DL/DLOps (2023)
Lab Assignment 6

- Pranay (B20AI030)

Model

1. ResNet-34

Inferencing Technique used	Model Size (MB)	Average Execution Time
Before Inferencing Technique	81.40	0.000843 seconds
ONNX	81.38	91.29 ms
ONNX Quantized	20.44	92.81 ms
Torchscript (CPU)	81.57	4.81 ms
Torchscript (GPU)	81.57	0.40 ms

Analysis

ResNet34 is a popular convolutional neural network architecture that was introduced in 2015. It consists of 34 layers, including a series of convolutional layers, batch normalization layers, and ReLU activation functions. ResNet34 utilizes residual connections, which allow for the training of much deeper networks without the problem of vanishing gradients.

ONNX conversion:

When ResNet34 is converted to ONNX format, the resulting model can be run on any platform that supports ONNX. This allows for easy portability and deployment of the model.

ONNX quantization:

Quantization is a process that reduces the precision of the weights and activations in the model, which can significantly reduce the model's memory footprint and increase inference speed. When ResNet34 is quantized using ONNX, the resulting model is smaller and faster but has a slightly lower accuracy than the original model.

Torchscript conversion:

When ResNet34 is converted to Torchscript, the resulting model can be executed on any platform that supports PyTorch. This format provides excellent performance and flexibility but may require additional code modifications to work efficiently on certain platforms.

2. DenseNet-121

Inferencing Technique used	Model Size (MB)	Average Execution Time
Before Inferencing Technique	30.63	0.000843 seconds
ONNX	30.11	72.00 ms
ONNX Quantized	8.44	62.52 ms
Torchscript (CPU)	31.39	5.02 ms
Torchscript (GPU)	31.39	1.11 ms

Analysis

DenseNet121 is another popular convolutional neural network architecture that was introduced in 2017. It consists of 121 layers and has a dense block structure, where each layer is connected to every other layer in the block. This structure allows DenseNet121 to achieve high accuracy with fewer parameters than other networks.

ONNX conversion:

When DenseNet121 is converted to ONNX format, the resulting model can be run on any platform that supports ONNX. This allows for easy portability and deployment of the model.

ONNX quantization:

When DenseNet121 is quantized using ONNX, the resulting model is smaller and faster but has a slightly lower accuracy than the original model.

Torchscript conversion:

When DenseNet121 is converted to Torchscript, the resulting model can be executed on any platform that supports PyTorch. This format provides excellent performance and flexibility but may require additional code modifications to work efficiently on certain platforms.

3. EfficientNet-B0

Inferencing Technique used	Model Size (MB)	Average Execution Time
Before Inferencing Technique	20.37	0.001096 seconds
ONNX	20.15	31.40 ms
ONNX Quantized	5.33	32.80 ms
Torchscript (CPU)	20.80	4.68 ms
Torchscript (GPU)	20.80	1.22 ms

Analysis

EfficientNet-B0 is a state-of-the-art convolutional neural network architecture that was introduced in 2019. It was designed to achieve high accuracy with fewer parameters and less computational cost than other networks. EfficientNet-B0 uses a combination of depth-wise convolutions, squeeze-and-excitation blocks, and compound scaling to achieve high accuracy.

ONNX conversion:

When EfficientNet-B0 is converted to ONNX format, the resulting model can be run on any platform that supports ONNX. This allows for easy portability and deployment of the model.

ONNX quantization:

When EfficientNet-B0 is quantized using ONNX, the resulting model is smaller and faster but has a slightly lower accuracy than the original model.

Torchscript conversion:

When EfficientNet-B0 is converted to Torchscript, the resulting model can be executed on any platform that supports PyTorch. This format provides excellent performance and flexibility but may require additional code modifications to work efficiently on certain platforms.

4. ConvNeXt-T

Inferencing Technique used	Model Size (MB)	Average Execution Time
Before Inferencing Technique	338.14	0.001221 seconds
ONNX	338.16	303.18 ms
ONNX Quantized	85.42	406.41 ms
Torchscript (CPU)	338.31	10.94 ms
Torchscript (GPU)	338.31	0.68 ms

Analysis

ConvNeXt-T is a recent convolutional neural network architecture that was introduced in 2020. It uses a combination of depth-wise and group convolutions to achieve high accuracy with fewer parameters than other networks. ConvNeXt-T also utilizes tensor decomposition techniques to reduce the model's computational complexity.

ONNX conversion:

When ConvNeXt-T is converted to ONNX format, the resulting model can be run on any platform that supports ONNX. This allows for easy portability and deployment of the model.

ONNX quantization:

When ConvNeXt-T is quantized using ONNX, the resulting model is smaller and faster but has a slightly lower accuracy than the original model.

Torchscript conversion:

When ConvNeXt-T is converted to Torchscript, the resulting model can be executed on any platform that supports PyTorch. This format provides excellent performance and flexibility but may require additional code modifications to work efficiently on certain platforms.

Conclusion

ONNX quantization can significantly reduce the model's memory footprint and increase inference speed but may result in a slightly lower accuracy than the original model. Overall, choosing the appropriate model architecture and conversion technique depends on the specific use case and platform requirements.

Note: All the Models are trained for 2 epochs due to Memory Constraints. Google Colab keeps on crashing if we train the model for more than 2 epochs while profile the model using Tensorboard