Custom ResNet Architecture for CIFAR-10 and CIFAR-100

Marin Marius

November 24, 2023

1 Introduction

This document examines the custom ResNet (Residual Network) implementation tailored for image classification on the CIFAR-10 and CIFAR-100 datasets. We focus on a custom ResNet34 architecture, specifically detailing its structure, forward function mechanics, and the gradient flow during backpropagation.

2 Model Architecture

The ResNet model leverages convolutional layers for feature extraction, batch normalization for input stabilization, ReLU activation for non-linearity, and shortcut connections for mitigating vanishing gradients. The architecture includes an initial convolutional layer, followed by sequential applications of the 'BasicBlock' class, and concludes with a global average pooling and a fully connected layer for classification.

2.1 Layers of the Model

The architecture's layers serve the following functions:

- Convolutional Layers: Extract salient features from input images.
- Batch Normalization: Normalizes and stabilizes the learning process.
- ReLU Activation: Introduces non-linearity to the model.
- Shortcut Connections: Facilitate the training of deeper networks by providing alternate paths for gradient flow.
- Fully Connected Layer: Outputs the classification results based on extracted features.

3 BasicBlock Structure

The ‘BasicBlock‘ class represents the fundamental building block of the custom ResNet model. It comprises two convolutional layers with batch normalization, followed by ReLU activations, and includes a shortcut connection that enables gradients to bypass these layers, preventing gradient dilution during backpropagation.

4 Custom ResNet Architecture

Our custom ‘ResNet‘ class constructs a network with several ‘BasicBlock‘ instances. The network begins with an initial convolutional layer, continues through the ‘BasicBlock‘ layers, and concludes with a pooling layer that feeds into a fully connected layer for classification. Our CIFAR-10 model employs a specific sequence of blocks, which is detailed further in the document.

5 Forward Function

The ‘forward‘ method in our ‘ResNet‘ class defines the journey of an input tensor through the network, detailing how it traverses the initial convolution, the ‘BasicBlock‘ instances, and the output layers, culminating in a class prediction.

6 Gradient Flow in Backpropagation

During backpropagation, the shortcut connections in the ‘BasicBlock‘ ensure a robust flow of gradients, allowing for the effective training of deep architectures. These connections are essential in preserving gradient magnitudes through the network’s depth.

7 Contribution of Layers to the Result

Each layer type in our ResNet model plays a critical role in the CIFAR-10 classification task:

- Convolutional layers extract relevant features from raw images.
- Batch normalization contributes to more stable and faster convergence.
- ReLU activations allow the model to capture complex patterns.
- Shortcut connections are pivotal for training deep models effectively.

Empirical observations and training data suggest that these layers significantly contribute to the model’s high classification accuracy.

8 Transfer Learning Implementation

For transfer learning, we utilize the weights from the pretrained ‘ResNet34‘ model available in ‘torchvision.models‘. These weights are transferred to our custom ‘ResNet34‘ architecture, which follows the same layer structure. We ensure that the final fully connected layer is adjusted to the number of target classes (10 for CIFAR-10 and 100 for CIFAR-100), and during fine-tuning, the entire network's weights are updated to adapt to the new dataset.

9 Conclusion

Our custom ResNet architecture, with its unique ‘BasicBlock‘ components, demonstrates high efficacy for image classification on CIFAR datasets. The architecture's ability to retain gradient flow through shortcut connections enables the successful training of deep models, and the transfer learning approach allows for leveraging pretrained weights effectively, resulting in significant performance gains.