Comparative Analysis of Machine Learning Techniques for Image Classification on the CIFAR-10 Dataset

Yixiao Chen YCHEN3@UNC.EDU **Shihang Gui** SHIHANG@AD.UNC.EDU **Boyang Mu** BMU5@UNC.EDU **Danny Wang** DWANG1@UNC.EDU

Abstract

Image classification has many uses in real-world applications ranging from autonomous driving to content categorization. The CIFAR-10 dataset, known for its diversity across ten object categories, provides a challenging yet tractable benchmark for image classification. This paper explores the application of four distinct machine learning models—Logistic Regression, K-Nearest Neighbors (KNN), Random Forest, and Convolutional Neural Networks (CNN)—on the CIFAR-10 dataset to compare their performance.

1. INTRODUCTION

In the dynamic field of machine learning, image classification continues to be a fundamental and complex challenge, with applications extending across diverse domains such as autonomous vehicles, medical diagnostics, security systems, and more. This task forms the core of many computer vision systems, requiring robust and efficient algorithms capable of high accuracy and fast processing times. The CIFAR-10 dataset, consisting of 60,000 small 32x32 pixel images in 10 distinct classes, serves as a crucial benchmark for evaluating the performance of various machine learning models. This project explores the effectiveness of four different algorithms: Logistic Regression, K-Nearest Neighbors (KNN), Random Forests, and Convolutional Neural Networks (CNN). These models range from ensemble methods known for their capacity to capture complex patterns through multiple decision trees to the intricate layers of deep learning architectures designed for image recognition tasks. Our code is available at https://github.com/cyixiao/Analysis-of-ML-CIFAR-10.

2. DATASET

2.1. *Description*

The CIFAR-10 dataset is a subset of the much larger "80 million tiny images dataset", assembled by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. This is one of the most widely used datasets for machine learning research. It comprises 60,000 color images of 32x32 resolution, uniformly distributed across ten distinct classes, making it a standard benchmark in machine learning for image recognition tasks. The CIFAR-10 dataset encompasses a diverse range of categories, each representing different real-world objects. The ten classes are as follows: Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship, and Truck. CIFAR-10 is structured into six batches, where each batch contains 10,000 images. Specifically, the dataset is split into five training batches and one test batch, ensuring comprehensive coverage for model training and evaluation.

2.2. *Processing*

For models such as Logistic Regression and KNN, the image data was flattened. This means converting each 32x32x3 image into a 3072-dimensional vector, which is required as these models do not inherently process two-dimensional image data. This step is crucial for utilizing traditional machine learning algorithms that expect input data in a flat, vector form. For the CNN models, the class labels were encoded using a one-hot encoding scheme. This transforms categorical integer labels into a binary matrix representation to be used in conjunction with categorical cross-entropy as the loss function during the training of neural networks.

3. METHODOLOGY

3.1. *Logistic Regression*

Logistic Regression is primarily known for its effectiveness in binary classification tasks, where the objective is to determine the likelihood of outcomes that are typically represented as 0 or 1. However, applying it to object recognition problems, particularly those involving multiple classes such as CIFAR-10, presents unique challenges. CIFAR-10 includes 60,000 images categorized into ten classes, which complicates the use of Logistic Regression due to its inherent design for binary outcomes. Therefore, as part of the training process, we first reshape each image into a flat vector to adapt the input for the model.

After training the Logistic Regression model on this flattened dataset, the model's performance was evaluated using key metrics such as accuracy and the classification report. Despite the complexity and multi-class nature of the CIFAR-10 dataset, the Logistic Regression model achieved an overall accuracy of approximately 35% on the test data.

The classification report showed that performance varied a lot across different classes. The precision ranged from a high of 56% for *ship* to a low of 19% for *bird*. This shows that while the logistic regression model did well with some classes, it had difficulties with others. The recall rate was highest for *bird* at 67%, which means the model identified most instances of this class but also made many incorrect predictions. These variations in precision and recall highlight the logistic regression model's challenges when used for multi-class classification tasks.

Overall, the Logistic Regression model managed to achieve a moderate level of accuracy on the CIFAR-10 test dataset. Although it demonstrated some strengths in identifying some classes, the overall performance was hindered by the constraints in dealing with flattened data and multiple classes. This outcome underscores the necessity of choosing more suitable algorithm models for multi-class image classification tasks such as this one.

3.2. *K-Nearest Neighbors*

The K-Nearest Neighbors (KNN) algorithm is one of the most intuitive methods used for both classification and regression tasks in machine learning. One of the distinguishing features of KNN is its simplicity and ease of implementation. It doesn't require any assumptions about the underlying data distribution, making it particularly useful when the data is non-linear or lacks a clear separation between classes. At its core, KNN operates on the principle of similarity. It assumes that similar data points tend to belong to the same class or have similar characteristics. The algorithm calculates the distance between the query instance and all the training samples to identify the K nearest neighbors. These neighbors then "vote" to determine the class of the query instance in classification tasks or contribute to the prediction in regression tasks.

In our initial tests with KNN on the CIFAR-10 dataset, we experimented with different values of K, ranging from 3 to 8. We discovered that accuracy was highest at $K = 4$ and decreased as K increased. Despite this optimization, the best accuracy we achieved was around 30%, which is relatively low. This prompted us to investigate the reasons behind the poor performance. We identified that the main challenge was the high dimensionality of the input data, consisting of 3072 dimensions that correspond to the pixels and colors of 32x32 RGB images. This high dimensionality can undermine the effectiveness of the KNN algorithm, a problem commonly known as the "curse of dimensionality." To address this, we applied Principal Component Analysis (PCA), a method used to reduce the dimensions of data by focusing on the most important features. We tried to maintain 98%, 95%, and 90% of the most significant features, respectively. However, the performance improvements were minimal after dimensionality reduction. This result suggests that each pixel and color channel carries important information, and reducing any of these features can lead to significant information loss.

Considering these challenges, we conclude that KNN may not be the most suitable model for image recognition tasks on the CIFAR-10 dataset. This experience prompted us to explore other models that can better handle the complexity of image data.

3.3. *Random Forest*

The Random Forest algorithm utilizes an ensemble of decision trees to enhance prediction accuracy and robustness by aggregating decisions from multiple models. This method operates on subsets of features to

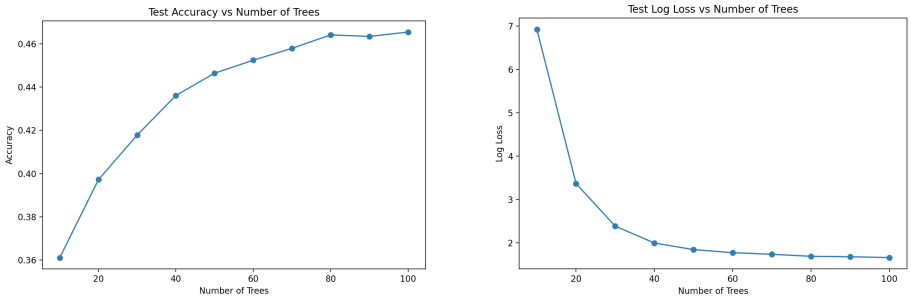

FIG. 1.— On the left, Test Accuracy vs Number of Trees, showing how accuracy in the Random Forest model improves as the number of trees increases. On the right, Test Log Loss vs Number of Trees, illustrates the decrease in log loss with more trees.

make individual decisions, which are then combined to form a final decision, aiming to improve accuracy and prevent over-fitting.

In our experiment, the Random Forest classifier was initially trained with default parameters to establish a baseline performance. Subsequently, we experimented with increasing the number of decision trees from 10 to 100 in increments of 10 to observe the impact on model accuracy and log loss. The experiment results, illustrated in Figure 1, show a clear trend: as the number of trees in the forest increases, the accuracy improves, and the log loss decreases. This outcome highlights Random Forest's strength in handling overfitting — more trees in the model reduce variance without significantly increasing bias, enhancing the model's generalization capabilities on unseen data.

RFs are known for their computational efficiency, which is particularly advantageous when computational resources are limited. They do not require input transformation, simplifying the implementation process. However, RFs do not inherently process spatial relationships within image data, which may limit their effectiveness for more complex image classification tasks. The Random Forest algorithm proved effective for the image classification task, demonstrating substantial improvements in accuracy with an increased number of trees. However, the complexity and computational cost also increase with more trees. Therefore, there is a trade-off between performance and computational efficiency that needs to be balanced according to the specific requirements of the application. Nonetheless, the Random Forest algorithm is a powerful tool for image classification tasks, providing high accuracy and robustness against overfitting.

3.4. *Convolutional Neural Network*

When developing a convolutional neural network (CNN) for the CIFAR-10 dataset, considering the small size of the 32x32 pixel images, we adopted a special structural strategy to enhance model performance. The traditional strategy of alternating convolution layers and pooling layers performs well in large-size images, but for smaller images, this approach may lead to rapid loss of spatial information. To overcome this, our network emulates the design of VGG, using two consecutive convolutional layers before each pooling operation. This structure not only delays the reduction in feature dimensions but also allows the network to capture more spatial features through continuous convolutional operations

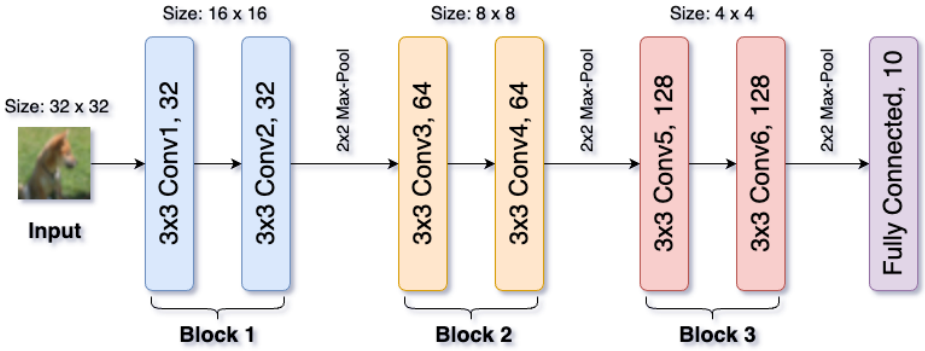

FIG. 2.— Schematic of the CNN architecture showing three blocks of convolutional and pooling layers followed by a fully connected layer for classification.

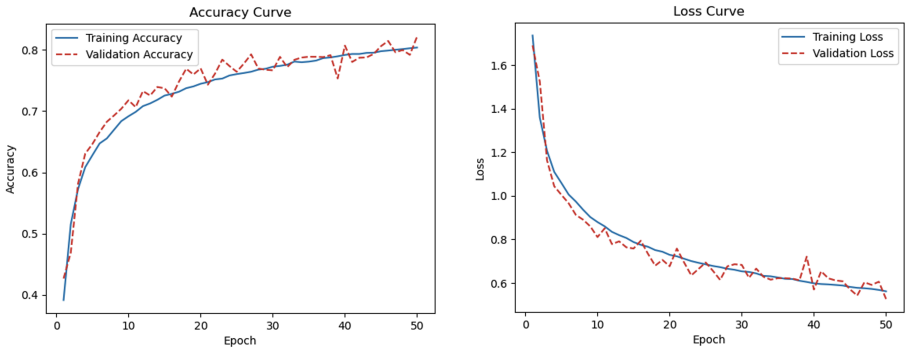

FIG. 3.— On the left, the Accuracy Curve displays Training and Validation Accuracy over 50 epochs, showing gradual improvement. On the right, the Loss Curve shows the Training and Validation Loss, demonstrating a sharp decline initially and minor fluctuations as epochs increase.

TABLE 1
Grid Search for Hyperparameters

Wd	Lr		
	0.1	**0.01**	**0.001**
0.1	19%	49%	77%
0.01	46%	72%	81%
0.001	62%	68%	69%

TABLE 2
Model Accuracy Comparison

Model	**Accuracy**
Logistic Regression	35%
KNN	30%
Random Forest	46%
CNN	81%

before reducing the feature dimensions.

We chose to use the Exponential Linear Unit (ELU) as the activation function, which, compared to the traditional ReLU, can reduce the problem of dying neurons and enhance the model's sensitivity to input variations. Additionally, each convolutional layer is followed by Batch Normalization, which not only speeds up the convergence of the model but also enhances the stability during training and helps to mitigate the issue of overfitting.

During the construction of the network, we conducted multiple experiments comparing the performance of kernel sizes 3x3, 5x5, and 7x7, and finally decided to use 3x3 kernel size because it showed the highest accuracy in our tests. In the model, as the depth of the network increases, we gradually increased the number of convolutional cores, which not only increased the complexity of the model but also enhances its feature detection capability.

Overall, the CNN model we designed is primarily composed of multiple convolutional blocks, each containing two consecutive 3x3 convolutional layers designed to deepen the model's learning of image features before reducing spatial dimensions. After each convolutional layer, we apply Batch Normalization to standardize the output, followed by an ELU activation function to introduce non-linearity. A 2x2 max pooling layer follows the second convolutional layer in each block to reduce computational burden and the risk of overfitting. We also have added a Dropout layer after each pooling operation, with dropout rates set to 0.2, 0.3, and 0.4 respectively, to gradually increase the regularization intensity as the network depth increases. After passing through three such convolution blocks, the network transforms the image data into a series of high-level feature representations, which are then mapped by a fully connected layer to the output of ten class predictions (see Figure 2).

Additionally, we identified the optimal combination of hyperparameters through grid search, setting the learning rate at 0.01 and the weight decay coefficient at 0.001. With these settings, the model achieved an accuracy of 81% on the CIFAR-10 test dataset, as shown in Table 1.

4. DISCUSSIONS

Our study compared the performance of four machine learning models—Logistic Regression, K-Nearest Neighbors, Random Forest, and Convolutional Neural Networks—in image classification using the CIFAR-10 dataset. Through analysis, it was found that the innovative CNN outperformed the other models, achieving an accuracy of 81% on the test dataset (see Table 2). Our findings underscore the importance of selecting appropriate models based on the complexity of the dataset and task at hand. While traditional models like Logistic Regression and KNN demonstrated limitations in handling the multi-class nature and high dimensionality of images, ensemble methods like Random Forest exhibited improved accuracy but still faced trade-offs in computational efficiency. Convolutional Neural Networks, with their ability to capture spatial relationships and learn hierarchical features, behaved as the most effective model for image classification tasks. Particularly for smaller input images like those in the CIFAR-10 dataset, inspired by the VGG structure, we departed from the traditional alternate layout of convolution and pooling layers. Instead, we adopted a novel approach of using consecutive convolutional layers followed by one pooling layer, which was then fine-tuned with batch normalization and dropout to control feature learning and combat overfitting. This adaptation made our model more effective on the CIFAR-10 dataset.

5. CONCLUSION AND FUTURE WORK

In conclusion, our study offers insights and recommendations for model selection in image recognition and highlights the importance of customizing network architectures to suit different datasets. In the future, we plan to explore the integration of more advanced regularization techniques and alternative architectures such as ResNets and DenseNets to further enhance model performance. We also aim to extend our research to include larger and more varied datasets to validate the robustness of our proposed modifications.