Investigating the effectiveness of feature recalibration networks on image classification

Kevin Wang
University of Toronto
imkevin.wang@mail.utoronto.ca

Chang Yuan
University of Toronto
chang.yuan@mail.utoronto.ca

Abstract

In this study, we investigate the effectiveness of Squeeze-and-Excitation (SE) and Skip-Layer-Excitation (SLE) modules in image classification tasks. While SLE modules were originally developed for Generative Adversarial Networks (GANs), their potential impact on image classification remains unexplored. We compare the performance and efficiency of both SE and SLE modules when applied to a ShuffleNetV2 architecture on the CIFAR10 dataset. Our results demonstrate that both SE and SLE modules improve classification performance, with SE modules offering a more favourable performance-efficiency trade-off.

1 Introduction

Convolutional neural networks are used in a variety of fields, for tasks such as classification, and image generation[6].

Feature recalibration modules have been shown to increase performance in state-of-the-art convolutional networks, helping model dependencies between channels[3]. We aim to investigate the effectiveness of two such modules: Squeeze-and-Excitation(SE) blocks, as well as its variant, Skip-Layer-Excitation(SLE) blocks.

SLE blocks were first proposed to create an effective and fast GAN model. GANs are generative networks that commonly use convolutional architectures, and face issues with unstable training specific to its class of models[1]. To address these issues, novel techniques have been proposed to improve the speed and gradient flow in GANs. Many of these techniques can also be applied to other convolutional networks, but the literature on this subject is limited.

Thus, this study aims to analyze the effectiveness of SLE modules, originally proposed for GANs, when applied to other CNNs, and compare it with baseline architectures. We will analyze its effect on the performance and efficiency of a ShuffleNetV2 image classification model, which is optimized for performance-efficiency tradeoffs.

2 Related Works

The goal of efficient CNNs has given rise to many lightweight architectures with speed-accuracy tradeoffs, such as depth-wise convolutions and grouped convolutions. ShufflenetV2 is a model that is built around many guidelines for efficient network design, and is optimized for speed, while retaining relatively high accuracy on many computer vision tasks [6]. Researchers continue to strive to produce novel architectures that give reasonable speed-accuracy tradeoffs.

Squeeze-and-excitation networks are channel-wise feature recalibration modules that can be added after convolutional blocks of a network. Each module takes a set of feature maps as input, and outputs a vector of the same dimension that is used to scale the feature maps, thus modelling interdependencies

CSC413 Final Project.

between them. It's been shown that augmenting existing architectures with these modules decreases the error rate of many convolutional networks, without requiring many additional computations[3].

Skip-layer excitation modules are a variant of the SE network, that uses channel-wise feature recalibration to allow long-range skip-connections between layers. While the SE network augments a single convolutional block, the SLE network can connect two convolutional blocks with different numbers of channels, and different spatial dimensions. The module aggregates channel-wise information from one layer in the convolutional network into a vector, that is used to re-weight feature maps in another, resulting in better results in few-shot learning with GANs [5].

3 Methods

We used a ShufflenetV2 architecture as our baseline model. We created a variation of it, augmented with skip-layer excitation modules and another variation augmented with squeeze-and-excite modules, totalling three models. We analyzed each model's performance and inference speed on the CIFAR10 dataset.

The ShufflenetV2 architecture is broken down into stages of residual blocks that vary in the spatial dimension. In total, there are 5 stages.

The SE model will augment the base model with squeeze-and-excitation blocks after each stage. We also considered adding SE modules after each residual block, but it led to lower performance in our preliminary experiments.

The SLE model will augment the base model with skip-layer excitation blocks between stages 1 and 2, 2 and 3 and 3 and 4. Below are the algorithms for the augmentations. Note that σ represents the sigmoid function, $\odot$ represents element/channel-wise multiplication, and each W_i and b_i are the weights and biases of a fully-connected layer.

Algorithm 1 SE block, after stage i

```
$x$ is output of stage i
$x' \leftarrow$ channel-wise mean($x$)
$x' \leftarrow reLU(xW_0 + b_0)$
$x' \leftarrow \sigma(xW_1 + b_1)$
$x = x' \odot x$
```

Algorithm 2 SLE, between stages i and j

```
Require: $i < j$
$x$ is output of stage i
$y$ is output of stage j
$x \leftarrow$ AdaptivePool($x$, $4 \times 4$)
$x \leftarrow$ Conv2d($x$, kernel_size=4)
$x \leftarrow x \odot \sigma(x)$ (siLU)
$x \leftarrow \sigma(xW + b)$
$y = x \odot y$
```

4 Experiments

We used a ShuffleNetV2 implementation with less pooling, as CIFAR10 contains relatively small images. We also used the $0.5x$ model with fewer intermediate output channels due to time constraints. We used Kaiming initialization and the Adam optimizer for training[2, 4]. We used the Adam parameters of $\beta_0 = 0.9, \beta_1 = 0.999$, and a weight decay rate of $1e - 4$ for all trials.

We used negative log-likelihood as the loss function, as it is common in classification tasks.

$\mathcal{L}_{NLL} = -\mathbb{E}[\sum_i p(t_i) log(\mathcal{F}(\mathbf{x})_i)]$

In NLL, t is the target vector of class probabilities, and $\mathcal{F}$ is the processing of the classification network on the input $\mathbf{x}$.

For each model, we performed sensitivity analysis on learning rate and batch size, using grid search with learning rates of 1e-2, 1e-3 and 1e-4, and batch sizes of 32, 64 and 128.

We evaluated the performance of the resulting models using the highest top-1 validation accuracy achieved during training. We evaluated model efficiency using the number of parameters, the number of multiply-and-aggregate calculations(Macs), and image/batch speed during inference on a NVIDIA GeForce RTX 3080 GPU. We used the 'ptflops' library to measure Macs and parameters, to reduce human error[7]. We averaged the results of multiple trials of inference experiments to reduce bias in our results.

5 Results

Model	Learning rate	Batch size	Maximum top1 validation accuracy
ShuffleNetV2+SE	0.001	64	$90.26_{(+0.29)}$
ShuffleNetV2+SLE	0.001	64	$90.19_{(+0.22)}$
ShuffleNetV2	0.001	64	89.97

Table 1: Performance results of the three models, using optimal batch sizes and learning rates ordered by maximum validation accuracy achieved during training

We reached 89.97 validation accuracy in our reimplementation of ShuffleNetV2, our baseline model for this investigation. We see that adding SLE and SE modules resulted in a higher accuracy. We found the optimal hyperparameters for all three architectures was a learning rate of 1e-3 and batch size of 64.

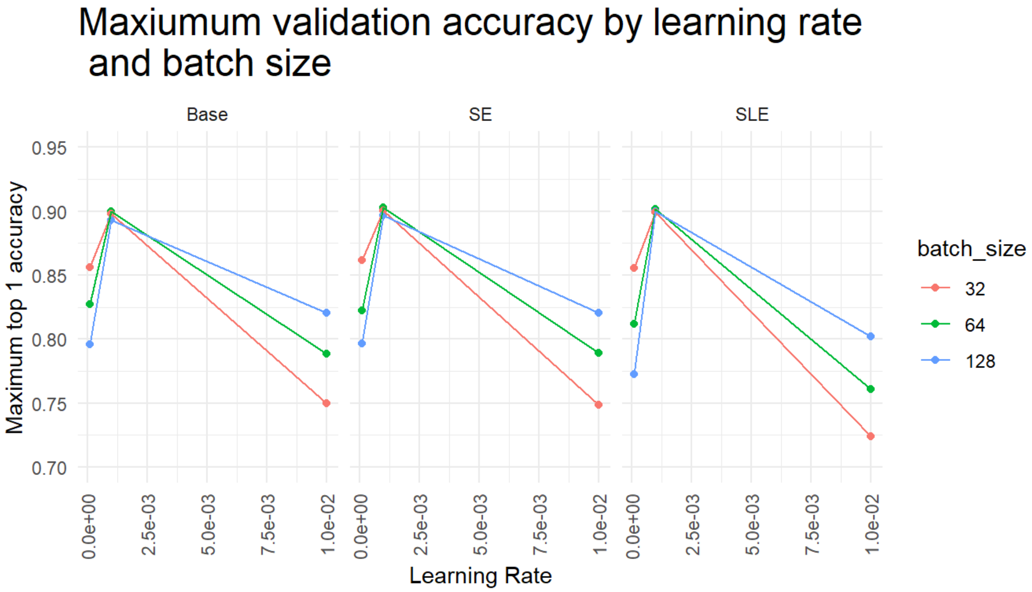

Figure 1: Maximum validation accuracy by learning rate and batch size, across the three models.

We found that batch size and learning rate had similar effects across the three models. The smallest batch size is preferred when using a lower learning rate, whereas the largest batch size is preferred when using a higher learning rate.

We found that the SE and SLE models had higher sensitivity to changes in hyperparameters, typically seeing worse performance when using suboptimal learning rates, and seeing larger variance in performance when altering the batch sizes.

Model	#Parameters	MMac	Inference images/sec	Inference batches/sec
ShuffleNetV2	$1.264e6$	$4.613e7$	8113.59	127.38
ShuffleNetV2+SLE	$3.744e6$	$4.868e7$	7382.32	115.90
ShuffleNetV2+SE	$1.494e6$	$4.650e7$	6353.53	99.75

Table 2: Efficiency results of the three models, when running inference on the CIFAR10 test set(batch size 64). Ordered by images/sec

We see that the base model and the SE model have similar numbers of parameters and multiply-and-aggregate operations. However, the SLE model has around 2.5 times the parameters as the base model and higher Mac operations than the other two models.

The base model had the fastest inference times, followed by the SLE model then the SE model. The SLE model processed 91.0% as many batches per second as the base model, and the SE model processed 78.3%.

6 Discussion

We managed to reasonably reproduce the accuracy results shown in the ShuffleNet paper. Although the paper used the ImageNet dataset, we were able to achieve relatively high accuracy on the CIFAR10 dataset using the base model. We were also able to reproduce the results given by squeeze-and-excitation networks, demonstrating an increase in validation accuracy.

Firstly, our results show the SE network had the highest improvement in validation accuracy, while retaining a similar number of parameters compared to the base model. However, we observed that it processes fewer images/batches per second compared to the SLE model, despite its smaller size. This may be because convolutional operations are more easily parallelized. Although the SLE model has more parameters and computations, many of the additional computations can be parallelized. Overall, the SLE model has fewer additional blocks compared to the SE model and runs faster on GPU.

The strengths of SE modules are that they do not add much complexity to the base model, are easy to integrate into existing architectures, and model dependencies between channels, which has been shown to be a limitation in existing CNNs.

Secondly, our results show the SLE network also led to higher validation accuracy. However, it dramatically increased the number of parameters in the network. Thus, we believe it does not offer a favourable performance-efficiency tradeoff when compared to the base network, and underperforms compared to the existing alternative, the SE network.

We believe the main weakness of this block is that it was originally intended to solve problems in GANs, and is less effective when applied to other CNNs. The SLE block facilitates communication between layers, strengthening gradient flow through the network. However, gradient flow is less important in our model due to the small model size, and it is already somewhat addressed with channel splits and shuffling in the base model.

In addition, SLEs were shown to decouple style and content in GANs[5]. Intuitively, the early layers of our CNN contain low-level features, with later layers modelling higher-level features. However, this is the opposite for GANs, which take a latent vector as input and output an image. Thus, in GANs, the SLEs modify low-level feature maps using earlier "high-level maps," but in our CNN it does the opposite. This is somewhat counterintuitive. Compared to the SE network, the motivations behind the SLE block are less relevant and logical given our task and model architecture.

Finally, a weakness common to both SE and SLE networks is that they impact the speed of the model, and can cause overfitting issues in smaller networks. Fully connected layers have relatively high parameters and time complexity, and thus can significantly impact the speed and efficiency of smaller models, such as the one used in this study.

We observed that the SE and SLE models have slightly higher sensitivity to hyperparameters compared to the base model, supporting the idea that overfitting and regularization are more significant issues than in the baseline. In addition, we noticed substantial increases in inference time and space complexity when comparing them with the base model. Finally, in preliminary trials, we found that adding more SE blocks than that described in the methods section often led to degraded performance and overfitting.

Conclusion

We have explored the effectiveness of SE and SLE networks. Both are shown to marginally improve performance in CNNs, although we prefer the performance-efficiency tradeoff in SE networks.

We believe further work could be done to investigate the effectiveness of these two architectures on larger models, datasets, and alternative tasks. Alternative architectures that combine channel-wise recalibration with skip-connections could also be investigated.

Code for this report can be found here: https://github.com/dungwoong/CSC413Final

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

[3] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017.

[4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[5] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards faster and stabilized GAN training for high-fidelity few-shot image synthesis. *CoRR*, abs/2101.04775, 2021.

[6] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: practical guidelines for efficient CNN architecture design. *CoRR*, abs/1807.11164, 2018.

[7] Vladislav Sovrasov. ptflops: a flops counting tool for neural networks in pytorch framework, 2018.