CIFAR-100 Classification with Convolutional Neural Networks

Mike Nguyen

Contents

1 Introduction

This report presents the results of training convolutional neural networks (CNNs) on the CIFAR-100 dataset to perform image classification. The primary goal was to explore different activation functions, optimizers, and hyperparameters to identify the best-performing models. By utilizing hyperparameter tuning to optimize model performance and compare results with existing benchmarks.

2 Model Architecture

Employed a consistent CNN architecture for all experiments to ensure a fair comparison among different configurations.

2.1 CNN Structure

1. **Convolutional Layer 1**:
    - Input Channels: 3 (RGB images)
    - Output Channels: 32
    - Kernel Size: 3×3
    - Padding: 1
2. **Activation Function**:
    - Variable (ReLU, Leaky ReLU, or ELU)
3. **Max Pooling Layer 1**:
    - Kernel Size: 2×2
    - Stride: 2
4. **Convolutional Layer 2**:
    - Input Channels: 32
    - Output Channels: 64
    - Kernel Size: 3×3
    - Padding: 1
5. **Activation Function**:
    - Same as above
6. **Max Pooling Layer 2**:
    - Kernel Size: 2×2
    - Stride: 2
7. **Fully Connected Layer**:
    - Input Features: $64 \times 8 \times 8 = 4,096$
    - Output Features: 100 (number of classes)

2.2 Total Number of Parameters

The total number of trainable parameters in the EasyCNN model is calculated as follows:

- **Convolutional Layer 1**:

$$\text{Parameters} = (3 \times 3 \times 3 \times 32) + 32 = 896$$

- **Convolutional Layer 2**:

$$\text{Parameters} = (3 \times 3 \times 32 \times 64) + 64 = 18,496$$

- **Fully Connected Layer**:

$$\text{Parameters} = (4,096 \times 100) + 100 = 409,700$$

- **Total Parameters**:

$$\text{Total} = 896 + 18,496 + 409,700 = 429,092$$

Each model has approximately **429,092 trainable parameters**.

3 Hyperparameter Tuning

To optimize model performance, conducting hyperparameter tuning using Ray Tune. The following hyperparameters were explored:

- **Activation Functions**: ReLU, Leaky ReLU, ELU
- **Optimizers**: SGD, Adam, RMSprop
- **Learning Rates (lr)**: Log-uniform distribution between 1×10^{-4} and 1×10^{-2}
- **Batch Sizes**: 64, 128, 256

3.1 Tuning Results

After running 10 trials, the results are summarized in Table 1.

4 Top Three Configurations

Based on validation accuracy, the top three configurations identified were:

4.1 Configuration 1

- **Activation Function**: ELU
- **Optimizer**: RMSprop
- **Learning Rate**: 3.19×10^{-4}
- **Batch Size**: 64
- **Validation Accuracy**: 37.99%
- **Test Accuracy**: **35.42%**

4.2 Configuration 2

- **Activation Function**: Leaky ReLU
- **Optimizer**: RMSprop
- **Learning Rate**: 1.10×10^{-3}
- **Batch Size**: 64
- **Validation Accuracy**: 35.20%
- **Test Accuracy**: **35.41%**

4.3 Configuration 3

- **Activation Function**: ReLU
- **Optimizer**: RMSprop
- **Learning Rate**: 2.35×10^{-3}
- **Batch Size**: 128
- **Validation Accuracy**: 32.93%
- **Test Accuracy**: **28.45%**

Activation	Optimizer	Learning Rate	Batch Size	Validation Accuracy
ReLU	SGD	4.04×10^{-4}	128	20.53%
ELU	RMSprop	3.19×10^{-4}	64	37.99%
ELU	RMSprop	2.65×10^{-3}	64	28.40%
Leaky ReLU	SGD	1.04×10^{-3}	128	28.66%
ReLU	RMSprop	2.35×10^{-3}	128	32.93%
Leaky ReLU	Adam	3.36×10^{-3}	64	30.75%
ReLU	RMSprop	2.16×10^{-3}	256	28.60%
ELU	SGD	1.62×10^{-3}	256	28.30%
ReLU	Adam	3.78×10^{-3}	256	32.16%
Leaky ReLU	RMSprop	1.10×10^{-3}	64	35.20%

Table 1: Hyperparameter Tuning Results

5 Retraining and Evaluation

Each of the top three models was retrained on the full training dataset (combining sub-training and validation sets) and evaluated on the test dataset over 50 epochs.

5.1 Configuration 1

- **Activation Function**: ELU
- **Optimizer**: RMSprop
- **Learning Rate**: 3.19×10^{-4}
- **Batch Size**: 64
- **Test Accuracy**: 35.42%

5.1.1 Benchmarking Results

- **Epoch 1**: Train Loss = 3.2729, Train Accuracy = 23.48%
- **Epoch 25**: Train Loss = 0.1189, Train Accuracy = 97.25%
- **Epoch 50**: Train Loss = 0.0266, Train Accuracy = 99.40%

5.1.2 Training Plots

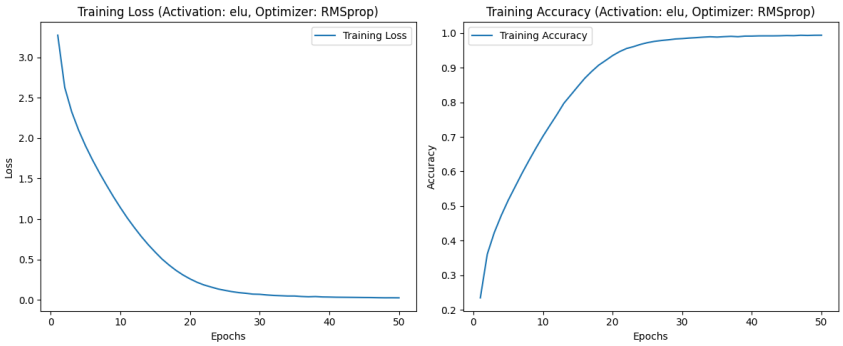

Figure 1: Training Loss and Accuracy for Configuration 1

5.2 Configuration 2

- **Activation Function**: Leaky ReLU
- **Optimizer**: RMSprop
- **Learning Rate**: 1.10×10^{-3}
- **Batch Size**: 64
- **Test Accuracy**: 35.41%

5.2.1 Benchmarking Results

- **Epoch 1**: Train Loss = 3.4217, Train Accuracy = 20.56%
- **Epoch 25**: Train Loss = 0.0598, Train Accuracy = 98.12%
- **Epoch 50**: Train Loss = 0.0448, Train Accuracy = 98.74%

5.2.2 Training Plots

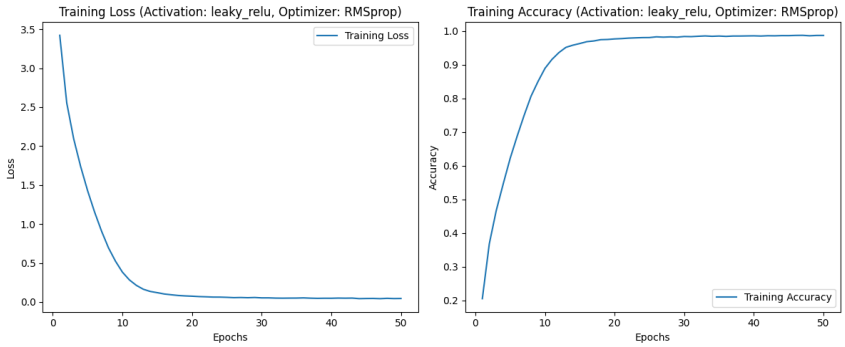

Figure 2: Training Loss and Accuracy for Configuration 2

5.3 Configuration 3

- **Activation Function**: ReLU
- **Optimizer**: RMSprop
- **Learning Rate**: 2.35×10^{-3}
- **Batch Size**: 128
- **Test Accuracy**: 28.45%

5.3.1 Benchmarking Results

- **Epoch 1**: Train Loss = 3.7492, Train Accuracy = 16.72%
- **Epoch 25**: Train Loss = 0.6500, Train Accuracy = 80.41%
- **Epoch 50**: Train Loss = 0.2746, Train Accuracy = 90.86%

5.3.2 Training Plots

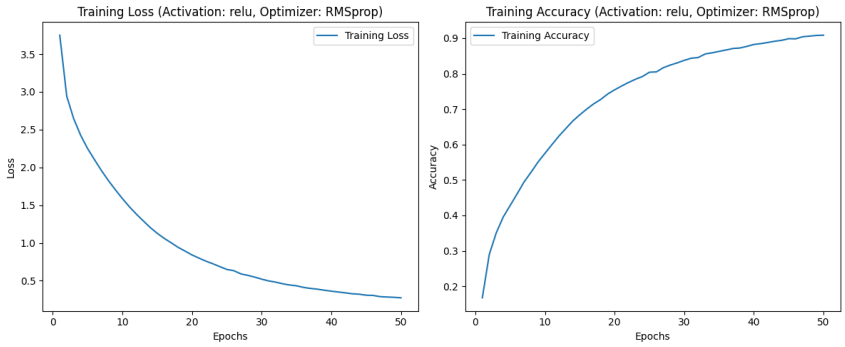

Figure 3: Training Loss and Accuracy for Configuration 3

6 Comparison with Other CIFAR-100 Rankings

To contextualize our results, we compared our top model's test accuracy with other models ranked on CIFAR-100:

Rank	**Model**	**Test Accuracy (%)**
38	CNN39	42.64
39	CNN36	36.07
40	CNN37	35.05

Table 2: Selected CIFAR-100 Rankings

The best model achieved a test accuracy of 35.42%, which is comparable to the performance of CNN37 (35.05%) at rank 40. However, it falls short of CNN36 (36.07%) and CNN39 (42.64%).