VGGNet REVIEW

SongSeungWu

1 What is VGGNet?

VGGNet is a deep convolutional neural network (CNN) that achieved high performance in the 2014 ImageNet Challenge. Developed by the Visual Geometry Group (VGG) at the University of Oxford, this network is characterized by a simple yet very deep architecture, consisting of stacked convolutional layers. The key feature of VGGNet is the use of small 3x3 convolutional filters applied repeatedly, which is optimized for extracting complex features. The network periodically applies Max Pooling to reduce the spatial dimensions, and the final classification is done through fully connected layers.

VGGNet is deep (typically 16 to 19 layers) and uses fixed-size convolutional kernels (3x3) to capture fine-grained features from images. However, its depth and small kernel size lead to high computational cost and memory usage.

2 Problems with VGGNet

- Computational cost: VGGNet requires a significant amount of computation due to its depth and the repeated use of small convolutional kernels. This can slow down training and make it challenging to handle large datasets.
- Model size: VGGNet has a large number of parameters, which leads to high memory consumption. The model requires substantial hardware resources.
- Training time: Due to its deep architecture, VGGNet may take a long time to train, and convergence can be slow.

3 Problem-Solving Approach

While VGGNet's deep and complex structure leads to high computational costs, this issue can be addressed by incorporating Batch Normalization. In the code, BatchNorm2d layers were added after each convolutional layer in VGGNet, which helps mitigate the vanishing gradient problem and accelerates convergence.

Batch Normalization normalizes the input distribution at each layer during training, increasing the training speed and stability of the model. This code follows the VGG-16 architecture and uses MaxPooling to progressively reduce the spatial dimensions, followed by a fully connected layer that classifies the inputs into 3 categories.

4 Results Analysis

The experimental results show that the initial accuracy started at 65.17% and eventually improved to 92.3%. Below is a summary of the key results.

Accuracy and Loss per Epoch:

- Epoch 0: Loss = 0.7897, Accuracy = 65.17%
- Epoch 5: Loss = 0.2579, Accuracy = 91.73%
- Epoch 10: Loss = 0.0201, Accuracy = 91.27%
- Epoch 15: Loss = 0.2811, Accuracy = 92.97%
- Epoch 19: Loss = 0.1180, Accuracy = 92.30%

Elapsed Time:

- The time per epoch was approximately 1.94 to 1.96 seconds.

Key Findings:

- Training Speed: The short time per epoch suggests that the model was trained efficiently on a dataset of appropriate size.
- Loss Reduction: The loss started at 0.7897 in Epoch 0 and decreased significantly to 0.1180 by Epoch 19. This indicates that the model learned effectively and reached a low final loss value.
- Accuracy Improvement: The model achieved an accuracy of 92.3% by Epoch 19, demonstrating that the network converged well and achieved high performance.

5 Summary of Results

The experiments using VGGNet showed highly promising outcomes. Initially, the accuracy was low, but as training progressed, the loss decreased significantly, and the accuracy rapidly improved. By the end, the model achieved an accuracy of 92.3% with a loss of 0.118.

Moreover, the addition of Batch Normalization appears to have resolved the vanishing gradient problem and improved the stability and speed of training. The training time per epoch was under 2 seconds, indicating that the model was trained quickly with sufficient computational resources.

ResNet REVIEW

SongSeungWu

1 What is ResNet?

ResNet (Residual Network) is a deep convolutional neural network (CNN) developed by Microsoft Research in 2015. The key innovation of ResNet is the introduction of residual connections. These residual connections effectively solve the vanishing gradient problem, which often occurs in very deep networks. By using these connections, ResNet allows much deeper networks to be trained successfully.

Key Features of ResNet:

- Residual Connections: ResNet incorporates skip connections, which pass the input directly to the next block of the network. This helps the network to learn even when it becomes very deep.
- Block Structure: ResNet is built with blocks that consist of 1x1 and 3x3 convolutional layers. These blocks can be stacked very deeply (e.g., ResNet-50, ResNet-101).
- Performance Improvement: The residual connections enable the network to grow deeper without suffering from degradation in performance, and they stabilize the learning process.

2 Problems with ResNet

- Complex Model Structure: As ResNet gets deeper, the number of parameters increases, leading to high computational costs.
- Time Consumption: Deeper networks take longer to train, and this is particularly noticeable when working with large images or complex datasets.
- Memory Usage: ResNet's deep architecture consumes a large amount of memory, especially for models with 50 or more layers, which can put a significant load on GPU memory.

3 Problem-Solving Approach

The issues caused by ResNet's complexity and depth are mitigated through Batch Normalization and Residual Connections.

- Batch Normalization: BatchNorm layers were added after each convolutional layer to improve the training speed and to help solve the vanishing gradient problem. This led to faster convergence and more stable training.
- Residual Connections: The residual connections, a key feature of ResNet, allowed for deeper networks to be trained by preventing the network from "forgetting" or "getting stuck" during learning. This greatly alleviated issues related to vanishing gradients.

4 Results Analysis

The experimental results show that the initial accuracy started at 74.67% and eventually improved to 82.1%. Below is a summary of the key results:

Accuracy and Loss per Epoch:

- Epoch 0: Loss = 0.8624, Accuracy = 74.67%
- Epoch 5: Loss = 0.1959, Accuracy = 81.57%
- Epoch 10: Loss = 0.4399, Accuracy = 79.97%
- Epoch 11: Loss = 0.0602, Accuracy = 82.10%

Elapsed Time:

- Each epoch took around 11.60 seconds.

Key Insights:

- Training Speed: Each epoch took approximately 11.60 seconds, which remained consistent throughout, indicating relatively fast training despite the deep architecture.
- Loss Reduction: The initial loss was 0.8624 and it dropped significantly by Epoch 5 to 0.1959, eventually reaching 0.0602. This demonstrates that the model was learning effectively and achieving a low final loss.
- Accuracy Improvement: By the final epoch, the model achieved an accuracy of 82.1%, showing that the deep ResNet model successfully converged and performed well on the dataset.

5 Summary of Results

The experiments using ResNet-50 showed that the model benefited greatly from residual connections and batch normalization, enabling stable training even in deep networks. Starting with an initial accuracy of 74.67%, the model improved to 82.1%. Moreover, the loss decreased from 0.8624 to 0.0602.

The residual connections prevented vanishing gradient issues, while batch normalization enhanced training speed and stability. Each epoch took approximately 11.6 seconds, showing that the model trained efficiently with consistent resource usage throughout the process.