Optimizing Convolutional Neural Networks for Object Classification on the COIL-100 Dataset

Machine Learning Project

Course:

Applied Machine Learning

Student:

Mohamed BOUIDAINE

Supervisors:

Mr. Mobyen Uddin Ahmed

Mr. Shaibal Barua

Contents

Abstract:

This report presents the training and evaluation of a Convolutional Neural Network (CNN) for object classification using the COIL-100 dataset, which contains 100 distinct object categories. The primary goal is to develop a model capable of accurately classifying images from this dataset, which includes images captured from multiple angles for each object. To optimize the performance of the CNN, hyperparameter tuning was conducted using grid search, focusing on critical parameters such as learning rate and batch size. The optimal values identified through this process were a learning rate of 0.00167 and a batch size of 32. The best-performing model achieved a high training accuracy of 97.18% and an impressive validation accuracy of 99.83%, indicating strong generalization capabilities. Further evaluation on an unseen test set demonstrated the model's effectiveness in real-world scenarios, with high precision (0.9970), recall (0.9965), and F1-score (0.9965), suggesting minimal false positives and false negatives. These results confirm the model's potential for accurate object classification. This report outlines the methodology used to train and evaluate the model, provides detailed results, and discusses the model's overall performance and possible improvements.

1. Introduction:

Object recognition is a core problem in the field of computer vision, aiming to automatically detect and classify objects within digital images. This problem has practical applications across numerous industries, including autonomous vehicles, security, and retail, where accurately identifying objects in images or video feeds is crucial. The COIL-100 dataset, a widely used benchmark in object classification, consists of 100 different object categories, with 72 images per category captured under varying conditions. These conditions include changes in orientation, lighting, and background, which add complexity to the classification task.

Convolutional Neural Networks (CNNs) have become the go-to deep learning architecture for such tasks due to their remarkable ability to automatically learn hierarchical features from raw image data. CNNs are particularly effective because they are able to capture spatial relationships within images, making them highly suitable for tasks that require pattern recognition, such as object classification. In this study, we aim to optimize the performance of a CNN model on the COIL-100 dataset by fine-tuning key hyperparameters, such as learning rate and batch size, which play a significant role in the model's convergence and generalization ability.

To achieve this, we employ grid search, a technique that systematically tests a range of hyperparameter values to identify the combination that maximizes the model's accuracy. We focus on finding the optimal values that allow the model to achieve high performance not only on the training data but also on unseen test data, ensuring that the model can generalize effectively. This report outlines the methodology used to train and fine-tune the CNN model, presents the results of the experiments, and provides insights into the model's performance, highlighting its ability to generalize to previously unseen images.

2. Background and related work:

COIL-100 dataset:

The COIL-100 dataset, short for Columbia Object Image Library, is a widely utilized dataset in the field of computer vision and machine learning, particularly for object recognition tasks. It was introduced by Nene, Nayar, and Murase in 1996 at Columbia University. The dataset contains 100 different objects, each captured from 72 different angles, resulting in 7,200 images. These objects vary in shape, size, and color, providing a rich set of images for researchers to develop and test machine learning models for object classification and recognition.

The COIL-100 dataset was created to address the growing need for benchmark datasets in object recognition research. During the 1990s, the field of computer vision was advancing rapidly, yet there was a scarcity of standardized datasets to evaluate and compare the performance of different algorithms. The COIL-100 dataset filled this gap by providing a comprehensive set of images with consistent lighting and background conditions, allowing for controlled experiments and reproducible results [1].

Each object in the COIL-100 dataset is placed on a turntable, which rotates to capture images at 5-degree intervals, producing a total of 72 images per object. All images are 128x128 pixels, which is relatively small by today's standards but was sufficient at the time of its release. This resolution provides a balance between computational efficiency and sufficient detail for effective object recognition. The objects included range from household items like mugs and bottles to complex shapes like toys and sculptures, providing a diverse set of classes for classification tasks.

Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) have become the cornerstone of modern computer vision tasks, including object recognition, due to their ability to automatically and adaptively learn spatial hierarchies of features from input images. Inspired by the visual cortex of animals, CNNs are designed to process data with a grid-like topology, such as images [2]

A CNN typically consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers apply filters to the input to create feature maps that capture various aspects of the image, such as edges, textures, and more complex structures. Pooling layers reduce the spatial dimensions of the feature maps, which helps in reducing computational load and controlling overfitting. Finally, fully connected layers interpret the high-level features and output the classification result.

Related Work

Since its inception, the COIL-100 dataset has been extensively used in various research studies, particularly in developing and testing algorithms for object recognition, classification, and pose estimation. Early studies focused on traditional machine learning techniques such as k-nearest neighbors (k-NN), support vector machines (SVM), and decision trees. These methods relied heavily on handcrafted features like edges, corners, and textures to differentiate between objects [3].

With the advent of deep learning, particularly Convolutional Neural Networks (CNNs), the COIL-100 dataset found renewed relevance. CNNs, which can automatically learn hierarchical features from raw images, significantly outperformed traditional methods in object recognition tasks. Researchers used the COIL-100 dataset to benchmark CNN architectures, test different training techniques, and explore transfer learning capabilities.

One notable study by Krizhevsky, Sutskever, and Hinton in 2012 [4], which introduced the AlexNet architecture, demonstrated the power of deep learning in image classification. Although AlexNet was primarily trained on larger datasets like ImageNet, the principles and architecture were applied to smaller datasets like COIL-100, yielding impressive results. This highlighted the importance of deep learning in advancing object recognition tasks across different datasets.

Moreover, the COIL-100 dataset has also been used in the development of novel approaches such as unsupervised learning and domain adaptation. Researchers leveraged the dataset's rich variability to test models' ability to generalize across different viewing angles and object orientations, a critical factor in real-world applications of object recognition systems. For example, this paper “Unsupervised domain adaptation by domain invariant projection.” [5] discusses unsupervised domain adaptation and highlights the importance of datasets like COIL-100 for testing models' generalization across varying conditions.,

3. Approach and Implementation

Problem formulation:

The problem to be solved is developing a robust image classification model that can accurately identify and categorize objects from the COIL-100 dataset. The challenge lies in ensuring high performance despite the dataset's limited size and variations in object appearances.

Data Pre-processing:

The COIL-100 dataset is employed to train a CNN model for image classification. Given that the COIL-100 dataset is relatively clean from noise, there is no requirement for applying noise-reduction filtering algorithms.

Labels:

The dataset does not include a CSV file for label descriptions; instead, labels are derived from the filenames. Each filename starts with the category name, ranging from 1 to 100 (e.g., Obj1__0, Obj1__1, etc.). Labels are extracted by taking the initial segment of the filename before the double underscores. Additionally, it is noteworthy that the dataset is balanced, with each class containing an equal number of samples (72 images per class).

Normalizing data:

To ensure the CNN model receives appropriate input, the images are normalized by dividing the pixel values by 255, scaling them to the range [0, 1].

Data-splitting:

Given the relatively small size of the dataset, a significant portion is allocated for training. The dataset is split with 80% for training and 20% for testing. Further, the training set is divided into training and validation sets in a 90/10 ratio. This division facilitates proper hyperparameter tuning using the validation set and effective evaluation of the model on unseen test data.

Model and hyperparameter selection:

A Convolutional Neural Network (CNN) is an ideal choice for addressing image classification problems due to its strong feature learning capabilities, particularly for inputs with spatial structure. In summary, state-of-the-art research consistently demonstrates that CNNs deliver excellent performance in image classification tasks.

CNN Network:

Designing a CNN model primarily depends on the task and the dataset. Since our dataset is relatively small and the input size is also modest (128x128), a typical shallow CNN (not very deep) should be sufficient to achieve good performance in solving the problem.

- CNN Layers:
    - Input layer: Accepts a three-dimensional image with 3 channels (RGB) (128, 128, 32).
    - First convolutional layer: Applies 32 filters, each with a kernel size of 3x3. It uses the ReLU activation function, and the output shape is (126, 126, 32) after applying the convolution operation (input size 128x128 reduced by 2 on both dimensions due to kernel size)
    - First max-pooling layer: Performs 2x2 max-pooling, which reduces the spatial dimensions by a factor of 2. The output shape after this layer is (63, 63, 32), reducing the width and height by half.
    - Second convolutional layer: Applies 64 filters, each with a kernel size of 3x3, and uses ReLU as the activation function. The output shape is (61, 61, 64) after applying the convolution operation to the feature maps from the previous layer.
    - Second max-pooling layer: Performs 2x2 max-pooling, reducing the spatial dimensions by a factor of 2 again. The output shape after this layer is (30, 30, 64).
    - Third convolutional layer: Applies 128 filters, each with a kernel size of 3x3, and uses ReLU as the activation function. The output shape is (28, 28, 128) after applying the convolution operation.
    - Third max-pooling layer: Performs 2x2 max-pooling, which again reduces the spatial dimensions by half. The output shape after this layer is (14, 14, 128).
    - Flatten layer: Flattens the 3D feature maps (14x14x128) into a 1D vector of size 25088 (14 * 14 * 128), which is required before passing the data into fully connected layers.
    - First dense layer: A fully connected layer with 128 neurons, using ReLU as the activation function. The output shape is (128,) as it consists of 128 neurons in a 1D vector.
    - Dropout layer: A dropout rate of 0.5 means that 50% of the neurons in the previous layer will be randomly dropped during training to prevent overfitting.
    - Output dense layer: A fully connected layer with 100 neurons (one for each class in the COIL-100 dataset), using softmax as the activation function. The output shape is (100,), where each element represents the probability of our object classes.

Tuning hyperparameters:

To optimize the hyperparameters, we will use the Grid Search algorithm to tune the key hyperparameters. These are the parameters that have the greatest impact on the performance of the CNN model. Due to limited computing resources, we will focus on two critical hyperparameters: learning rate and batch size. If the performance with the tuned parameters meets expectations, further adjustments to other parameters will not be necessary. While other hyperparameters, such as filter sizes, layer density, and number of epochs, can also influence performance, we will limit the search space by setting these to typical values suited for our use case (small dataset). The values selected for the grid search are commonly used in CNN models.

Values chosen for grid search:

Learning rates: [0.00167, 0.0001, 0.01]

NB: for the 0.00167, before doing grid search, I tried using Hyperband tuning [6] but that took a long time, but the value 0.00167 for learning rate gave a good performance.

Batch sizes: [8, 16, 32]

4. Evaluation:

After performing the grid search, the best model was the one constructed with the following hyperparameters:

- Learning rate: 0.00167.
- Batch size: 32.

For both accuracy and loss, Figure 1 shows a quick convergence in the first epochs. Then it converges slowly until reaching approximately 15 epochs.

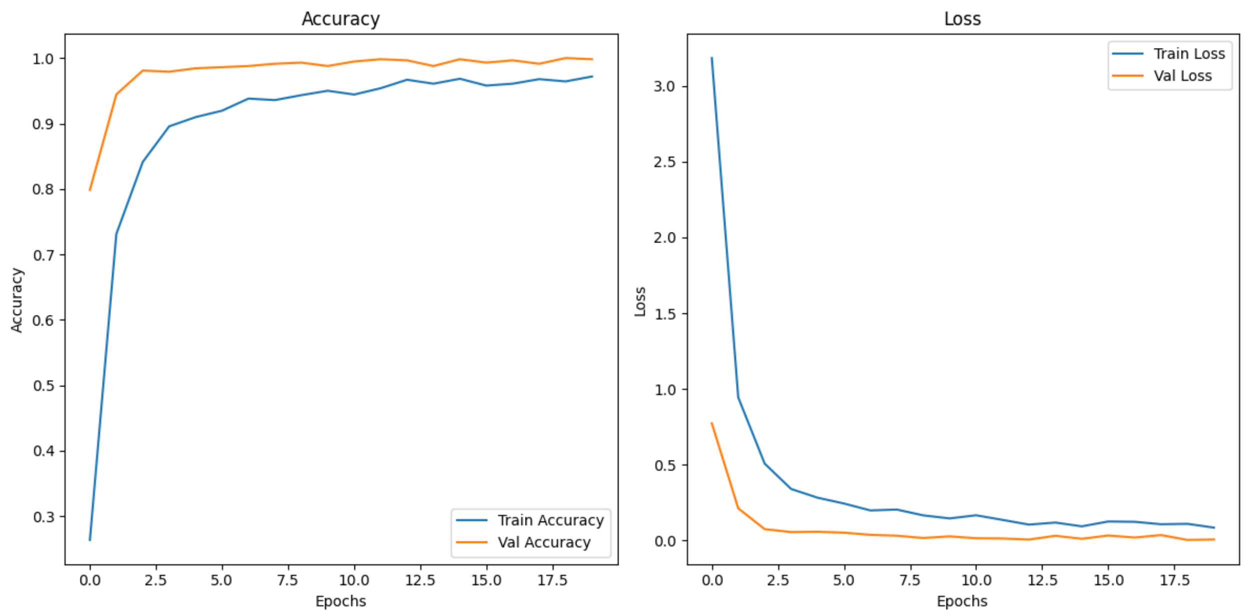

Figure 1. Training/ Validation Accuracy/Loss curves

Best model performance after 20 epochs:

- Train accuracy: 0.9718364477157593
- Val accuracy: 0.9982638955116272
- Train loss: 0.08466453105211258
- Val loss: 0.006558573339134455

Tested on unseen test data (20%), the model is performing well

- Precision: 0.9970, indicating a very few false positives.
- Recall: 0.9965, indicating a very few false negatives.
- F1 Score: 0.9965, that reflects the overall performance. (classes are balanced, so a good f1-score was expected).

5. Discussion:

The results of the CNN model on the COIL-100 dataset reveal a well-performing architecture. The grid search for hyperparameter tuning identified the optimal values of a learning rate of 0.00167 and a batch size of 32, which led to rapid convergence in the early epochs and slower refinement in subsequent epochs. The model achieved an impressive training accuracy of 97.18% and a validation accuracy of 99.83%, indicating a robust generalization capability. The low training loss of 0.084 and the even lower validation loss of 0.006 further confirm that the model is able to learn effectively from the data without overfitting.

When evaluated on the unseen test set, the model demonstrated a precision o(0.9970), recall (0.9965), and F1-score (0.9965). These metrics indicate that the model correctly classified the majority of the test data with very few false positives and false negatives. The balanced nature of the COIL-100 dataset likely contributed to the high F1-score, as the model was able to learn discriminative features across the 100 classes. This result highlights the potential of CNNs for multi-class image classification tasks. Although the architecture's performance was strong, the training process could be further optimized through techniques such as data augmentation or exploring alternative architectures. Nevertheless, the model demonstrated strong generalization capabilities and performed well even on previously unseen images, showcasing its potential for real-world applications in object recognition.

6. Conclusion:

The CNN model trained on the COIL-100 dataset demonstrated excellent classification performance, achieving high accuracy and low loss values. The grid search for hyperparameter tuning successfully identified optimal values for learning rate and batch size, resulting in a robust model. The high precision and recall values on the test set indicate that the model is effective in minimizing classification errors. Future work could explore further fine-tuning of the architecture or the inclusion of data augmentation techniques to improve performance further.

References:

[1] Nene, S.A., Nayar, S.K. and Murase, H., 1996. Columbia object image library (coil-20).

[2] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), pp.2278-2324.

[3] Hart, P.E., Stork, D.G. and Duda, R.O., 2000. *Pattern classification*. Hoboken: Wiley.

[4] Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25.

[5] Baktashmotlagh, M., Harandi, M.T., Lovell, B.C. and Salzmann, M., 2013. Unsupervised domain adaptation by domain invariant projection. In Proceedings of the IEEE international conference on computer vision (pp. 769-776).

[6] Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A. and Talwalkar, A., 2018. Hyperband: A novel bandit-based approach to hyperparameter optimization. Journal of Machine Learning Research, 18(185), pp.1-52.