IR-NET

NIKHIL DARWIN BOLLEPALLI
NXB200019@utdallas.edu

Abstract

Over the past years, CNNs have become more feasible for usage in embedded and mobile devices. Many tools have further accelerated opportunities for the application of deep learning in embedded devices. This led to various CNN architectures with many derived and improved version from basic class of standard ImageNet design those are suitable for mobiles and embedded devices. The design described in this paper is based on such standard ImageNet structures which is flexible and utilizes the bottlenecks that we have seen in mobileNet by reducing the number of parameters

1. Introduction

Accuracy is good metric to compare different models, but accuracy is not the only metric one should be concerned while evaluating x-NN based models. X-NNs are already notorious for their huge training time, power and compute resources. Striking the right balance between performance and accuracy based on the use case scenario is the best strategy. It is very important sometimes to make a decision in a short time interval rather than wait and rely on more accurate decision by just few decimals and vice-versa.

2. Related Work

The network in this paper was heavily inspired by SENet [1], MobileNetV2 [2], MnasNet [3], CondConv [4], MobileNetV3 [5] and EfficientNet [6].

MobileNetV2 [2] used inverted residuals to improve performance of mobile models on multiple tasks and benchmarks. Non-linearities in narrow layers are removed in order to maintain representational power.
MnasNet [3] added squeeze and excite operations to the inverted residuals and used neural architecture search space with best tradeoffs between accuracy and latency to optimize the network depth and width.
MobileNetV3 [5] replaced the swish nonlinearity with a hard swish nonlinearity which improved the design of the final encoder stages and used a new neural architecture search to optimize the depth and width.
EfficientNet [6] refined the mobile baseline network and focused on the combined scaling of network input resolution, depth and width to design larger networks.
A variant of EfficientNet replaced the convolution operations with conditional convolution operations [4] which improves the performance and inference cost trade-off of several existing convolutional neuralnetwork architectures on both classification and detection tasks.

3. Design

This design follows relatively a standard ImageNet structure as shown in Figure 1and Table 1 where the first stride of length 2 operations is replaced with stride by length of 1 as the images used in training had been down sampled by factor of 4 which leads to quarte of rows and columns of a typical ImageNet image.
The design describes about three different types of building blocks which involves a basic inverted residual block (IRB), IRB with squeeze and excitation block (Figure 3) and finally enhanced IRB with conditional convolution block(Figure 4). Introduction of theses enhanced blocks didn't disrupt any network layer causing dimensional operations mismatch and therefore can be used interchangeably in the table1 described below.
It can also be noted that differences with this design and EfficientNet is that this design used only 3x3 filters in the fully grouped convolution and modified the stem width, depth and width of various blocks. The design described also modified the inverted residual expansion ratio and simplified the nonlinearity choice to ReLU (or Sigmoid where its appropriate).

For implementations based on the SE enhanced building block, the internal rank reduction ratio R = 4.

For implementations based on SE and conditional convolution enhanced building blocks, the number of experts M = 4.

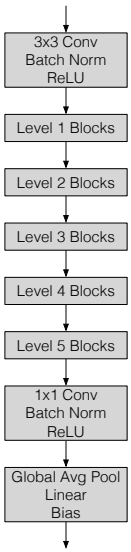


Figure 1: Network structure; the linear layer output dimension and bias dimension is the number of classes

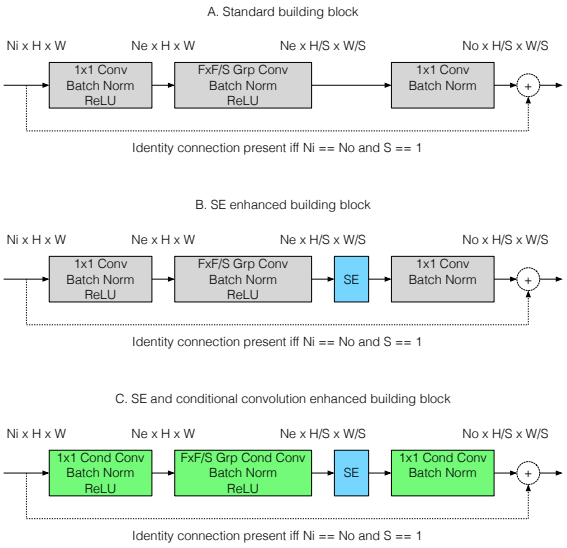


Figure 2: [A] Standard building block, [B] SE enhanced building block and [C] SE and conditional convolution enhanced building block

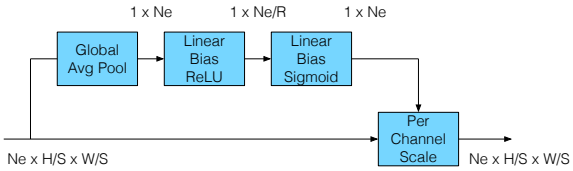


Figure 3: Squeeze and excite uses a learned input dependent per channel weighting to re weight input feature maps with internal rank reduction R

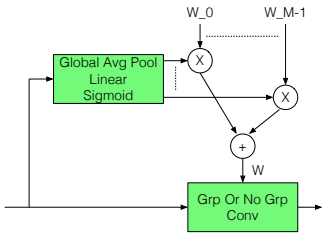


Figure 4: Conditional convolution uses 3D / 4D convolution weight tensors W_m from M experts combined to a single 3D / 4D weight tensor W via a learned input dependent weighted sum with fully grouped convolution / not fully grouped convolution

Re-peat	Input NixWxH	Operation
1	3x56x56	Conv (3x3/1), Batch Norm, ReLU
1	16x56x56	Block (Ne=4Ni, F=3, S=1, ID=True)
1	16x56x56	Block (Ne=4Ni, F=3, S=1, ID=False)
1	24x56x56	Block (Ne=4Ni, F=3, S=1, ID=True)
1	24x56x56	Block (Ne=4Ni, F=3, S=2, ID=False)
2	40x28x28	Block (Ne=4Ni, F=3, S=1, ID=True)
1	40x28x28	Block (Ne=4Ni, F=3, S=2, ID=False)
3	80x14x14	Block (Ne=4Ni, F=3, S=1, ID=True)
1	80x14x14	Block (Ne=4Ni, F=3, S=2, ID=False)
4	160x7x7	Block (Ne=4Ni, F=3, S=1, ID=True)
1	160x7x7	Block (Ne=4Ni, F=3, S=1, ID=False)
1	320x7x7	Conv (1x1/1), Batch Norm, ReLU
1	1280x7x7	Global Avg Pool, Linear, Bias
1	1x100	Output

Table 1: Network specification; block is either a [A] standard building block, [B] SE enhanced building block or [C] conditional convolution enhanced building block

4. Training

Table 2 includes a summary of all training hyper parameters. Note that this is a ~ generic ImageNet training routine such as one would find in RegNetX/Y [7]. Training routines that use more complex data augmentation, additional data, different train and test resolutions, more epochs, can achieve higher accuracies.

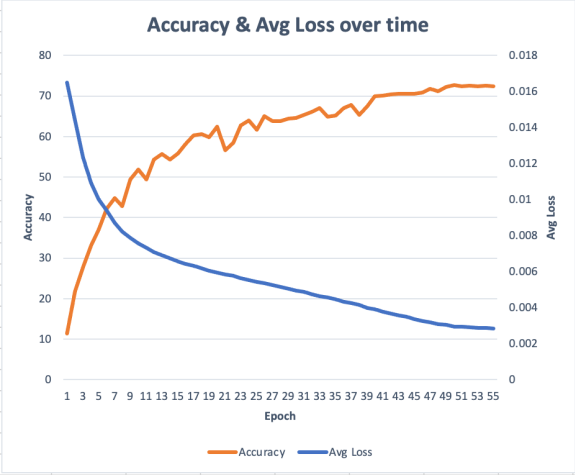
Table 3 includes final training results and figure 5 shows a plot of the per epoch accuracy and loss curves.

Parameter	Value
TRAIN_LR_MAX	0.2
TRAIN_LR_INIT_SCALE	0.01
TRAIN_LR_FINAL_SCALE	0.001
TRAIN_LR_INIT_EPOCHS	5
TRAIN_LR_FINAL_EPOCHS	50
TRAIN_NUM_EPOCHS	55
DATA_BATCH_SIZE	256
DATA_NUM_CLASSES	100

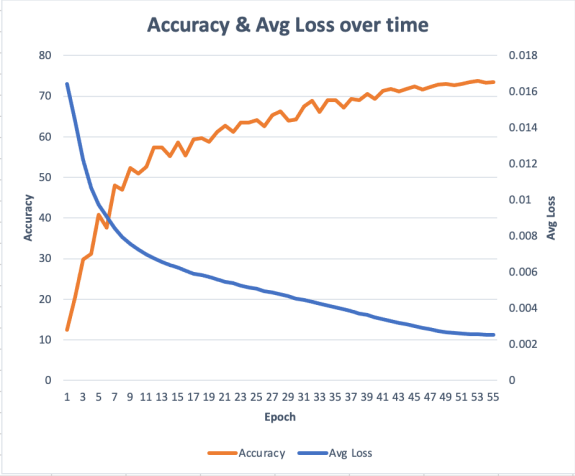
Table 2: Training hyper parameters

Block	Training time	Accuracy
Standard	5.5 hours	72.54
SE enhanced	6 hours	73.72
SE and cond conv enhanced		

Table 3: Training final results



Training results with simple inverted residual block



Training results with SE block

Figure 5: Training per epoch accuracy and loss curves

5. Implementation

Table 4 shows per operation MACs and number of filter coefficients for the stem convolution, convolutions in all standard blocks (taking into account repeats) and the head convolution and matrix multiplication, along with their sum for the full network.

Operation	Rep	MAC	Filter Cx
Conv (3x3/1), Batch Norm, ReLU	1	1354752	432
Block (Ne=4Ni, F=3, S=1, ID=True)	1	8228864	38912
Block (Ne=4Ni, F=3, S=1, ID=False)	1	9834496	39424
Block (Ne=4Ni, F=3, S=1, ID=True)	1	17160192	87552
Block (Ne=4Ni, F=3, S=2, ID=False)	1	5494272	89088
Block (Ne=4Ni, F=3, S=1, ID=True)	2	22328320	486400
Block (Ne=4Ni, F=3, S=2, ID=False)	1	7808640	249600
Block (Ne=4Ni, F=3, S=1, ID=True)	3	31799040	2918400
Block (Ne=4Ni, F=3, S=2, ID=False)	1	7667520	998400
Block (Ne=4Ni, F=3, S=1, ID=True)	4	41269760	15564800
Block (Ne=4Ni, F=3, S=1, ID=False)	1	15335040	3993600
Conv (1x1/1), Batch Norm, ReLU	1	20070400	409600
Global Avg Pool, Linear, Bias	1	128000	0
Total	–	193898304	24876208

Table 4: Per operation and total MAC and filter coefficient counts for all trainable operations

6. Conclusion

This paper describes a CNN architecture that is based on standard ImageNet structure which is flexible and achieves decent accuracies compared to a typical ImageNet CNN with not so many parameters required for training with inverted residual block which follows narrow-wide-narrow structure.

In future this block can be modified with different filter sizes, width & depth, repeating blocks, residual expansion factors, SE rank reduction factors and number of mixtures of experts while implementing conditional grouped convolution.

References

[1] J. Hu et. al., "Squeeze-and-excitation networks," arXiv:1709.01507, 2017.

[2] M. Sandler et. al., "MobileNetV2: inverted residuals and linear bottlenecks," axXiv:1801.04381, 2018.

[3] M. Tan et. al., "MnasNet: platform-aware neural architecture search for mobile," arXiv:1807.11626, 2018.

[4] B. Yang et. al., "CondConv: conditionally parameterized convolutions for efficient inference," arXiv:1904.04971, 2019.

[5] A. Howard et. al., "Searching for MobileNetV3," arXiv:1905.02244, 2019.

[6] M. Tan and Q. Le, "EfficientNet: rethinking model scaling for convolutional neural networks," arXiv:1905.11946, 2019.

[7] I. Radosavovic et. al., "Designing network design spaces," arXiv:2003.13678, 2020.