Université Internationale de Rabat
THE INNOVATIVE UNIVERSITY

HomeWork

Introduction to Computer Vision

Realised by:

Hajar Lekrafi
115303

Université Internationale de Rabat - 2023 / 2024

Contents

1 Introduction to Computer Vision

Computer vision is an interdisciplinary field that combines elements from computer science, artificial intelligence, and image processing to enable machines to interpret and understand the visual world. This field has a wide range of applications, from autonomous vehicles and medical image analysis to facial recognition and augmented reality. In this project, we focus on image classification, a core task in computer vision, where the goal is to assign a label to an input image from a predefined set of categories. Specifically, we use the CIFAR-10 dataset to classify images into two categories: objects that can fly and objects that cannot. Convolutional Neural Networks (CNNs) are employed due to their powerful feature extraction capabilities and their success in handling image data.

2 Goal of the Project

The primary goal of this project is to develop an accurate and efficient binary classification model capable of distinguishing between images of objects that can fly (such as airplanes and birds) and those that cannot (such as cars, cats, and other objects). Achieving this involves several specific objectives:

- **Understanding the Dataset:** Familiarize ourselves with the CIFAR-10 dataset, which consists of 60,000 32x32 color images across 10 different classes, and prepare it for binary classification.
- **Designing a CNN Architecture:** Create a Convolutional Neural Network that can effectively learn from the training data and generalize well to new, unseen images.
- **Training the Model:** Optimize the CNN by selecting appropriate parameters, such as learning rate and batch size, and by training it over multiple epochs.
- **Evaluating Performance:** Use a variety of performance metrics, including accuracy, precision, recall, and the confusion matrix, to assess the model's effectiveness.
- **Improving the Model:** Explore potential improvements, such as deeper network architectures, data augmentation, and hyperparameter tuning, to enhance the model's performance.

3 Importing Libraries

To build, train, and evaluate our CNN model, we utilize several key libraries:

- **TensorFlow and Keras:** TensorFlow is a robust open-source platform for machine learning developed by Google. Keras, which operates as an API within TensorFlow, provides a high-level interface for building and training deep learning models. These libraries simplify the development of neural networks and include a wide range of pre-built components, such as layers, loss functions, and optimizers.
- **NumPy:** NumPy is fundamental for scientific computing in Python. It supports large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. This is essential for handling the large datasets and performing the numerical operations required in machine learning.
- **Matplotlib:** This library is used for data visualization in Python. It allows us to create a wide variety of plots and graphs, which are crucial for visualizing the training process, evaluating model performance, and presenting results in an understandable manner.
- **Scikit-learn:** Scikit-learn provides simple and efficient tools for data mining and data analysis. It includes modules for model evaluation, such as precision and recall metrics, and tools for generating confusion matrices, which help in assessing the performance of our model.

4 Loading CIFAR-10 Dataset

The CIFAR-10 dataset is a well-established benchmark in the field of machine learning and computer vision. Created by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, it consists of 60,000 32x32 color images in 10 distinct classes, with 6,000 images per class. The dataset is divided into 50,000 training images and 10,000 testing images. Each image is labeled with one of the ten classes, including airplanes, cars, birds, cats, and more. This dataset is sufficiently challenging to test various image classification algorithms but small enough to allow for quick experimentation and training.

5 Data Preprocessing

Data preprocessing is a crucial step to ensure that the input data is in the right format for the neural network. In this project, we transform the CIFAR-10 dataset for binary classification:

- **Binary Classification Labels:** We redefine the labels into two categories: 'can fly' (including airplanes and birds) and 'cannot fly' (including the other eight classes). This involves mapping the original class labels to our binary classification labels.

- **Normalization:** Image data needs to be normalized to improve the training process. Normalization scales the pixel values from the original range (0-255) to a range of 0-1, which helps in faster convergence during training and ensures that the model learns efficiently. Normalized data also help in preventing numerical instabilities and improving the performance of the gradient descent algorithm.

6 Defining the CNN Architecture

The architecture of our Convolutional Neural Network (CNN) is designed to efficiently process and classify images:

- **Convolutional Layers:** These layers are the core building blocks of a CNN. They apply a set of filters to the input image, creating feature maps that highlight various aspects of the images, such as edges, textures, and shapes.

- **Pooling Layers:** MaxPooling layers reduce the spatial dimensions of the feature maps, which helps in reducing the computational load and the number of parameters, preventing overfitting.

- **Fully Connected Layers:** After the convolutional and pooling layers, the output is flattened and passed through fully connected (dense) layers. These layers perform the final classification based on the extracted features.

- **Activation Functions:** The ReLU (Rectified Linear Unit) activation function is used in the convolutional layers to introduce non-linearity, which helps the network learn more complex patterns. The sigmoid activation function is used in the output layer to generate a probability score for the binary classification.

7 Compiling the Model

Compiling the model involves setting up the configuration needed for training. This includes selecting the optimizer, loss function, and evaluation metrics:

- **Optimizer:** The Adam optimizer is chosen for its efficiency and adaptability. It combines the advantages of two other extensions of stochastic gradient descent, specifically AdaGrad and RMSProp, to improve both the speed and performance of training.
- **Loss Function:** Binary cross-entropy is used as the loss function. It is particularly suitable for binary classification tasks as it measures the performance of the model by comparing the predicted probabilities with the true binary labels.
- **Metrics:** Accuracy is used as the primary metric to monitor during training and validation. It provides a straightforward measure of the proportion of correct predictions made by the model.

8 Training the Model

Training the model involves feeding the training data into the network and allowing it to learn the optimal weights through the backpropagation algorithm:

- **Epochs:** An epoch refers to one complete pass through the entire training dataset. The number of epochs determines how many times the learning algorithm will work through the dataset.
- **Validation:** During training, the model's performance is validated using a separate validation set (the test images). This helps in monitoring the model's ability to generalize to new data and in detecting overfitting.

9 Model Evaluation

After training, we evaluate the model on the test dataset to assess its performance. The evaluation process involves calculating the loss and accuracy on the test data. These metrics give us an indication of how well the model is likely to perform on unseen data. A high accuracy and low loss indicate that the model has learned to generalize well from the training data.

10 Predicting and Generating Metrics

To gain a deeper understanding of the model's performance, additional metrics such as precision and recall are calculated:

- **Precision:** Precision measures the accuracy of the positive predictions. It is the ratio of true positive predictions to the total predicted positives, indicating how many of the predicted flying objects are actually correct.
- **Recall:** Recall measures the model's ability to identify all relevant instances. It is the ratio of true positive predictions to the actual positives, indicating how many flying objects were correctly identified by the model.

11 Confusion Matrix

The confusion matrix is a powerful tool for visualizing the performance of a classification model:

- **True Positives (TP):** The number of instances correctly predicted as positive (flying objects).
- **True Negatives (TN):** The number of instances correctly predicted as negative (non-flying objects).
- **False Positives (FP):** The number of instances incorrectly predicted as positive.
- **False Negatives (FN):** The number of instances incorrectly predicted as negative.
  Analyzing the confusion matrix helps in understanding the types of errors made by the model and identifying areas for improvement.

12 Conclusion

This project demonstrated the application of Convolutional Neural Networks for binary image classification. The model achieved high accuracy and showed strong performance in distinguishing between flying and non-flying objects. Key findings include the effectiveness of data preprocessing steps, such as normalization and binary labeling, and the suitability of the CNN architecture for this task. Future work could explore deeper network architectures, data augmentation techniques, and hyperparameter tuning to further enhance the model's performance. Additionally, experimenting with other datasets and classification tasks could provide further insights into the versatility and robustness of CNNs in computer vision applications.