Comparing Depth-wise Separable Convolution and Ordinary Convolution

Calvin Yu
University of Toronto
cal.yu@mail.utoronto.ca

Ziyi Zhang
University of Toronto
ziyierin.zhang@mail.utoronto.ca

Huiting Chen
University of Toront
ht.chen@mail.utoronto.ca

Abstract

The present study compares two MobileNet[1] like networks that is differed on by the use of depth-wise separable convolution versus ordinary convolution using metrics such as classification accuracy, training time, and test time. When trained using a smaller data set such as on CIFAR, the two networks performs comparably but the one using depth-wise separable convolution takes significantly more time to train. In return, we discovered that the network using depth-wise separable convolution has a lower test time when the input is reasonably large which matches the discovery of MobileNet[1]. However, we also discovered an anomaly for extremely small input where the network with ordinary convolution actually takes less time to run on test.

1 Introduction

Convolution neural networks revolutionized the field of computer vision including image classification and recognition, with the introduction of networks such as AlexNet[2]. However, in the process of pursuing higher performance in competitions such as ImageNet, the complexity and depth of convolution neural networks have grown significantly and made computation cost impractical. Many solutions have come up to improve the computation cost, one of them is the use of depth-wise separable convolutions. The use of depth-wise separable convolutions in models such as MobileNet[1] have presented promising results in reducing the size of the model significantly with a relatively small reduction in performance. The present study aims to compare depth-wise separable convolution and ordinary convolution with similar structure in performance and efficiency on smaller data sets using a MobileNet like network.

2 Related Works

2.1 AlexNet

AlexNet is a CNN that is the winner of the ImageNet Large Scale Visual Recognition Challenge in 2012 with top-5 error of 15.3%. But with high accuracy, AlexNet is expensive to apply and train. Since this network has more than 60 million parameters and was trained with 1.2 million training images, they could only use GPUs, paired with a highly-optimized implementation of 2D convolution, to train. And except using the commonly used activation function tanh at that time, it uses non-saturating non-linearity ReLU to decrease the training time. Also, by applying overlapping pooling and data augmentation, they are able to improve accuracy and reduce overfitting. [2]

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

2.2 MobileNet

MobileNet is a neural net based on depth-wise separable convolutions which is described in more detail in section 3.1. It aims to reduce the calculation and simplify the model so that it can fit the requirement of mobile usage. Comparing to full convolution MobileNet, the depth-wise separable MobileNet have similar accuracy on ImageNet, as 71.7% for Full Convolution and 70.6% for depth-wise separable convolution. However, depth-wise separable MobileNet can reduce the mult-adds from 4866 million to 569 million, and reduced the number of parameters from 29.3 million to 4.2 million, which significantly improves the model size and training cost. And by introducing width multiplier α and resolution multiplier ρ, this cost can be even smaller, but with deduction on accuracy.[1]

3 Methods and Algorithm

3.1 Depth-wise Separable Convolution

The present study aims to compare depth-wise separable convolution with ordinary convolution. Depth-wise separable convolution separates the convolution process into two parts, depth-wise convolution and point-wise convolution[1]. Given an M input channels and N output channels, the standards convolution algorithm applies N kernels of dimension $D_k * D_k * M$. The depth-wise separable convolution separates the convolution process into depth-wise convolution and point-wise convolution. Depth-wise convolution applies M independent kernels of dimension $D_k * D_k * 1$ to each of the M input channels. Since the convolution is applied independently in each input channel, the output channel of depth-wise convolution is also M channel. Point-wise convolution is a standard convolution with a $1 * 1$ kernel which combine the output of depth-wise convolution and transforms the M input channels into N output channels.

Assuming an output feature map dimension of $D_f * D_f$ (width * height), the number of connections of standard convolution is $D_k * D_k * M * N * D_f * D_f$. Under the same assumption, the number of connections of depth-wise convolution is $D_k * D_k * M * D_f * D_f$ since it is equivalent to applying M convolution with 1 input channel and 1 output channel and the number of connections of point-wise convolution is $M * N * D_f * D_f$ since it is equivalent to a convolution with $1 * 1$ kernel. So, in total, the connections in a depth-wise separable convolution is $D_k * D_k * M * D_f * D_f + M * N * D_f * D_f$.

This reduce in number of connections should significantly reduce the computation cost of the network at both training and test time and aid the target of creating a more efficient network. However, since there is less connections, it is expected that the network has less expressive power and the classification accuracy may be adversely effected.

3.2 Convolution Layers

There are a total of three types of convolution used in the present study: Depth-wise Separable Convolution, Ordinary Convolution, and Transposed Convolution. For depth-wise separable convolution, there is a batch norm and a ReLU layer between the depth-wise and point-wise convolution and after the point-wise convolution. For the other two convolutions, there is a batch norm and a ReLU layer after the convolution layer. The structures of all three layers are shown in figure 1. In the following parts, reference to the convolution layers will assume that that the batch norm and ReLU activation is included in the layer.

3.3 Network Structure

The two network studied in the present study is referenced as DepthwiseConvNet and ConvNet. DepthwiseConvNet utilizes primarily depth-wise separable convolution layers while ConvNet utilizes primarily ordinary convolution.

Both network have similar structures that is based on MobileNet[1]. DepthwiseConvNet is exactly the same as MobileNet except for the first and last layer while ConvNet has exactly the same input channel, output channel, and relative feature dimension as MobileNet for those layers but with normal convolution instead of depth-wise separable convolution.

For the first layer, MobileNet uses a normal convolution with $3 * 3$ kernel with stride 2 which decrease the width and height of output features by a factor of 2 compared to that of input features. Since

the input image used in this study is much smaller than that used in the original study($32 * 32$ instead of $224 * 224$), DepthwiseConvNet and ConvNet replaces the convolution layer to a transposed convolution layer that increases the width and height of feature dimension by a factor of 2 to compensate for the smaller input size.

The final layer is replaced with a fully connected layer with 1024 input units. The change is made compared to MobileNet to match the number of classes in the data set.

The similarity of the structure of DepthwiseConvNet and ConvNet allows us to study the change brought by only the use of depth-wise separable convolution as the only difference between the two networks is that one uses normal convolution where the other uses depth-wise separable convolution.

The detailed layer by layer structure is shown in figure 2 and figure 3.

4 Experiment and Discussions

The notebook for the experiments can be found on: https://github.com/Calvin0722/CSC413-Project

4.1 Test Accuracy

One of the most important metric to consider in a classification task is the test accuracy of the network. This study will run the network on CIFAR10 and CIFAR100 data sets and compare the test accuracy. Both network has hyperparmeter tuned using linear search and is trained with stochastic gradient descent with weight decay normalization and learning rate decay. The number of parameters in ConvNet is 28377572, and the number of parameters in DepthwiseConvNet is 3320420, so we expect ConvNet to have a higher test accuracy than DepthwiseConvNet since ConvNet has higher capacity. This is also supported by the result of MobileNet[1] but it is also possible that DepthwiseConvNet has a higher test accuracy in this experiment due to the smaller size of CIFAR.

For CIFAR10, The DepthwiseConvNet achieves validation accuracy of 89.940% and ConvNet achieves a validation accuracy of 88.08%, so DepthwiseConvNet has a slightly higher validation accuracy than ConvNet but the two difference in the two networks are extremely small. Detailed training loss and accuracy graph is shown in figures 4 to 7. The result is almost identical for the two networks which might be due to the fact that the task is not challenging enough to differentiate the two networks.

For CIFAR100, The DepthwiseConvNet achieves validation accuracy of 68.990% and ConvNet achieves a validation accuracy of 67.230%, so the 2 models has roughly the same validation accuracy. Detailed training loss and validation graph can be found in Appendix section, figures 8 to 11. Similar to the result of CIFAR10, this might be the due to the complexity of the problem. Another possible explanation is that ConvNet might have overfitting when it is trained with low resolution images ($32 \times 32 \times 3$).

4.2 Training Time

Training time is also an important metric to consider when designing a network since longer training time will lead to more computation cost on training. We expect DepthwiseConvNet to takes less time per epoch than ConvNet, since DepthwiseConvNet has lower computation complexity as analysed in section 3.1.

As expected, the training time per epoch for depth-wise separable neural network is only half as much as the standard convolution. Standard convolution takes around 33.2 seconds per epoch, while depth-wise separable neural network takes only 17.7 seconds per epoch, so the training time halved by using the latter.

For the total training time on CIFAR100, DepthwiseConvNet takes 270 epochs to train and the total training time is 5322.46 seconds while ConvNet takes 80 epochs and and the total training time is 1770.90 seconds. So DepthwiseConvNet has a longer total training time than the ConvNet. One possible explanation for the larger amount of epochs needed to train DepthwiseConvNet is that depth-wise separable convolution is separated into depth-wise convolution and point-wise convolution. This caused the DepthwiseConvNet to be significantly deeper than ConvNet and vanishing gradient might play a role during the training process.

4.3 Test Time

Test time is the main concern that depth-wise convolution is trying to address in MobileNet[1] since it is the recurring cost that will be incurred every time the network is used. After training the two networks, the network is ran and timed on both the train and test set to evaluate the efficiency of the two networks. The training set contains 50,000 images of $32 * 32 * 3$ and the test set contains 10,000 images of $32 * 32 * 3$. For each network and each data set, the network is ran 10 times repetitively and the average of the recorded time in the 10 iterations will be considered as the time spent to decrease variance in the data collected. As explained in section 3.1, because the value of M and N are large and $D_k = 3$ in the two networks, the amount of connections and thus the amount of computation in the DepthwiseConvNet is significantly smaller compared to ConvNet. So, it is expected that the amount of time spent at test time is significantly lower for DepthwiseConvNet compared to ConvNet.

The experiment is executed using a data loader of batch size 256. For DepthwiseConvNet, it takes 1.98 seconds to go through the test set and 9.15 seconds to go through the training set. For ConvNet, it takes 2.17 seconds to go through the test set and 10.09 seconds to go through the test set. So, it takes 9.5% and 10.3% more time for ConvNet to run compared to DepthwiseConvNet on test set and train set respectively. The result matches our expectation that DepthwiseConvNet will perform better than ConvNet in this part. However, the difference that is observed is less significant than expected. This might be explained by the fact that DepthwiseConvNet has less parallelize computation compared to ConvNet since the convolution process is separated into smaller serialized operations. Also, there are extra batch norm and ReLU inside the depth-wise separable convolution compared to normal convolution which might also contribute to the small time difference between the two networks.

The fact that there is less parallel computation in DepthwiseConvNet also lead to a potential situation when the input is not able to saturate the parallel computing capability of the device. Since there is more serialized operation in DepthwiseConvNet, the hypothesis is that this may lead to a situation where it may take DepthwiseConvNet longer to run compared to ConvNet.

To simulate such a situation, the experiment above is carried out again with a data loader with batch size of 4 to decrease the amount of parallel computation. For DepthwiseConvNet, it takes 15.22 seconds to go though the test set and 75.18 seconds to go through the the training set. For ConvNet, it takes 12.82 seconds to go through the test set and 63.84 seconds to go through the test set. This means that it takes 18.7% and 17.8% more time for DepthwiseConvNet to run compared to ConvNet in test set and training set. This result matches with the hypothesis that when the input is small enough, because there is less parallel computation and DepthwiseConvNet is more serialized, the computation time for ConvNet will actually be less than DepthwiseConvNet even though there is theoretically much more connections and computations in ConvNet.

5 Summary

From the result of the experiments, with a smaller data set such as CIFAR, the use of depth-wise separable convolution and ordinary convolution provides a similar level of classification accuracy on test time. This differs from the result from MobileNet where using ordinary convolution provides a better classification accuracy. For the training time, it takes less time for DepthwiseConvNet to run each epoch but take longer to train the whole model due to the need to run more epochs compared to ConvNet. For test time, DepthwiseConvNet takes shorter amount of time when batch size is reasonably large but the study also discovered a situation where DepthwiseConvNet takes longer to run when input is extremely small.

One limitation that we expect in the result is the size of the data set used. The networks used is based on MobileNet[1] which is designed for a much larger data set and the smaller data set may eliminate some of the difference discovered in that study. Another limitation is that only a pair of models that are almost identical in structure and is differed only in the use of the type of convolution layer is tested in this study due to limitation in time and computation power. The use of a similar structure limits the scope of the conclusion that we will be able to make since it is possible that the two types of convolution behave differently in different structures. Future studies can investigate the generalization ability of the conclusion of the current study by comparing the two type of convolution behave in different network structures or explore if the result from this study holds when using larger data set.

Attribution

Abstract: Calvin Yu

Introduction: Calvin Yu, Ziyi Zhang

Related Works: Ziyi Zhang

Method and Algorithm: Huiting Chen, Calvin Yu

Experiment and Discussion

-Test Accuracy: Calvin Yu, Huiting Chen

-Training Time: Huiting Chen

-Test Time: Calvin Yu

Summary: Calvin Yu

Code: Calvin Yu

References

[1] Andrew, G.H., Menglong, Z., Bo, C., Dmitry, K., Weijun, W., Tobias, W., Marco, A. & Hartwig, A. (2017) MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. https://arxiv.org/pdf/1704.04861v1.pdf.

[2] Alex, K., Ilya, S. & Geoffrey, E.H. (2012) ImageNet Classification with Deep Convolutional Neural Networks. https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

Appendix

Normal Convolution:

3x3 Normal Convolution
Batch Normalization
ReLU

Depth-wise Separable Convolution:

3x3 Depthwise Convolution
Batch Normalization
ReLU
1x1 Pointwise Convolution
Batch Normalization
ReLU

Transposed Convolution:

3x3 Transposed Convolution
Batch Normalization
ReLU

Figure 1: Structure of convolution layers used

Layer Type	Input Channels	Output Channels
Transposed Conv	3	32
Normal Conv	32	64
Normal Conv	64	128
Normal Conv	128	128
Normal Conv	128	256
Normal Conv	256	256
Normal Conv	256	512
Normal Conv	512	512
Normal Conv	512	512
Normal Conv	512	512
Normal Conv	512	512
Normal Conv	512	512
Normal Conv	512	1024
Normal Conv	1024	1024
Adaptive Average Pool	1024	1024
Linear Layer	1024	Classes

Figure 2: Structure of ConvNet

Layer Type	Input Channels	Output Channels
Transposed Conv	3	32
Depth-wise Separable Conv	32	64
Depth-wise Separable Conv	64	128
Depth-wise Separable Conv	128	128
Depth-wise Separable Conv	128	256
Depth-wise Separable Conv	256	256
Depth-wise Separable Conv	256	512
Depth-wise Separable Conv	512	512
Depth-wise Separable Conv	512	512
Depth-wise Separable Conv	512	512
Depth-wise Separable Conv	512	512
Depth-wise Separable Conv	512	512
Depth-wise Separable Conv	512	1024
Depth-wise Separable Conv	1024	1024
Adaptive Average Pool	1024	1024
Linear Layer	1024	Classes

Figure 3: Structure of DepthwiseConvNet

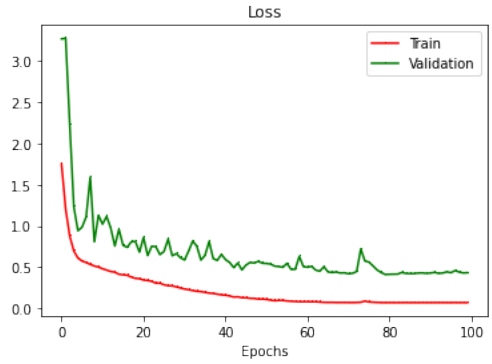

Figure 4: DepthwiseConvNet training loss on CIFAR10

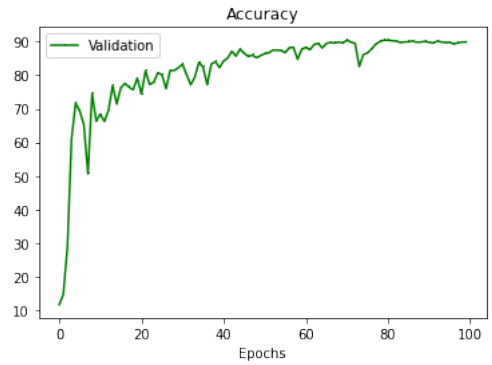

Figure 5: DepthwiseConvNet validation accuracy on CIFAR10

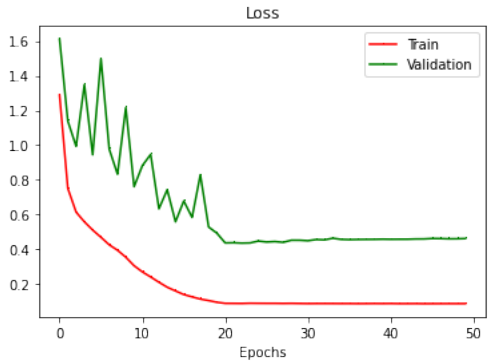

Figure 6: ConvNet training loss on CIFAR10

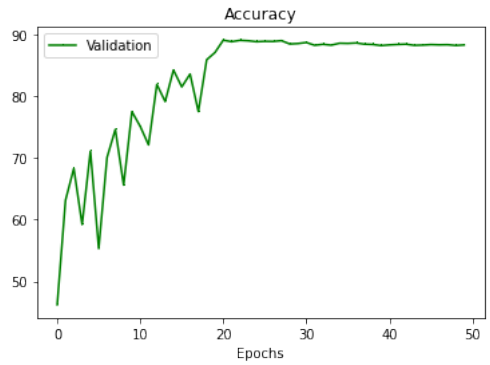

Figure 7: ConvNet validation accuracy on CIFAR10

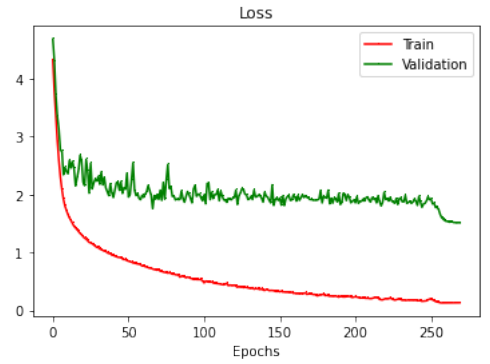

Figure 8: DepthwiseConvNet training loss on CIFAR100

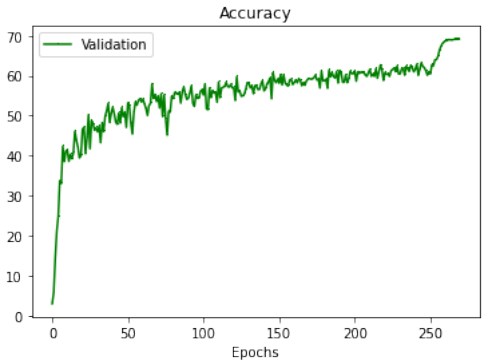

Figure 9: DepthwiseConvNet validation accuracy on CIFAR100

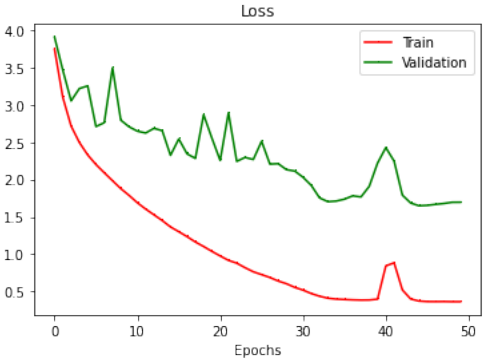

Figure 10: ConvNet training loss on CIFAR100

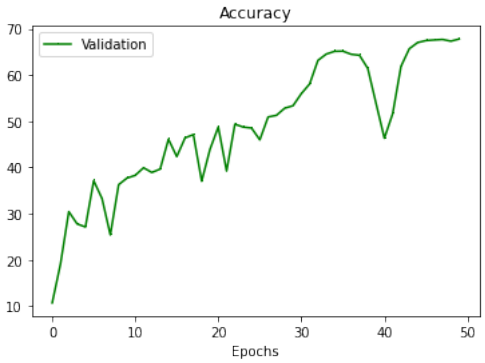

Figure 11: ConvNet validation accuracy on CIFAR100