# Project 4 - Reproducibility of SqueezeNet

Stone Chen, 260654457, Yunwen Ji, 260738554, Yan Miao, 260711311

***Abstract*—In this project, we successfully reproduced SqueezeNet [14] on the CIFAR-10 [2] dataset with the same level of performance as AlexNet but 9x reduction in model size. By exploring the architectural features of SqueezeNet, we gained more insights that were in turn used to further optimize our baseline SqueezeNet. Our optimized SqueezeNet achieved 91.15% validation accuracy while still retaining a model size smaller than that of the AlexNet.**

## I. Introduction

In this project, we aimed at reproducing the success of SqueezeNet [14] on a well-known but smaller dataset, CIFAR-10 [2]. Specifically, we applied the strategies proposed in the original SqueezeNet paper to construct an adapted SqueezeNet that achieved 9x model size reduction compared to the AlexNet, while maintaining the same level of accuracy. Furthermore, we performed explicit ablation studies on the key architectural components of the SqueezeNet. With insights on the quantitative impact of each of those components, we succeeded in constructing an optimized SqueezeNet that reached 91.15% validation accuracy, yet it was still smaller than the AlexNet.

The rest of this report is organized as follows. In section II, we reviewed key concepts of the SqueezeNet. Then in section III, IV and V, we described the setup in our data, model configuration, training and validation, as we trained all the models from scratch. In section VI, we discussed the reproducibility and observations in our training processes. In section VII, we presented the quantitative ablation studies in each of the three main architectural components in SqueezeNet. After that, we used insights gained from section VII to construct an optimized model, presented in section VIII. Lastly, in section IX, we concluded our whole project in terms of the model reproducibility and optimization.

## II. Related Work

Ever since the success of AlexNet [5] that marked the start of deep learning era of machine learning, much of the research on deep convolutional neural networks (CNNs) has focused on building deeper networks to achieve higher accuracy on computer vision datasets [8, 10, 13]. However, for a given accuracy level, there are typically multiple network architectures of varying sizes available. Given the same level of accuracy, those CNN architectures with significantly fewer parameters will in practice be more efficient in its distribution as well as more convenient in its implementation into real-world applications [15].

Provided with these advantages, Iandola et al. (2016) investigated the possibility of minimizing the number of parameters in a CNN architecture that would still achieve equivalent accuracy compared to a well-known model. In particular, they

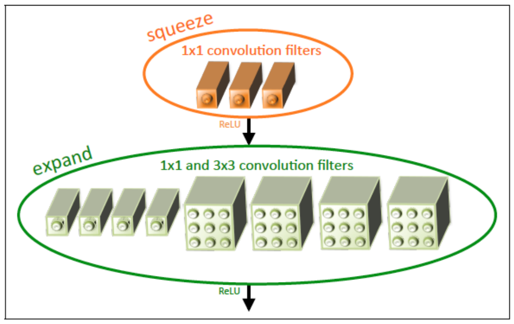


Fig. 1: Iandola et al. (2016). Organization of convolution filters in the fire module. In this example, $s_{1x1} = 3$, $e_{1x1} = 4$, and $e_{3x3} = 4$.

discovered an architecture, named SqueezeNet, that preserved the accuracy of AlexNet [5] on ImageNet dataset [1], despite its model size was reduced by 50 times. Such a huge reduction in size (number of parameters) was accomplished by replacing a subset of $3\times3$ filters with $1\times1$ filters and also decreasing the number of input channels feeding into these $3\times3$ filters. These two strategies were applied by defining the fire module, which is composed of a *squeeze* convolution layer ($1\times1$ convolution filters), feeding into an *expand* layer which contains a mix of $1\times1$ and $3\times3$ convolution filters, illustrated in Figure.1 [14]. There are three tunable hyperparameters associated with this module:

- $s_{1x1}$ = number of filters (all $1\times1$) in the squeeze layer;
- $e_{1x1}$ = number of $1\times1$ filters in the expand layer;
- $e_{3x3}$ = number of $3\times3$ filters in the expand layer;

The CNN architecture that achieved equivalent accuracy to the AlexNet utilized 8 such fire modules with maxpooling layers in-between, flanked by two convolutional layers. In addition, with a budget of parameters, the author also applied a third strategy to downsample late in the network so that convolution layers have larger activation maps, which potentially result in higher accuracy [6]. This is achieved by setting stride parameter in all convolution layers to 1 and placing pooling layers relatively later in the overall network architecture.

## III. Dataset and Setup

The SqueezeNet model proposed in Iandola et al. (2016) was evaluated on the ImageNet [1], which is beyond our computational capacity. Thus we turned to another well-known but significantly smaller dataset CIFAR-10 [2] in reproducing and studying the architectural features of SqueezeNet. The CIFAR-10 dataset consists of 60000 32x32x3 RGB images in 10 different classes, with 6000 images per class. Specifically, there are 50000 training images and 10000 test images already partitioned. Before inputting data to our model, we normalized our data by subtracting and dividing each sample by overall sample mean and standard deviation, respectively for each channel.

In addition, to prevent overfitting and increase generalization performance, we attempted to generate variations based on our current samples. By inspecting a number of existing samples, we realized that they were basically photographs taken from different angles of those ten classes. We came to a list of augmentations that could be done to benefit the training process, including rotation, horizontal and vertical shift, shear, as well as zoom. Those augmentation processes were chosen so that they would not alter the intrinsic properties of those photographed samples; a counter example could be a horizontal flip of a bird which rarely occurs in a photographed scene. To achieve such augmentations on our dataset, we utilized the Image Generator Class from Keras API [11] which applied those specified augmentations randomly during training time. The settings for the Image Data Generator are:

- rotation range=20
- width shift range=0.1
- height shift range=0.1
- shear range=0.5
- zoom range=(0.9,1.1)

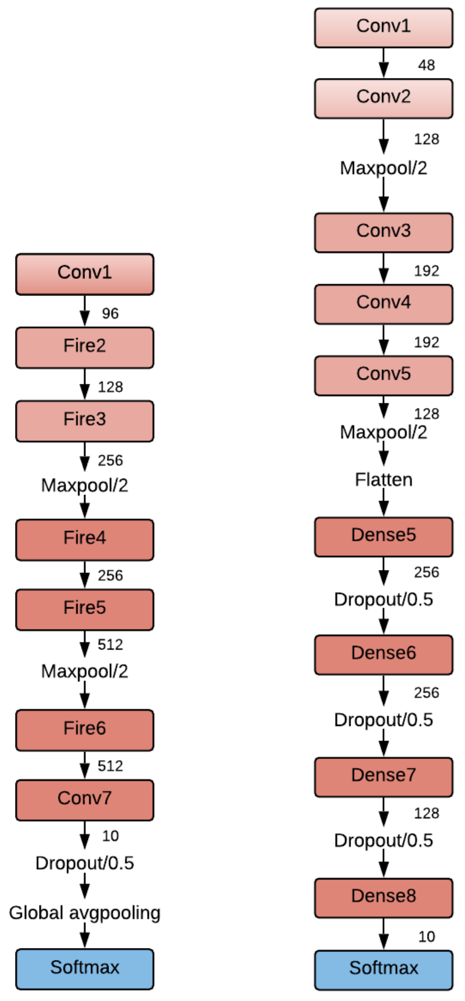
## IV. Model Configurations and Setup

There are a few metaparameters associated with the architecture of a SqueezeNet we need to define first:

- $base_e$ = number of expand filters in the first fire module;
- $e_i$ = number of expand filters in the $ith$ fire module; in particular, $e_i = base_e + (incr_e * \lfloor i/freq \rfloor)$, which means that after every $freq$ fire modules, the number of expand filters is increased by $incr_e$;
- $pct_{3x3} = e_{i,3x3}/e_i$ = number of $3 \times 3$ expand filters in the $ith$ fire module / number of expand filters in the $ith$ fire module;
- $SqueezeRatio(SR) = s_{i,1x1}/e_i$ = number of squeeze filters / number of expand filters.

The original SqueezeNet in Iandola et al. (2016) has the following metaparameters: $base_e = 128, incre = 128, pct_{3x3} = 0.5, freq = 2$, and $SR = 0.125$. However, since we were dealing with CIFAR-10 [2] with much smaller images (32x32x3), the original SqueezeNet model designed for ImageNet (256x256x3) [1] could not be directly applied. To adapt the SqueezeNet architecture to classify CIFAR-10 dataset, we re-designed the SqueezeNet to not overfit on CIFAR-10 based on the following principles:

- To reduce the model size, we started by decreasing the model width (number of filters in each convolution layer) before decreasing the model depth. Our intuition was that it would be more important to preserve the representation of abstract features in deep layers in order to maintain the same level of accuracy.
- To maintain activation maps at moderate size, given 32x32x3 input images, we removed early pooling layers and set the hyper-parameter padding to 'same' in convolution layers.

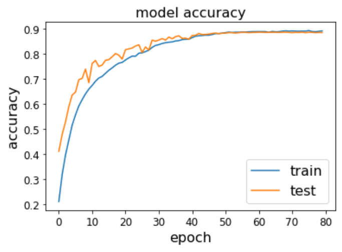


(a) Adapted SqueezeNet (b) Adapted AlexNet

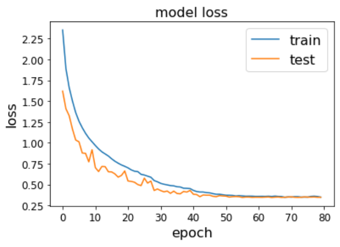
Fig. 2: Architecture of our two baseline models, SqueezeNet and AlexNet adapted to CIFAR-10 dataset.

- We maintained the fire module architecture by using $pct_{3x3} = 0.5$ and $SR = 0.125$, same as the fire module in the original baseline model. As for other metaparameters, the new baseline model had a different organization and number of fire modules to avoid overfitting, thus using the following: $base_e = 128, incre = 128, freq = 1$.
- Additional dropout layers [9] were added if necessary, especially when overfitting occurs at higher squeeze ratios. Existing dropout layers in the original model were not removed.

Our final baseline SqueezeNet was illustrated in Fig.2(a). Furthermore, SqueezeNet with bypass connections was also implemented by adding input and output of each fire module as final output to the next module, using the Add layer in Keras API [11]. Those models with bypass connections are illustrated in [Fig.S1,Fig.S2]. In addition, we were not able to find any official records of AlexNet's performance on CIFAR-10, so we went through the similar processes above to construct a smaller AlexNet, illustrated in Fig.2(b).



(a) Training Time Accuracy Curve



(b) Training Time Loss Curve

Fig. 3: Typical learning curve with our training setup. The two examples above are taken from training of our reduced-size AlexNet

Additional design choices are listed below, the intuition behind them can be found in the paper cited:

- ReLU [4] is applied as activation of all weighted layers;
- note the lack of fully-connected layers in SqueezeNet; this design choice was inspired by the Network in Network architecture [7];
- kernel initialization utilizes Xavier uniform [3] (Default initialization for convolutional layers in Keras [11]; we observed no issue in convergence);
- batch normalization [12] is applied over activation of all weighted layers.

## V. Training and Validation Setup

Our implementation and validation of models were based on the Keras API [11]. Besides the data augmentation performed during training time, we also implemented two callbacks from Keras API [11] to help us control the learning rate and epochs as follows:

- ReduceLROnPlateau: this callback monitors the validation accuracy and if it does not increase for 3 epochs, it will reduce current learning rate by half for the following epochs. Minimum learning rate is set at $1e-6$.
- EarlyStopping: this callback monitors the validation loss and if it does not decrease for 5 epochs, training is stopped and the weights at last layer are saved.

These two callbacks were added so that we did not have to explicitly tune our learning and decaying rate as well as the epoch for each model. The initial learning rate is the default learning rate for the optimizer (most of the time we used Adam optimizer [16] and the default learning rate is 0.001) and the default maximum epoch is set at 100. With categorical cross-entropy, our typical training curves looked like that in Fig.3, where convergence occurred and training stopped before hitting 100 epochs (usually around 80 epochs). Note that there were usually some fluctuations at the beginning of training that we attributed to a high initial learning rate. However we did not adjust the initial learning rate as the learning was able to stabilize and converge after the reduction in learning rate at later epochs. Besides, maintaining the initial learning rate at relatively high value in our case prevented it from becoming too small after the reduction at later epochs.

In terms of our validation setup, the CIFAR-10 dataset is already partitioned into 50000 training and 10000 validation images, so we did not perform k-fold cross validation, neither did we own the computational resources to do in reasonable time, given the amount of ablation studies we performed. Nevertheless, we always ensured that each of our model was not over- or under-fitting before using its validation accuracy in our ablation studies.

## VI. Reproducibility Assessment

### A. *SqueezeNet Baseline*

After a series of careful model reduction, our adapted AlexNet [Fig.2(b)] achieved 88.47% validation accuracy [Table.I]. By referring to a GitHub Repository that collected baseline validation accuracy of current models on CIFAR-10 [17], we saw that the performance of our optimized AlexNet was close to that of the VGG models [8]. Thus we would use this adapted model to compare with the performance of our adapted SqueezeNet in our reproducibility assessment.

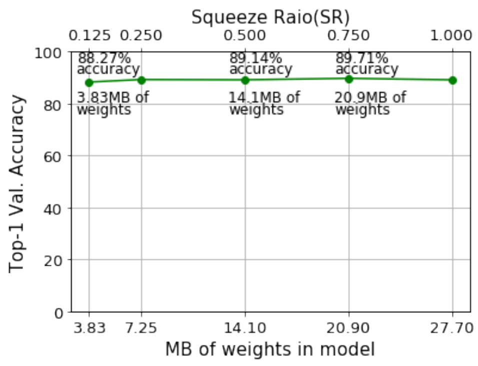
Given our training and validation setup, we attempted to reproduce the success of SqueezeNet in matching performance of AlexNet with 50x reduction in model size on the CIFAR-10 dataset. Our adapted SqueezeNet [Fig.2(a)] reached the same level of performance as AlexNet at approximately 88.27% validation accuracy [Table.I]. However, the reduction in model size was only around 9x, which was reasonable as the model was adapted to a overall smaller dataset. Thus, by using the strategies proposed in Iandola et al. (2016) and implemented in fire module, a SqueezeNet with equivalent performance as AlexNet on CIFAR-10 is reproducible, despite that the model size reduction may depend on the size of the problem (dataset).

### B. *SqueezeNet with Bypass Connections*

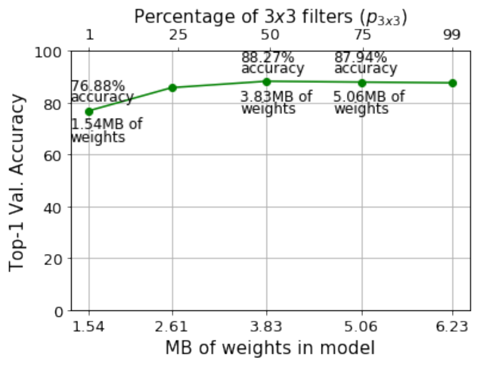
Furthermore, we also tried to reproduce the improvement using bypass connections; in particular, we trained two adapted SqueezeNet models with simple and complex bypass connections, respectively [Fig.S1,Fig.S2]. However, we did not produce any significant improvement, where the simple and complex bypass resulted in 87.81% and 88.46% validation accuracy, respectively [Table.I]. Reflecting on the training processes of those two model, there were slight overfitting issues in both, despite that the one with simple bypass had no increase in model size. We had to add additional dropout layers after max pooling layers to control this overfitting; thus the performance may be affected to a certain extent. The original paper did not mention if they encountered similar issues or present any solutions. Additionally, our adapted SqueezeNet architecture was limited in the number of bypass connections in the first place. For example, fire2 and fire3 module in the original SqueezeNet both contained 128 filters, where a bypass connection could be established. In contrast, ours contained different number of filters because we had to decrease the overall depth [Fig.S1,Fig.S2]. Therefore, the limited number of bypass connections in our adapted SqueezeNet might be the other reason that we could not reproduce any significant improvement.

TABLE I: Validation accuracy of some of the variant SqueezeNet models on CIFAR-10. By default, $base_e = 128, incre = 128, freq = 1, pct_{3x3} = 0.5$ and $SR = 0.125$ in all the following models unless otherwise specified. The adapted AlexNet served as the baseline to compare against our adapted SqueezeNet, it is not a variant of SqueezeNet.

| **Model Modifications** | **Size** | **Top 1 Val. Accuracy** |
|---|---|---|
| Adapted AlexNet | 35.9 MB | 88.47% |
| Adapted SqueezeNet | 3.83 MB | 88.27% |
| Simple Bypass | 3.83 MB | 87.81% |
| Complex Bypass | 4.39 MB | 88.46% |
| Valid Padding | 3.83 MB | 88.64% |
| Early Maxpooling | 3.83 MB | 87.78% |
| Acc. Optimization (SR = 0.5) | 26.55 MB | 91.15% |



(a) Exploring the impact of squeeze ratio (SR) on model size and accuracy; each point represents an independent model trained.



(b) Exploring the impact of the ratio of 3x3 filters in expand layers ($pct_{3x3}$) on model size and accuracy;each point represents an independent model trained.

Fig. 4: Fire module ablation studies in terms of its SR and $pct_{3x3}$.

## VII. Ablation Studies

Iandola et al. (2016) proposed to judiciously decrease the number of parameters in a model while preserving model accuracy by replacing subset of $3\times3$ filters with $1\times1$ ones and also decreasing the number of input channels to those $3 \times 3$ filters. Those strategies were achieved by implementing the fire module with two tunable hyper-parameters, $p_{3x3}$ and squeeze ratio (SR). We started our ablation studies by exploring how those two hyper-parameters were correlated with the model size and accuracy. The original paper also explored these two aspects so we also checked if similar trends was produced. In all our experiments, we used $base_e = 128, incre = 128, freq = 1$ unless otherwise specified.

### A. Squeeze Ratio SR

The squeeze ratio reflects the relative number of filters in the squeeze layer versus the expand layer; in terms of the information flow in such arrangement, SR $= 1$ indicates that all output information from previous module can feed to the expand layer, otherwise the information flow is limited to different extents as SR decreases. We used our adapted SqueezeNet as a starting model with SR $= 0.125$ and trained multiple models that had different squeeze ratios in the range $[0.125, 1.0]$. Our results is presented in Fig.4(a), where each point on the graph represents an independent model trained from scratch. Our results indicated that increasing squeeze ratio beyond $0.125$ did further increase the validation accuracy, which however plateaus at $89.71\%$ with SR = $0.75$. These observations of trends correspond exactly to those seen in the original paper. Given that setting SR $= 1$ only increased model size but not model accuracy, a model should be able to maintain its accuracy even if the information flow between layers is slight limited (e.g. SR $\geq 0.75$). Therefore, the first strategy to introduce squeeze layers is plausible and important in significantly reducing the number of parameters while preserving model accuracy.

### B. Percentage of $3 \times 3$ Filters in Expand Layers $p_{3x3}$

$p_{3x3}$ reflects the number of $3 \times 3$ filters used in an expand layer, whereas the rest are replaced by $1 \times 1$ filters. For example, $p_{3x3} = 100$ is equivalent to an expand layer with only $3\times3$ filters whereas $p_{3x3} = 0$ is equivalent to an expand layer with only $1 \times 1$ filters. Within the CNN architecture intrinsically, $1 \times 1$ filters have substantially smaller spatially resolution compared to $3 \times 3$ ones; thus our investigation also concerns the importance of spatial resolution in CNN filters. As in the previous experiment, we trained multiple models based on the adapted SqueezeNet by varying $p_{3x3}$ from 1 to 99. Our result is presented Fig.4(b), where the adapted SqueezeNet model with $p_{3x3} = 50$ corresponds to that of SR $= 0.125$ in Fig.4(a). Immediately, we observed that using more than $50\%$ of $1 \times 1$ filters in an expand layer had strong negative effects on accuracy, indicating the importance of using filters with spatial resolution more than $1 \times 1$. No further improvement in accuracy was observed beyond using $50\%$ of $3 \times 3$ filters, which was consistent with the trend seen in the original paper. Therefore, despite the importance of spatial resolution for CNN filters, there is a lower-bound (e.g. $p_{3x3} \geq 50$) in the number of such filters with high spatial resolution needed in defining a good accuracy for the model overall. The second strategy to replace subset of $3 \times 3$ filters with $1 \times 1$ ones was indeed another reasonable approach in model parameter reduction.

Note that in Fig.4(b), when $p_{3x3} > 50$, there is slight decrease in accuracy; however it does not imply any correlation between additional $3 \times 3$ filters and negative effects on accuracy. In fact, we were expected to see the accuracy plateau beyond $p_{3x3} = 50$ as in the original paper. Such discrepancies can be attributed to the use of additional dropouts used in the training process due to the significant overfitting issues in the last two models, where $p_{3x3} = 75$ and $p_{3x3} = 99$.

### C. *Downsample Late in the Network*

Moreover, there was a third strategy proposed to delay downsampling in the network, which aims at maximizing accuracy given a budget of parameters. Such delayed downsampling was accomplished by setting padding = "same" in convolution layers and also delaying max pooling operations. To investigate the necessity of such modifications, we reverted both of them, respectively; in particular, we set padding = "valid" in the squeeze layer and inserted an additional max pooling layer before fire1. Neither reverse modifications showed significant differences compared to the adapted SqueezeNet. In particular, valid padding resulted in slightly higher validation accuracy at 88.64% and early pooling led to slightly lower validation accuracy at 87.78%. Neither of those preliminary results could help conclude whether those modifications to delay downsampling were necessary. However, we at least could conclude that such strategy would not bring any significant benefit or harm to our model accuracy. Given that this strategy does not increase our model size either, we may just keep it in our future optimizations.

Note that there are two possible reasons that we did not observe any advantages in delayed downsampling. The first is that we used CIFAR-10 [2] with smaller image samples (32x32x3) compared to the ImageNet [1] with substantially larger ones (256x256x3). A downsampling operation, for example, a max pooling operation with stride 2, will cause considerable loss of pixel information in the large samples than the small ones. In addition, our model also had a limited depth, where each downsampled output was connected to fewer modules (e.g. only two fire modules between two max pooling layers). Our intuition was that the few modules, which were using those downsampled output, were not sufficient to demonstrate the actual effects of delayed downsampling in the network.

## VIII. Model Optimizations

Given the exploration of each network component corresponding to the three strategies in SqueezeNet, we further investigated possible optimizations on our adapted SqueezeNet. A very important thing to note in this section is that "optimization" is a tricky term in the context of SqueezeNet. It is not simply equivalent to the increase in model accuracy, but it also incorporates the consideration of model size. One can infinitely explore deeper and wider models with no squeeze to increase model accuracy, however at the cost of model size. Since our aim was not to produce a model for real life applications, had no domain knowledge limiting our model size; thus it is not practical to consider optimization on model size in our case. Therefore, we explored possible optimizations on model accuracy while maintaining certain model size reduction versus the AlexNet.

All training setup was the same as presented previously, except for the callbacks. We decided not to have early stopping and also switched to a more manual version of learning rate schedule with the following specifications:

- Initial LR $= 1e - 3$;
- If epoch $> 50$, LR $1e - 4$;
- If epoch $> 70$, LR $1e - 5$;
- If epoch $> 85$, LR $1e - 6$;

Such specifications were generated based on the learning results of our series of experiments. With such a LR schedule, the training of our optimized SqueezeNet was demonstrated in Fig.S4. There were still fluctuations before 50 epochs due to high learning rate, but it stabilized afterwards. The underfitting before 50 epochs was due to regularization of our additional dropouts. The training accuracy however caught up and gradually converged after adjustment of learning rate after 50 epochs.

Through a series of experiment on the organization of fire modules, we came down to slightly deeper model with one additional fire module with 128 filters after Conv1 [Fig.S3]. This additional fire module had the largest activation map as its input was never downsampled. Additionally, we were also able to establish an additional simple connection around fire3. Additional 0.2 dropouts were added after conv1 and each max-pooling layer to control overfitting. The metaparameters used were $base_e = 128, incre = 128, pct_{3x3} = 0.5, freq = 2$, and $SR = 0.5$. In this scheme, the optimized model size was 25.55 MB and the optimized model accuracy is 91.15% [Table.I]. Based on the GitHub Repository collecting CIFAR-10 baselines tfbaseline, our optimized model accuracy has surpassed that of the VGG16 (90.02%) VGGNet, while at a model size smaller than that of the AlexNet.

## IX. Discussion and Conclusion

### A. *SqueezeNet Reproducibility*

In our attempt to reproduce SqueezeNet on CIFAR-10 Krizhevsky09learningmultiple, utilizing the three strategies proposed in Iandola et al. (2016), we successfully reproduced the baseline SqueezeNet that achieved the same level of accuracy as AlexNet with 9x model size reduction. Due to limitations on our model depth, we were not able to reproduce the success of further improving SqueezeNet with bypass connections. However, based on the results presented in the original paper, such bypass modifications are still promising for larger networks in boosting model accuracy while maintaining the model size.

Additionally, we observed the same trends in altering the squeeze ratio and $p_{3x3}$ as what was explored in the original paper. The observation of both trends plateauing demonstrated that model accuracy can be maintained at a certain level, despite the decrease in information flow between layers and the decrease in spatial resolution of a subset of convolution filters. These observations set the directions for model size optimizations whenever necessary in real life applications.

Furthermore, during training of models with increasing squeeze ratio, we had to deal with increasing level of overfitting. Note that the squeeze ratio is proportional to the amount of information flowing from one fire module to the next. Thus, our intuition led us to think that besides decreasing the amount of model parameters, these squeeze layers also have certain regularization effects. Such regulation effects are negatively correlated with the squeeze ratio.

### B. *Optimization is Tricky in SqueezeNet*

Optimization is a tricky process in the context of SqueezeNet, as it considers not only model accuracy, but also model size. It will require domain knowledge to define what optimization really is for a specific application. For example, for the models presented in Fig.4(a), a model with SR $= 0.5$ demonstrated approximately $1\%$ of accuracy improvement, but at the cost of around 3x increase in model size. The trade-off between the two will require careful consideration of the need in the application.

In our case, we performed fundamental studies into the SqueezeNet's potential in increasing accuracy, with relieved constraint on model size. We were able to construct a model that rivalled performance of VGG16 VGGNet at a model size smaller than that of the AlexNet. We believe that SqueezeNet indeed is a promising method in terms of model size optimization, and its reproducibility can be extended to build models that match performance with other complex models beyond AlexNet.

## Statement of Contribution

- SITONG: reproduce and optimize model; write the report;
- YUNWEN: reproduce model; make the figures and write the report;
- YAN: reproduce and optimize model; write the report.

## References

[1] J. Deng et al. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *CVPR09*. 2009.

[2] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[3] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256. URL: http://proceedings.mlr.press/v9/glorot10a.html.

[4] Vinod Nair and Geoffrey E. Hinton. “Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair”. In: vol. 27. June 2010, pp. 807–814.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[6] Kaiming He and Jian Sun. “Convolutional Neural Networks at Constrained Time Cost”. In: *CoRR* abs/1412.1710 (2014). arXiv: 1412.1710. URL: http://arxiv.org/abs/1412.1710.

[7] Min Lin, Qiang Chen, and Shuicheng Yan. “Network In Network”. In: *CoRR* abs/1312.4400 (2014).

[8] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *CoRR* abs/1409.1556 (2014). arXiv: 1409.1556. URL: http://arxiv.org/abs/1409.1556.

[9] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[10] Christian Szegedy et al. “Going Deeper with Convolutions”. In: *CoRR* abs/1409.4842 (2014). arXiv: 1409.4842. URL: http://arxiv.org/abs/1409.4842.

[11] François Chollet et al. *Keras*. https://keras.io. 2015.

[12] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *CoRR* abs/1502.03167 (2015). arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167.

[13] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). DOI: 10.1109/cvpr.2016.90.

[14] Forrest N. Iandola et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and ¡1MB model size”. In: *CoRR* abs/1602.07360 (2016). arXiv: 1602.07360. URL: http://arxiv.org/abs/1602.07360.

[15] Forrest Iandola et al. “FireCaffe: Near-Linear Acceleration of Deep Neural Network Training on Compute Clusters”. In: June 2016, pp. 2592–2600. DOI: 10.1109/CVPR.2016.284.

[16] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. “On the Convergence of Adam and Beyond”. In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=ryQu7f-RZ.

[17] *CIFAR-baselines-tf*. https://github.com/wangjksjtu/CIFAR-baselines-tf. 2019.

## Supplemental Material

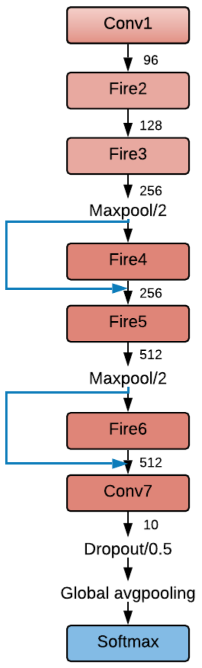


Fig. S1: Adapted SqueezeNet with simple bypass connections.

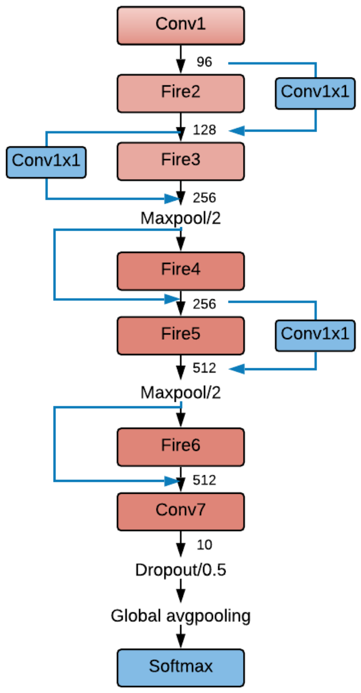


Fig. S2: Adapted SqueezeNet with complex bypass connections.

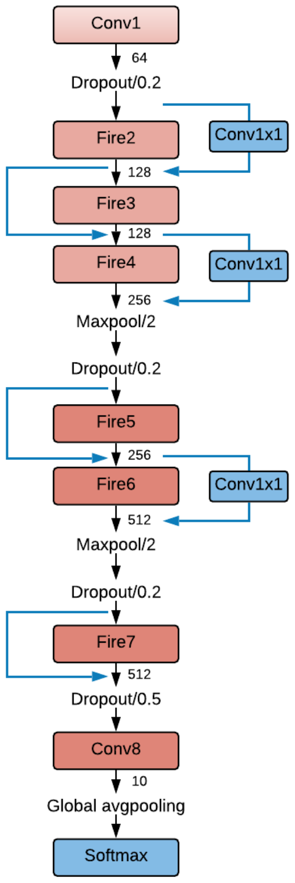


Fig. S3: Optimized SqueezeNet with complex bypass connections.

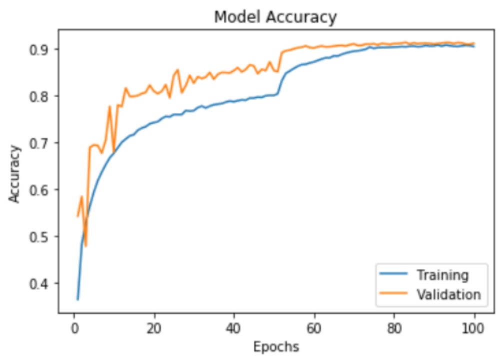


Fig. S4: Training time accuracy curve of optimized SqueezeNet.