Comparison of Machine Learning Methods for Image Classification

Ziqi Cheng zcheng@unc.edu
Qicheng Geng qgeng@unc.edu
Tao Huang thuang@unc.edu
Weiye Zhang weiye.zhang@unc.edu

Abstract

Image classification is one of the most important and prevalent tasks in the world. It forms the basis of computer vision problems. Image classification applications are used in many areas, such as medical imaging, traffic control, robot vision, and much more. Machine learning techniques are often employed for image classification tasks. This paper aims to employ four different machine learning techniques on the same *Caltech 101* dataset and compare their performances.

1 Introduction

In the rapidly evolving landscape of machine learning, image classification remains a fundamental and challenging task with widespread applications. It forms the basis of computer vision problems, and it is used in areas such as medical imaging, traffic control, robot vision, and much more. Machine learning algorithms and techniques are often employed for this type of task, and there are various tools to choose from. This project aims to explore four specific algorithms, and qualitatively measure and compare their efficacy on the dataset *Caltech 101*, a widely recognized dataset for image classification.

The four types of models that we employed range from simple to complex. We start off with Logistic Regression and SVM, which are simple yet effective linear models. Following that, we explore Random Forest, an ensemble method that excels at capturing intricate patterns in data through the combination of decision trees. Finally, we venture into the field of deep learning with CNN, a neural network architecture tailored for image-related tasks. Throughout our study, we employed techniques such as hyperparameter tuning using a validation set and data normalization to achieve the best classification performance for each model.

2 Data and Preprocessing

2.1 Data

The dataset that is used in this paper is the *Caltech 101* dataset collected in September 2003 by Fei-Fei Li, Marco Andreetto, and Marc'Aurelio Ranzato. The dataset has 101 categories in total with about 40 to 800 images per category. We chose the 12 categories with the most amount of images for our purpose. The 12 categories are airplanes, motorbikes, faces, watch, leopards, bonsai, car side, ketch, chandelier, hawksbill, grand piano, and brain, in order of most amount of images to least from 800 to 98. In total there are 3241 RGB images of varies sizes and aspect ratios.

2.2 Preprocessing

Before doing any modeling, we have to preprocess our data. Because all images are of different sizes and aspect ratios, we resized all images to 128 by 128 pixels while preserving their aspect ratios by zero-padding. Thus each image now has shape 3 (channels) by 128 by 128 pixels. Then we split our dataset into training, validation, and testing set, with proportions 80%, 10% and 10% respectively, stratified with respect to the categories. By the end, we have 2592 training data, 324 validation data,

and 325 testing data. For the neural network model, we also normalized the image pixels values so that they have mean of 0 and standard deviation of value 1.

3 Models

3.1 Logistic Regression

Logistic Regression is a fundamental machine learning algorithm widely employed for binary classification tasks. It is particularly useful when the outcome of interest is binary or dichotomous, meaning it falls into one of two categories. In the context of logistic regression, the goal is to predict the probability of an event occurring, with outcomes typically represented as 0 or 1. So this may not be the best model when facing object recognition problems.

The model training start after loading the data and labels. Then the data are reshaped to 1 dimension to combine, which is flatting each image into a one-dimensional vector. Logistic Regression typically expects a flat feature vector as input. Then train a logistic regression model on the flattened image data, we use the model from sklearn. After training, evaluate the performance of the Logistic Regression model using appropriate metrics and print out the accuracy and the classification report.

The overall accuracy of the model on the test data is approximately 85.23%. The classification report shows that some classes have high precision but relatively lower recall, indicating that the model is selective in predicting these classes. While other classes may be challenging for the model to correctly classify.

Overall, Logistic Regression model achieved a reasonable accuracy on the test data, with strengths in some classes and potential challenges in others.

3.2 Support Vector Machine

We also adopted the Support Vector Machine (SVM) approach, known for its robustness and efficiency in handling high-dimensional data. Unlike deep learning methods, SVM does not inherently process the spatial structure of images. Instead, it focuses on finding a hyperplane that best segregates the different classes in a high-dimensional space. The strength of SVM lies in its use of kernel functions, which enable it to handle non-linearly separable data by transforming the original feature space into a higher dimension where a linear separation is possible.

We utilized a standard SVM with various kernel options, including linear, radial basis function (RBF), and polynomial kernels. Hyperparameter tuning was a critical component in adapting the model to our dataset's specific characteristics, where the regularization parameter C and the kernel coefficient γ played significant roles.

Employing a grid search approach, we meticulously tuned these hyperparameters. The heatmap in Figure 1 showcases the performance across a range of values for C and γ, with the color intensity indicating the accuracy of the model at each combination. The optimal model, as indicated by the brightest region on the heatmap, utilized an RBF kernel with C=10 and γ='scale'. This configuration achieved a commendable validation accuracy of approximately 89.74%. In the test phase, the fine-tuned SVM model demonstrated an overall accuracy of 87.69% on the test set.

Despite being computationally less demanding than deep learning alternatives, the SVM displayed significant effectiveness in the image classification task at hand. The results clearly illustrate the SVM's capability to manage the dataset's diversity and complexity, especially when empowered by appropriately chosen hyperparameters.

3.3 Random Forest

We also employed the Random Forest (RF) algorithm for the image classification tasks. RF approaches the problem of image classification by leveraging an ensemble of decision trees. This method focuses on making decisions based on subsets of features and aggregating these decisions to enhance overall accuracy and mitigate overfitting.

We utilized a standard RF algorithm with different configurations, particularly focusing on the number of trees and the depth of each tree. Hyperparameter tuning was an important aspect of this

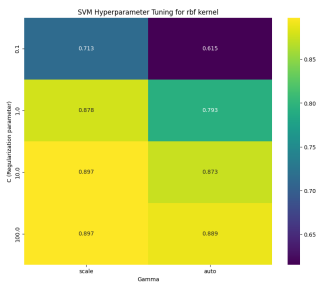

Figure 1: Heatmap of SVM hyperparameters and accuracy

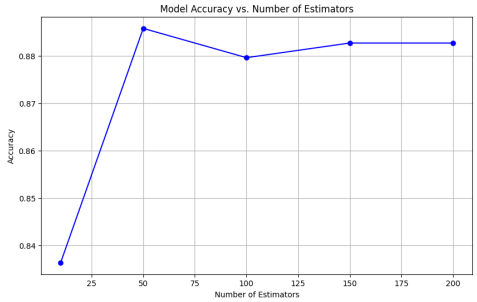

Figure 2: Random forest model accuracy vs. number of estimators

process, where parameters like the number of trees, maximum depth, minimum samples split, and minimum samples leaf were fine-tuned to adapt the model to the specific characteristics of our dataset. Using a grid search method, we successfully optimized hyperparameter. RF's best performance was achieved with n_estimators at 300, indicating its sensitivity to the number of trees in the forest. This configuration led to an validation accuracy of approximately 89.78%.

RFs are relatively efficient, especially given the lack of a need for input transformation. This efficiency is a significant consideration in scenarios where computational resources are limited. While RFs are powerful for general classification tasks, their lack of spatial data processing makes them less suited for complex image classification. However, their ease of implementation and interpretation can make them a good choice for simpler image classification tasks. In summary, Random Forests provide a robust and computationally efficient approach to image classification.

3.4 Convolutional Neural Network

We utilized deep learning, specifically convolutional neural networks (CNNs), for image classification as well. CNNs excel in preserving spatial structure and discerning spatial features. The convolutional layer, with multiple filters sliding over images, calculates dot products to identify matching features. The network learns filters to capture diverse core features of image classes. Following convolutional layers, fully connected layers process feature maps to generate class scores.

Our CNN architecture has the following structure: (1) several convolutional blocks that each consists of a convolutional layer, a ReLU activation function, and a 2 by 2 max pooling layer, (2) a dropout layer, and (3) several fully connected layers with ReLU. A 2 by 2 max pooling layer is used to downsample the images after each convolutional block to reduce the amount of information in an image while maintaining the essential features. A dropout layer is placed right before the fully connected layers to regularize the network and make the network more robust. The optimizer used is a stochastic gradient descent with a momentum of 0.9.

After grid-searching hyperparameters on the validation dataset, the best model found was the following shown in Figure 3: a convolutional block with four 5-by-5 kernels, a second convolutional block with eight 5-by-5 kernels, a dropout layer with drop probability 0.2, a fully connected hidden layer of dimension 64, and finally a fully connected layer that outputs the scores of 12 image classes. The best learning rate with 0.01 with an L2 regularizer of 0.01.

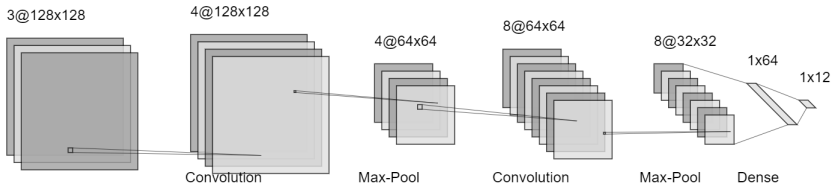

Figure 3: CNN architecture

Table 1: Model accuracies

Model	Accuracy
Logistic Regression	85.23%
SVM	87.69%
Random Forest	89.78%
CNN	90%

When training, we employed an early-stopping strategy which stops the training process when the validation loss fails to decrease in the past 10 epochs. The model was trained for 41 epochs with batch size 32 and resulted in a 92% accuracy on the validation set shown in Figure 4 and a 90% accuracy on the test set.

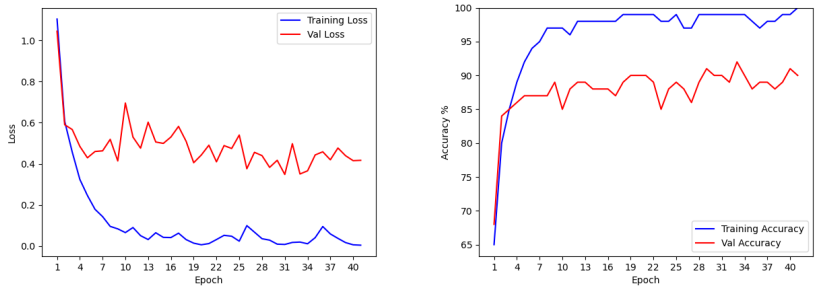

Figure 4: Left: training and validation loss curve. Right: training and validation accuracy curve

4 Discussions

The accuracies of each proposed model are shown in Table 1. We can see that every model performed well on out-of-sample images, which indicates that all of them are able to capture essential features of each image category. The relative ranking in the accuracies is as expected, with more complex models producing better results. However, it is surprising that a simple logistic regression was able to produce an accuracy of more than 85%. This shows that relatively simple models are capable of producing promising results on certain datasets and tasks. In fact, simple models are preferred in this case because they require less computing power to train and deploy for application purposes. The CNN model produced an accuracy of 90%, which we believe could be further improved with better and more extensive hyperparameter tuning. Further work can be done to improve the CNN model with better hyperparameter tuning techniques. Overall, all four proposed models achieved acceptable accuracy on the dataset, and the choice of which model should be preferred should be dependent on how much accuracy one can sacrifice for computer resources.

5 Conclusion

This paper shows the process of employing four different machine learning models for the image classification task on the *Caltech 101* dataset subset, and the comparison between their performances. We employed logistic regression, support vector machine, random forest, and convolutional neural network. Our conclusion is that in the context of this specific task, all four models are capable of achieving high accuracy, with more complex models performing better. However, more complex models require more computing resources, thus the compromise should be thought before making the decision of which is better. Further works can be done on the exploration of the performance of relatively simple models on tasks such as image classification.