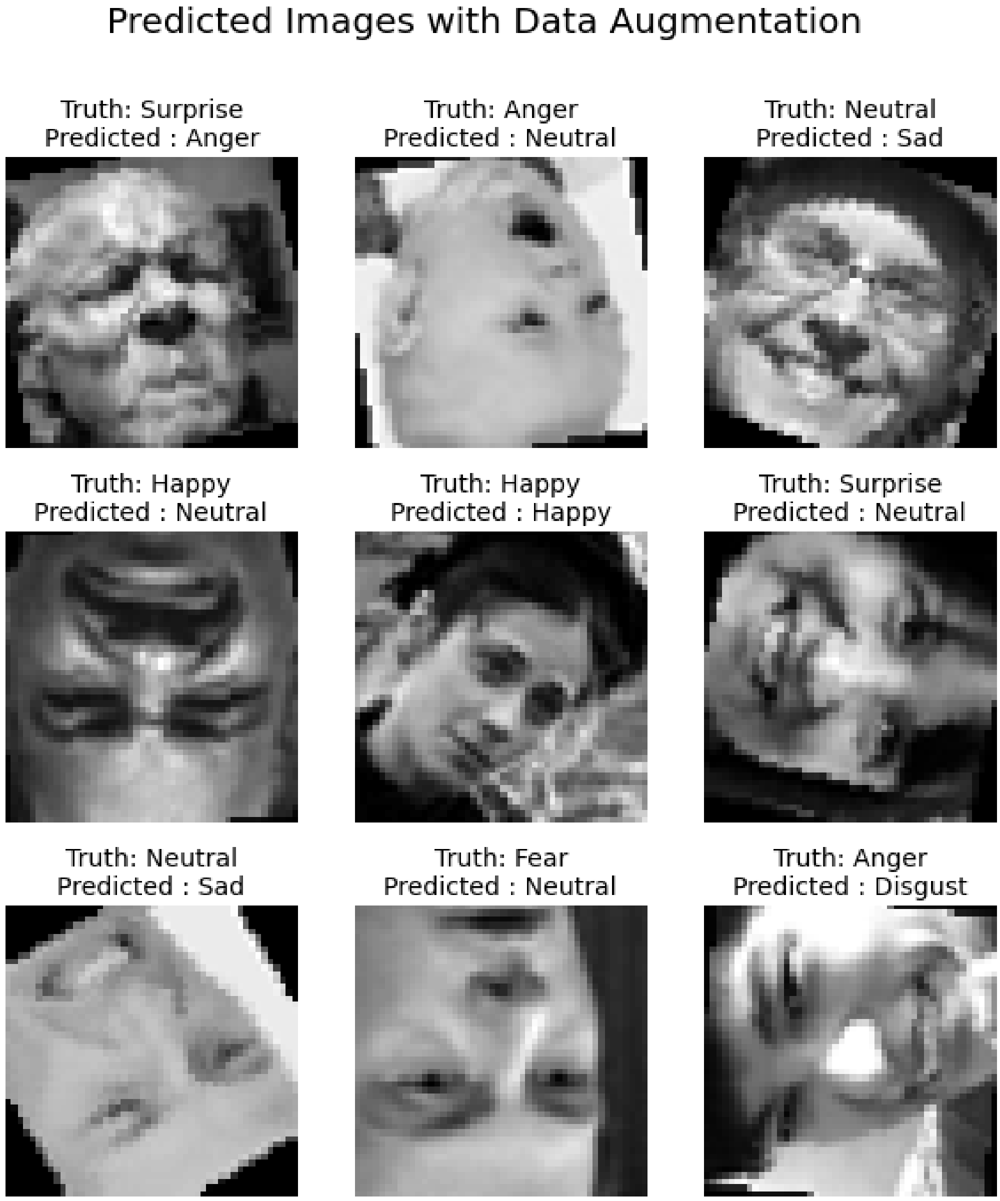
Accuracy	97.1136661503634
F1-score	0.8424999374194159
Recall	0.8426218277135707
Precision	0.8427190998953794

Table 5: Scores testing set with data augmentation

Accuracy	30.51948051948052
F1-score	0.30542444423190773
Recall	0.3051948051948052
Precision	0.30789649549076703

Table 6: Scores per class in testing set with data augmentation

	Precision	**Recall**	**F1-score**	**Support**
Anger	0.21	0.27	0.24	12320
Disgust	0.44	0.44	0.44	12320
Fear	0.18	0.18	0.18	12320
Happy	0.40	0.38	0.39	12320
Sad	0.21	0.18	0.19	12320
Surprise	0.45	0.45	0.45	12320
Neutral	0.27	0.24	0.25	12320
Accuracy			0.31	86240
Macro avg	0.31	0.31	0.31	86240
Weighted avg	0.31	0.31	0.31	86240



This suggests that more investigation is needed to understand the effects of data augmentation on the model's performance and how to optimize the balance between data augmentation and model complexity to improve the accuracy of emotion recognition.

Conclusion

In conclusion, our study presents a CNN model for facial expression recognition that achieves high accuracy on the training set but lower accuracy on the testing set. We identified the imbalanced dataset and the complexity of the model as potential reasons for the overfitting and proposed solutions for future work.

References

Analytics Vidhya. (2021). *Facial Emotion Detection Using CNN*. [online] Available at: https://www.analyticsvidhya.com/blog/2021/11/facial-emotion-detection-using-cnn/.

Bernhard, J. (2018). *Deep Learning With PyTorch*. [online] Medium. Available at: https://medium.com/@josh_2774/deep-learning-with-pytorch-9574e74d17ad [Accessed 17 Apr. 2023].

Bernhard, J. (2018, July 13). *Deep Learning With PyTorch*. Medium. https://medium.com/@josh_2774/deep-learning-with-pytorch-9574e74d17ad

Godoy, D. (2022). *Understanding PyTorch with an example: a step-by-step tutorial*. [online] Medium. Available at: https://towardsdatascience.com/understanding-pytorch-with-an-example-a-step-by-step-tutorial-81fc5f8c4e8e.

Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., ... & Bengio, Y. (2013). Challenges in representation learning: A report on three machine learning contests. In Neural Information Processing: 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, 2013. Proceedings, Part III 20 (pp. 117-124). Springer berlin heidelberg.

kaggle.com. (n.d.). *[SM] Facial Expression Recognition v05*. [online] Available at: https://www.kaggle.com/saramanrriquez/sm-facial-expression-recognition-v05 [Accessed 17 Apr. 2023].

kaggle.com. (n.d.). *Face expression recognition with Deep Learning*. [online] Available at: https://www.kaggle.com/jonathanoheix/face-expression-recognition-with-deep-learning [Accessed 17 Apr. 2023].

Zhu, X., Wu, X. (2004). *Class Noise vs. Attribute Noise: A Quantitative Study*. Artificial Intelligence Review 22, 177–210 . https://doi.org/10.1007/s10462-004-0751-8