ECE 662 - Deep Learning with Python

Project #2 - FER2013

Richard Groves

December 14, 2023

1 Introduction

This project explores the FER2013 dataset [1], a collection of grayscale images of human faces, each labeled with one of seven emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral. The dataset's variations in pose, illumination, and occlusion, make it a challenge for facial emotion recognition models.

The objective of this project was to develop a neural network that can accurately classify these facial expressions. This report details the FER2013 dataset, the neural network design, the training process, and the performance evaluation of the model. The findings are compared with existing literature in the field of emotion recognition using deep learning [2].

2 Background on the FER2013 Dataset

The FER2013 dataset was created by Pierre Luc Carrier and Aaron Courville. This dataset was constructed using the Google image search API and OpenCV face recognition to create a diverse collection of human facial expressions. It is categorized into seven emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral. The images were labeled and categorized into these emotions by Mehdi Mirza and Ian Goodfellow, ensuring a comprehensive representation of human emotions. [1]

3 Data Preparation and Preprocessing

The FER2013 dataset is comprised of 35,887 images. As downloaded from Kaggle.com [3], the dataset was only divided into training and test sets. The training set was split to make a training and a verification set. Each image was resized to 48x48 pixels. The dataset underwent normalization using calculated mean (0.50779873) and standard deviation (0.25495943) values. Data augmentation techniques, including random horizontal flips, rotations, and affine transformations, were applied to enhance the robustness of the training process.

Emotion	Training Set	Validation Set	Test Set	Total
Anger	3190	805	958	4953
Disgust	362	74	111	547
Fear	3318	779	1024	5121
Happiness	5770	1445	1774	8989
Sadness	3859	971	1247	6077
Surprise	2487	684	831	4002
Neutral	3981	984	1233	6198
Total	22967	5742	7178	35887

Table 1: Emotion category counts in the training, validation, and test sets of the FER2013 dataset

4 Approach

4.1 Model Selection and Architecture Testing

In my approach to selecting the optimal model architecture, I conducted extensive experiments with various neural network configurations. The training was executed using Stochastic Gradient Descent (SGD) with an initial learning rate of 0.01. To adaptively adjust the learning rate based on the validation loss and facilitate efficient convergence, I employed a learning rate scheduler, `torch.optim.lr_scheduler.ReduceLROnPlateau`.

I tested multiple configurations of convolutional neural networks (CNNs), exploring different layers, batch normalization, and dropout strategies. Each architecture was subjected to a series of transformations to understand its impact on model performance. The transformations included standard normalization and augmentation techniques like random rotation and affine transformations. This variety in architecture and transformation techniques allowed me to compare the effects of different network depths, complexities, and regularization methods under uniform learning conditions.

The performance of each architecture was evaluated based on the number of epochs run, the final training loss, and the achieved accuracy. The following table summarizes the different network configurations I tested, alongside their corresponding hyperparameters and performance metrics.

Network Configuration	Transformations	Epochs	Training Loss	Accuracy
Conv2d-10-3x3, MaxPool2d-2x2, Conv2d-20-3x3, MaxPool2d-2x2, Linear-720-300, Linear-300-150, Linear-150-7	Normalize	50	0.015543	52.6%
Conv2d-10-3x3, MaxPool2d-2x2, Conv2d-20-3x3, MaxPool2d-2x2, Linear-1000, Linear-250-1000, Linear-1000-7	Normalize	500	0.016786	51.3%
Conv2d-10-3x3, MaxPool2d-2x2, Conv2d-20-3x3, MaxPool2d-2x2, Linear-1000-1000, Linear-250-1000, Linear-100-250, Linear-7-100	Normalize	500	0.017463	51.4%
Conv2d-10-3x3, MaxPool2d-2x2, Conv2d-20-3x3, MaxPool2d-2x2, Conv2d-40-3x3, MaxPool2d-2x2, Linear-360-300, Linear-300-150, Linear-150-7	Normalize	100	0.015525	50.4%
Conv2d-64-3x3, BatchNorm2d-64, ResBlock-64 (2 layers), Conv2d-128-3x3, BatchNorm2d-128, ResBlock-128 (2 layers), Linear-8192-500, Linear-500-7	Normalize	100	0.012914	60.4%
Conv2d-64-3x3, BatchNorm2d-64, Conv2d-64-3x3, BatchNorm2d-64, ResBlock-64 (x2), Conv2d-128-3x3, BatchNorm2d-128, ResBlock-128 (x2), Linear-8192-500, Linear-500-7	Normalize, RandomRotation(10), RandomAffine(degrees=0, translate=(0.1, 0.1))	100	0.004955	65.9%
Conv2d-64-3x3, BatchNorm2d-64, Conv2d-64-3x3, BatchNorm2d-64, ResBlock-64 (x2), Conv2d-128-3x3, BatchNorm2d-128, ResBlock-128 (x2), Linear-8192-500, Dropout-500, Linear-500-7	Normalize, RandomRotation(10), RandomAffine(degrees=0, translate=(0.1, 0.1))	100	0.009753	63.3%

Table 2: Summary of network configurations, hyperparameters, and their performance

5 Experiment

5.1 Training

5.1.1 Data

The FER2013 training dataset was partitioned into training and validation with an 80-20 split respectively. Data preprocessing involved normalization using calculated mean and standard deviation values. The data augmentation techniques random horizontal flips, rotations, and affine transformations were applied to enhance the robustness of the training process.

5.1.2 Model Architecture

The ResNet model I designed combines convolutional layers, batch normalization, and ResBlocks in a deep architecture tailored for complex facial data. Starting with a 64-channel convolutional layer followed by batch normalization, the model includes multiple similar configurations and ResBlocks. As the architecture progresses, it scales up to 128 channels, maintaining the pattern of convolution and normalization. The two final layers are linear. They condense the features into 7 classes for facial expressions.

The architecture of the neural network is as follows:

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 48, 48]	576
BatchNorm2d-2	[-1, 64, 48, 48]	128
Conv2d-3	[-1, 64, 48, 48]	36,864
BatchNorm2d-4	[-1, 64, 48, 48]	128
Conv2d-5	[-1, 64, 48, 48]	36,864
BatchNorm2d-6	[-1, 64, 48, 48]	128
ResBlock-7	[-1, 64, 48, 48]	0
Conv2d-8	[-1, 64, 48, 48]	36,864
BatchNorm2d-9	[-1, 64, 48, 48]	128
Conv2d-10	[-1, 64, 48, 48]	36,864
BatchNorm2d-11	[-1, 64, 48, 48]	128
ResBlock-12	[-1, 64, 48, 48]	0
Conv2d-13	[-1, 128, 24, 24]	73,728
BatchNorm2d-14	[-1, 128, 24, 24]	256
Conv2d-15	[-1, 128, 24, 24]	147,456
BatchNorm2d-16	[-1, 128, 24, 24]	256
Conv2d-17	[-1, 128, 24, 24]	8,192
BatchNorm2d-18	[-1, 128, 24, 24]	256
ResBlock-19	[-1, 128, 24, 24]	0
Conv2d-20	[-1, 128, 24, 24]	147,456
BatchNorm2d-21	[-1, 128, 24, 24]	256
Conv2d-22	[-1, 128, 24, 24]	147,456
BatchNorm2d-23	[-1, 128, 24, 24]	256

```
         ResBlock-24          [-1, 128, 24, 24]               0
           Linear-25                  [-1, 500]       2,304,500
           Linear-26                    [-1, 7]           3,507
================================================================
Total params: 2,982,247
Trainable params: 2,982,247
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.01
Forward/backward pass size (MB): 20.25
Params size (MB): 11.38
Estimated Total Size (MB): 31.64
----------------------------------------------------------------
```

5.1.3 Training Parameters and Procedure

Key training parameters included a learning rate of 0.01, batch size of 100, and a training duration of 100 epochs. The Cross-Entropy Loss function was used as the loss criterion for the model.

5.2 Results

5.2.1 Validation and Testing

The model's performance on the validation set was continuously monitored during training. As seen in Figure 1, the validation accuracy improved until epoch 77. Figure 2 shows the validation loss decreasing until epoch 77.

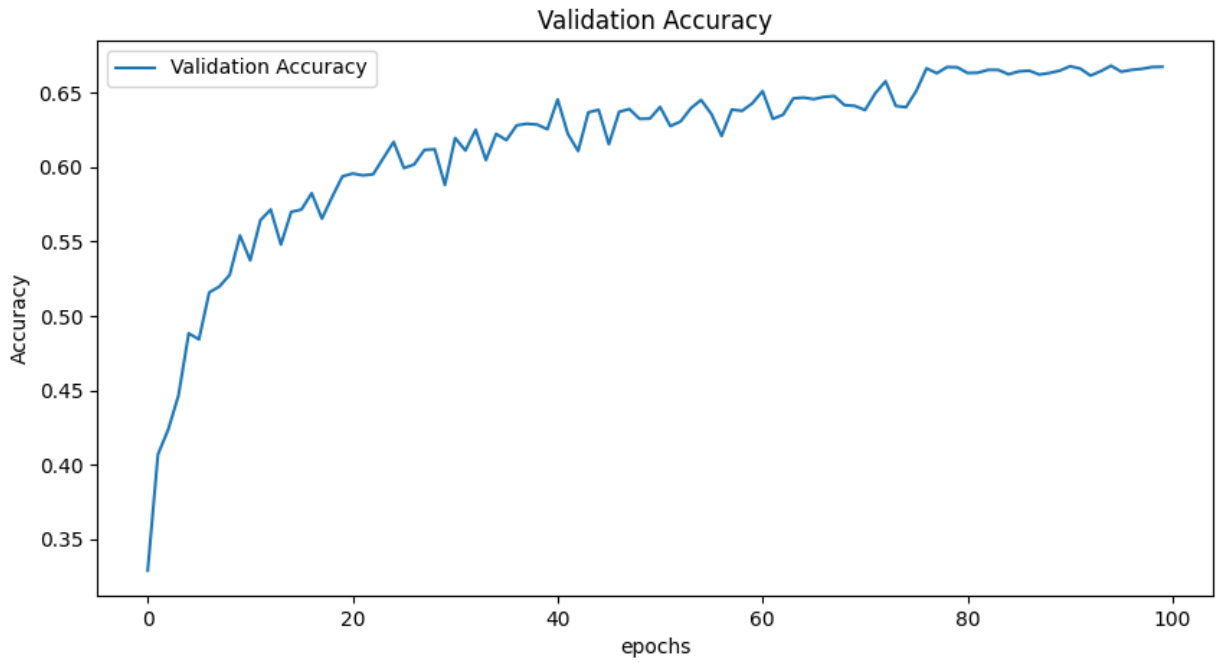

Figure 1: Validation accuracy over 100 epochs.

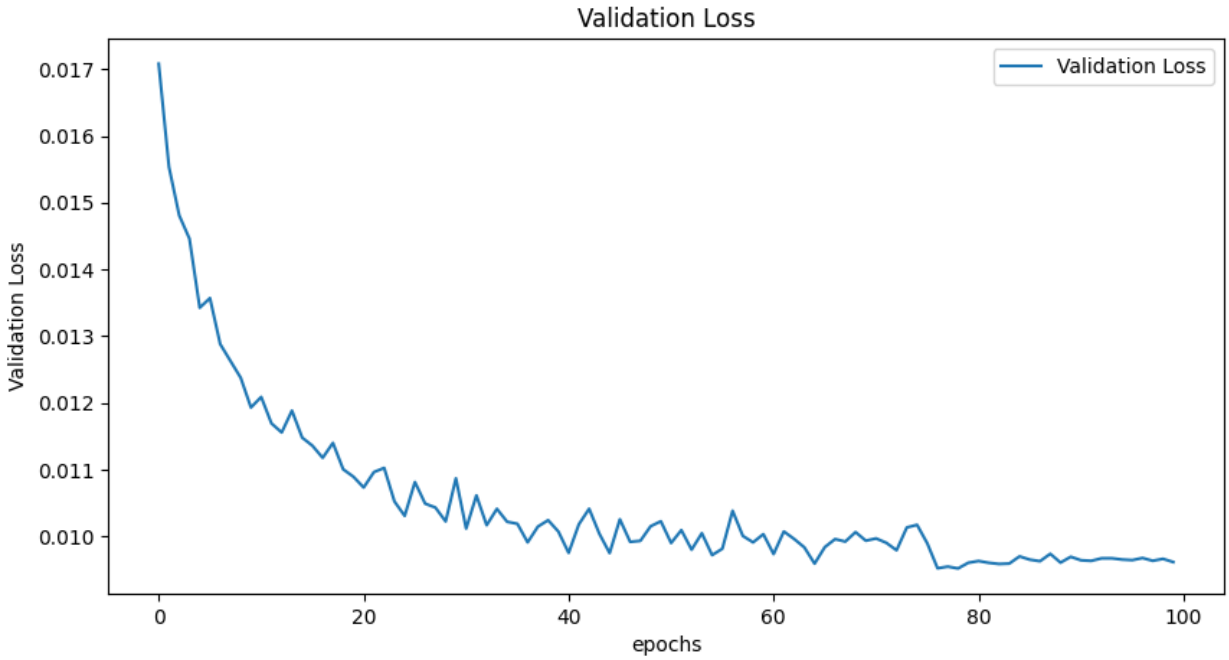

Figure 2: Validation loss over 100 epochs.

5.2.2 Model Predictions and Misclassifications

In the testing stage, there were instances where the model misclassified images. To showcase these misclassifications, three types of images are displayed for each class: the correctly predicted image, the image representing the actual category, and the image that was incorrectly classified. This arrangement aids in highlighting the areas where the model encountered difficulties.

Figure 3: Misclassification example for Angry category. Left: Example of predicted class Neutral. Middle: Incorrectly categorized image. Right: Actual class Angry.

Figure 4: Misclassification example for Disgust category. Left: Example of predicted class Angry. Middle: Incorrectly categorized image. Right: Actual class Disgust.

Figure 5: Misclassification example for Fear category. Left: Example of predicted class Sad. Middle: Incorrectly categorized image. Right: Actual class Fear.

Figure 6: Misclassification example for Happy category. Left: Example of predicted class Angry. Middle: Incorrectly categorized image. Right: Actual class Happy.

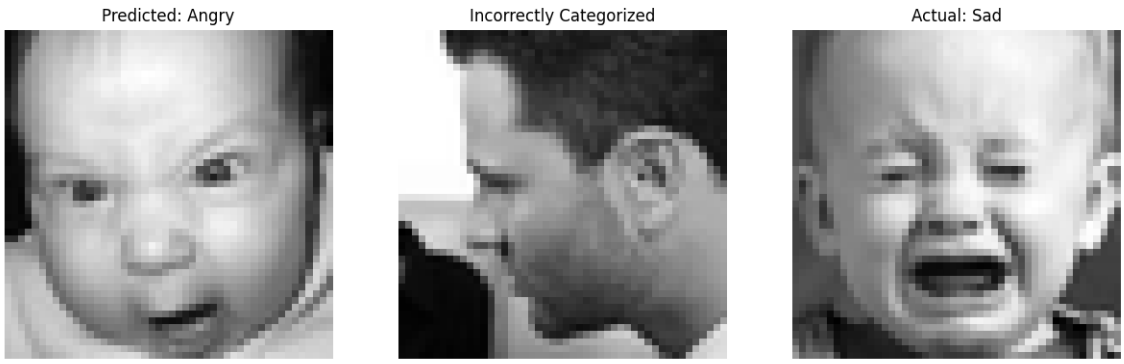

Figure 7: Misclassification example for Sad category. Left: Example of predicted class Angry. Middle: Incorrectly categorized image. Right: Actual class Sad.

Figure 8: Misclassification example for Surprise category. Left: Example of predicted class Happy. Middle: Incorrectly categorized image. Right: Actual class Surprise.

Figure 9: Misclassification example for Neutral category. Left: Example of predicted class Surprise. Middle: Incorrectly categorized image. Right: Actual class Neutral.

6 Conclusion

Human-level accuracy, representative of the average accuracy that a human can achieve on this task, is estimated at 65% with a 5% margin of error [1]. My ResNet model achieved an accuracy of 65.9%.

The Ensemble method, which combines the strengths of multiple models, reached an accuracy of 75.8% [2]. This underscores the potential of combining diverse models and approaches to handle the complexity of human emotional expression.

Table 3: Network parameters and results on FER2013 [2].

Model	**Accuracy**
(Human-level)	65%±5%
Baseline	64.0%
Five-Layer Model	66.3%
VGG16	70.2%
SeNet50	72.5%
ResNet50	73.2%
Ensemble	75.8%
My ResNet	65.9%

References

[1] Ian J Goodfellow et al. "Challenges in Representation Learning: A report on three machine learning contests". In: *arXiv preprint arXiv:1307.0414* (2013). URL: `https://arxiv.org/abs/1307.0414`.

[2] Amil Khanzada, Charles Bai, and Ferhat Turker Celepcikay. *Facial Expression Recognition with Deep Learning: Improving on the State of the Art and Applying to the Real World*. Stanford University - CS230 Deep Learning. Available: `https://github.com/amilkh/cs230-fer`. 2018.

[3] M. Sambare. *FER2013 Facial Expression Recognition Challenge Dataset*. Kaggle. URL: `https://www.kaggle.com/datasets/msambare/fer2013`.