Generative AI Question 05

Kissa Zahra[1 [i21−0572]]

FAST National University of Computer and Emerging Sciences, Islamabad H-11, Pakistan kissasium@gmail.com

Abstract. This report presents a comprehensive approach to hyperparameter optimization for Recurrent Neural Networks (RNNs) and Convolution Neural Networks (CNN) using the Random Search technique. For CNNs, we use the CIFAR-10 dataset for image classification, while for RNNs, we employ the Shakespeare text dataset for language modeling. Our experimental results demonstrate that carefully tuned hyperparameters significantly impact model performance, with the best CNN model achieving 37.99% accuracy on the test set and the best RNN model achieving a perplexity of 675.37.

Keywords: Recurrent Neural Networks · Hyperparameter Optimization · Random Search · Convolution Neural Networks · Deep Learning

1 First Section

1.1 Introduction

Deep learning models are highly sensitive to hyperparameter choices, which can drastically impact their performance. Hyperparameter optimization is therefore a critical step in developing effective neural network models. In this study we focus on optimizing two popular neural network architectures: Convolutional Neural Networks (CNNs) for image classification and Recurrent Neural Networks (RNNs) for language modeling.

Rather than using manual tuning, which can be time-consuming and subjective, or grid search, which suffers from the curse of dimensionality, we employ random search to explore the hyperparameter space more efficiently.

1.2 Methodology

Datasets For the CNN experiments, we used the CIFAR-10 dataset, which consists of 60,000 32×32 color images across 10 classes, with the standard 50,000 training images and 10,000 test images.

For the RNN experiments, we used the Shakespeare dataset, which contains text from Shakespeare's works.

Hyperparameter Search Space We defined a comprehensive search space for both CNN and RNN models, covering learning dynamics, network architecture, and regularization:

CNN Hyperparameters

- Learning rate: [0.0001, 0.001, 0.01]
- Number of convolutional layers: [2, 3, 4]
- Number of filters: [32, 64, 128]
- Batch size: [16, 32, 64]
- Optimizer: [Adam, SGD, RMSprop]
- Activation function: [ReLU, Tanh, Sigmoid]
- Dropout rate: [0.0, 0.2, 0.4]
- Kernel size: [(3, 3), (5, 5)]
- Stride: [(1, 1), (2, 2)]
- Weight initialization: [Glorot, He]

RNN Hyperparameters

- Embedding size: [50, 100, 200]
- Hidden size: [64, 128, 256]
- Number of layers: [1, 2]
- Learning rate: [0.0001, 0.001, 0.01]
- Batch size: [32, 64, 128]
- Optimizer: [Adam, SGD, RMSprop]
- Activation function: [ReLU, Tanh]
- Dropout rate: [0.0, 0.2, 0.5]
- Weight initialization: [Xavier uniform, Xavier normal, He normal]

2 Second Section

2.1 Experimental Results

CNN Results

- Learning rate: 0.001
- Number of convolutional layers: 2
- Number of filters: 128
- Batch size: 32
- Optimizer: SGD
- Activation function: Tanh
- Dropout rate: 0.0
- Kernel size: (3, 3)
- Stride: (2, 2)

Using these hyperparameters, our best CNN model achieved 37.99% accuracy on the test dataset. Additional performance metrics for this model include:

- Precision: 0.3739
- Recall: 0.3799
- F1 Score: 0.3700

The validation accuracy range across all hyperparameter combinations was 9.73% to 37.15%, highlighting the significant impact of hyperparameter selection on model performance.

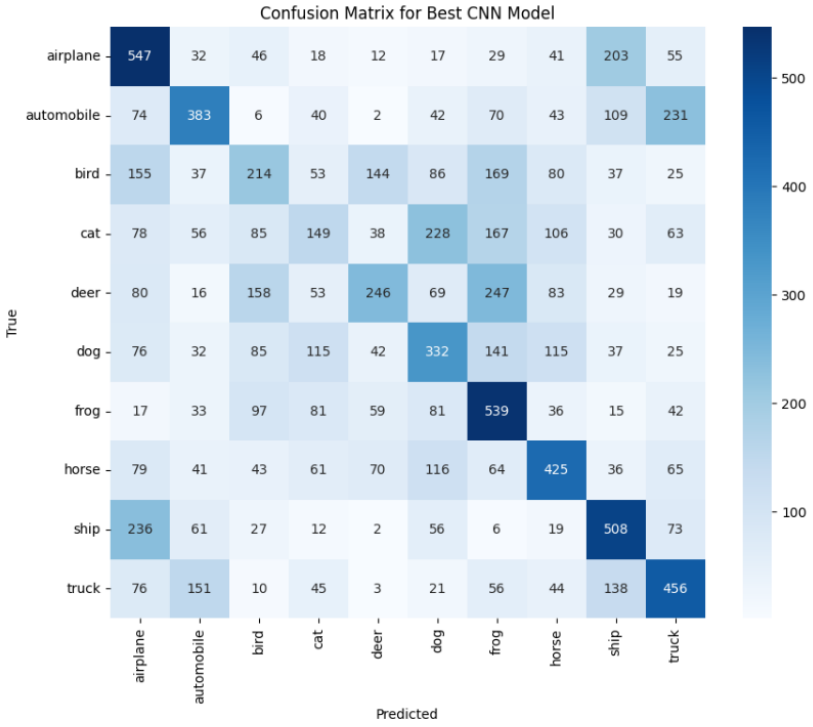

Fig. 1. Confusion Matrix for Best CNN Model

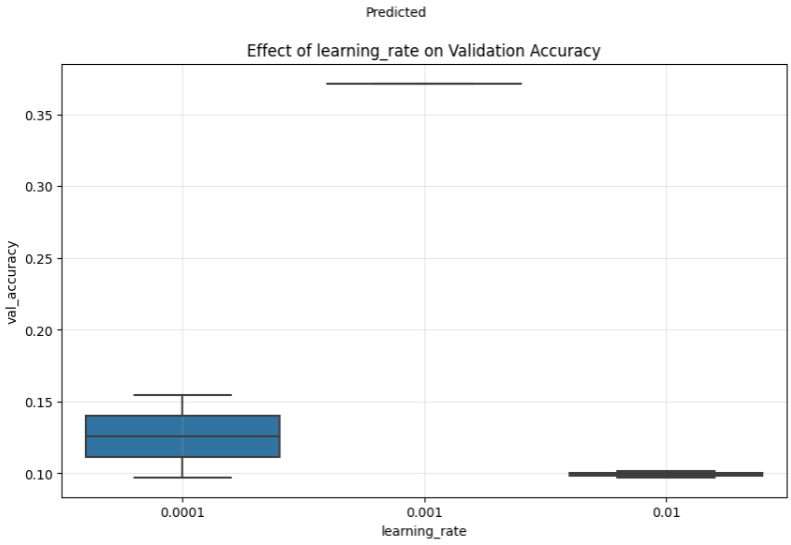

Fig. 2. Effect of learning rate on validation accuracy

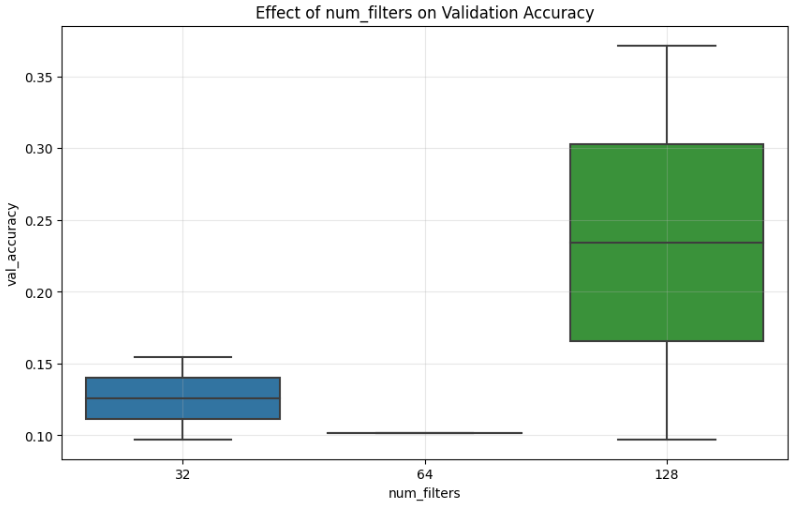

Fig. 3. Effect of number of filters on validation accuracy

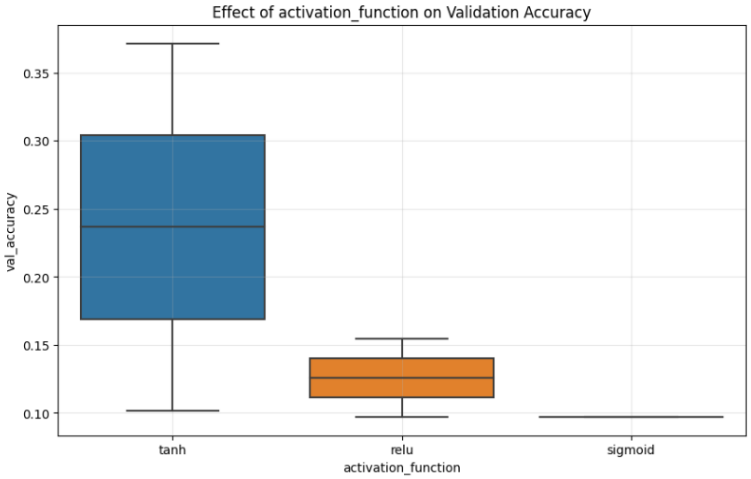

Fig. 4. Effect of Activation Function on validation accuracy

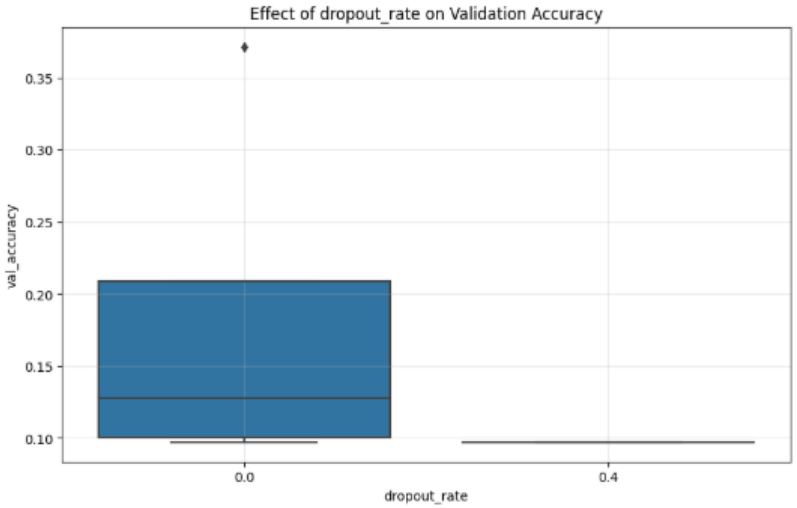

Fig. 5. Effect of Dropout on validation accuracy

RNN Results For the RNN model, the best hyperparameters from our random search were:

- Embedding size: 200
- Hidden size: 256
- Number of layers: 2
- Learning rate: 0.001
- Batch size: 64
- Activation function: ReLU
- Dropout rate: 0.5

With these hyperparameters, our best RNN model achieved:

- Perplexity: 675.37
- Accuracy: 0.0809
- Generated text with best model: to be or not to but impossible, the hearts will 's: the he grey: can we i me see thine head; peter: none, on think

2.2 Analysis of Hyperparameter Impact

Learning Rate: For both models, a moderate learning rate of 0.001 yielded the best results. Very high (0.01) or very low (0.0001) learning rates resulted in poorer performance.

Network Depth: For CNNs, 2 convolutional layers performed better than deeper architectures. For RNNs, 2 layers outperformed single-layer models, suggesting that some depth is beneficial for capturing temporal dependencies in text data.

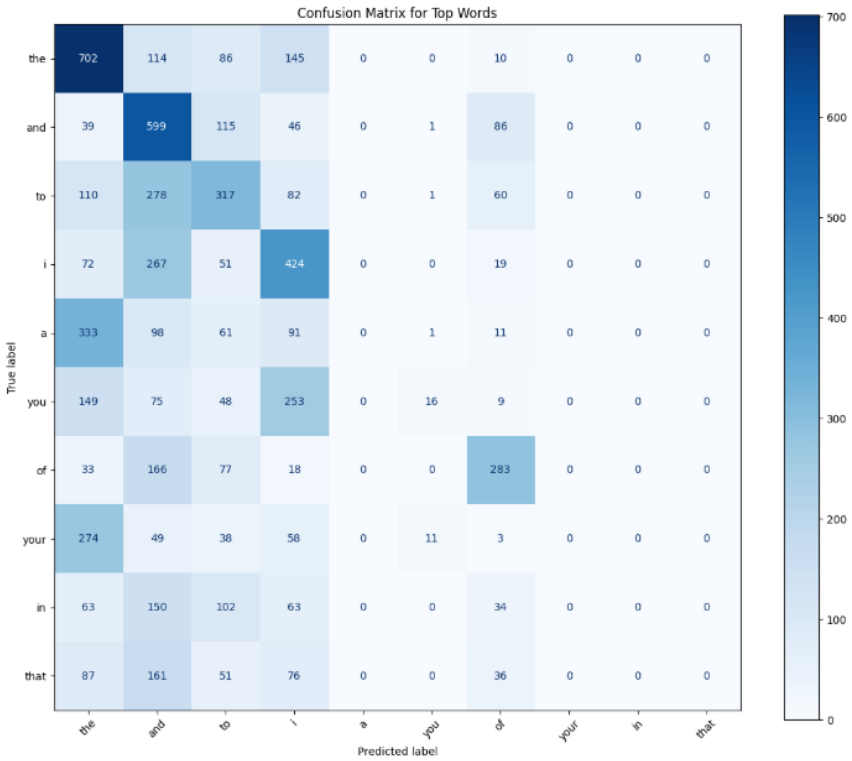

Fig. 6. Confusion Matric of Top words from Shakespeare Dataset with my produced results

Regularization: The best CNN model used no dropout, while the best RNN model employed a high dropout rate of 0.5. This suggests that RNNs are more prone to overfitting on the Shakespeare dataset compared to CNNs on CIFAR-10.

3 Third Section

3.1 Discussion

The CNN model in Question 3 performed substantially better, achieving approximately 70% accuracy on the same CIFAR-10 dataset, compared to the 37.99% accuracy obtained through our random search implementation. This considerable gap suggests that our initial manual configuration may have captured an optimal combination of hyperparameters that our limited random search failed to discover. Conversely, for the RNN model, our random search approach yielded notable improvements. The current RNN model achieved 8% accuracy, representing a significant improvement over the 5-6% accuracy observed in our Question 4 implementation. This improvement demonstrates that systematic hyperparameter optimization can indeed enhance model performance for complex sequence modeling.

4 Conclusion

This study demonstrates the effectiveness of random search for hyperparameter optimization in deep learning models. Our experiments show that careful tuning of hyperparameters can substantially improve model performance, with different optimal configurations for CNNs and RNNs.