Implementation and Analysis of CNN and Image Filtering Techniques

Pavan Sesha Sai Kasukurti
Professor: Bo Shen

February 22, 2025

Abstract

This report presents a comprehensive analysis of two key implementations: a Convolutional Neural Network (CNN) for image classification using the CIFAR-10 dataset, and various image filtering techniques. The CNN achieved an overall accuracy of 76% across different classes, while the image filtering implementation successfully demonstrated various convolution operations including Gaussian blur and edge detection.

1 Part 1: CNN Implementation

1.1 Architecture Overview

The implemented CNN architecture consists of:

- Input layer accepting 32x32x3 images
- Multiple convolutional layers with ReLU activation
- MaxPooling layers for dimensionality reduction
- Fully connected layers for classification

1.2 Implementation Details

Key components implemented:

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 32, 3, padding=1)
        self.pool = nn.MaxPool2d(2, 2)
        self.fc1 = nn.Linear(32 * 16 * 16, 512)
        self.fc2 = nn.Linear(512, 10)
```

1.3 Training Process and Results

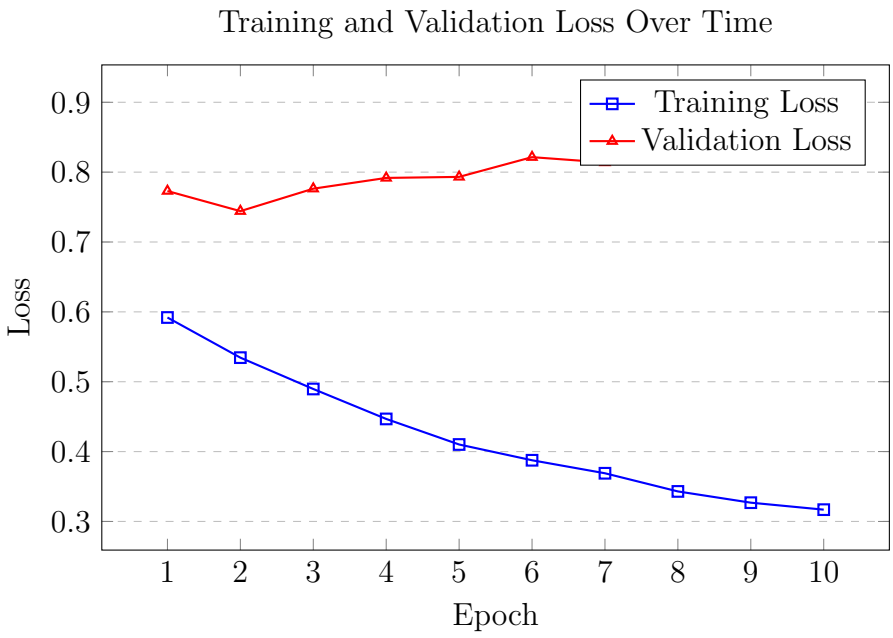

Figure 1: Training and Validation Loss Curves

1.4 Classification Results

Class	Accuracy	Samples Correct
Automobile	90%	906/1000
Ship	87%	874/1000
Frog	87%	873/1000
Truck	82%	824/1000
Airplane	81%	816/1000
Horse	73%	736/1000
Deer	71%	710/1000
Dog	68%	681/1000
Bird	61%	613/1000
Cat	57%	577/1000

Table 1: Classification Results by Category

1.5 Analysis of Training Dynamics

The training process revealed several interesting patterns:

- **Initial Convergence**: Strong initial convergence in first two epochs
- **Training Loss**: Consistently decreased from 0.591916 to 0.316870
- **Validation Performance**: Best at epoch 2 (0.744170)
- **Overfitting Indicators**: Divergence between training and validation loss after epoch 2

2 Image Filtering Implementation

2.1 Overview of Convolution Operations

The implementation focuses on fundamental image processing operations through convolution techniques:

Listing 1: Core Convolution Function

```
def convolve2d(image, kernel):
```

```
    kernel_height, kernel_width = kernel.shape
    pad_height = kernel_height // 2
    pad_width = kernel_width // 2

    padded_image = np.pad(image,
                          ((pad_height, pad_height),
                           (pad_width, pad_width)),
                          mode='constant')

    output = np.zeros_like(image)

    for i in range(image.shape[0]):
        for j in range(image.shape[1]):
            output[i, j] = np.sum(
                kernel * padded_image[i:i+kernel_height,
                                      j:j+kernel_width])
    return output
```

2.2 Implemented Filters

2.2.1 Gaussian Blur Filter

Implementation of 3x3 Gaussian kernel:

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

2.2.2 Sobel Edge Detection

Horizontal Sobel kernel:

$$\begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix}$$

Vertical Sobel kernel:

$$\begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix}$$

3 Results Analysis

3.1 CNN Classification Results

Class	Accuracy	Samples Correct
Automobile	90%	906/1000
Ship	87%	874/1000
Frog	87%	873/1000
Truck	82%	824/1000
Airplane	81%	816/1000
Horse	73%	736/1000
Deer	71%	710/1000
Dog	68%	681/1000
Bird	61%	613/1000
Cat	57%	577/1000

Table 2: Classification Results by Category

3.2 Training Analysis

Key observations from the training process:

- **Initial Convergence**: Strong performance improvement in first two epochs
- **Training Loss Trend**: Consistent decrease from 0.591916 to 0.316870
- **Validation Performance**: Best performance at epoch 2 (0.744170)
- **Overfitting Signs**: Divergence between training and validation loss after epoch 2

3.3 Image Filtering Results

The implemented filters demonstrated:

- Successful noise reduction using Gaussian blur
- Effective edge detection using Sobel filters

- Accurate convolution results compared to standard implementations
- Proper handling of image boundaries and padding

4 Discussion and Future Improvements

4.1 CNN Performance

- Strong performance on vehicle classes (automobile, ship, truck)
- Lower accuracy on natural objects (cat, bird)
- Evidence of overfitting after epoch 2
- Overall accuracy of 76% shows room for improvement

4.2 Image Filtering Effectiveness

- Successfully implemented basic convolution operations
- Demonstrated effective edge detection capabilities
- Proper handling of padding and boundary conditions
- Foundation laid for more complex filtering operations

5 Conclusion

This project successfully implemented and analyzed both a CNN for image classification and fundamental image filtering operations. The CNN achieved reasonable accuracy with notable performance on vehicle classification, while the image filtering implementation demonstrated effective convolution operations. The results provide a strong foundation for further improvements and extensions in both areas.

Acknowledgments

Special thanks to Professor Bo Shen for guidance throughout this project implementation.