# SqueezeNet Reproducibility and Analysis of its Architecture

**Adam Babs**
School of Computer Science
McGill University
Montreal, QC H3A 0G4

**Jizhou Wang**
School of Computer Science
McGill University
Montreal, QC H3A 0G4

**Nahiyan Malik**
School of Computer Science
McGill University
Montreal, QC H3A 0G4

## Abstract

In this project, we attempt to reproduce the results from the original SqueezeNet [9] paper describing parity with AlexNet [14]. We carry out our analysis in three parts: reproducibility, ablation studies and experiments. We use the CIFAR-10 dataset [13]. Although the default SqueezeNet model accuracy did not match the AlexNet results, we were able to modify the SqueezeNet architecture to match the AlexNet accuracy while maintaining a comparable size reduction ratio. Comparing parity, AlexNet achieved 75.42% accuracy compared to the default SqueezeNet model which attained 70.76% with a size reduction of 30x. Following ablations studies and experiments, we were able to achieve a final accuracy of 75.42% using SqueezeNet, with a size reduction of 64x.

## 1 Introduction

Reproducibility is the ability to recreate the results of a particular experiment given all the parameters and details associated with the original research. It is an essential part of scientific research. The issue of lack of reproducibility occurs in scientific research in many fields. It may lead to severe consequences as further going conclusions may be based on theories that turn out to be irreproducible. This problem has also occurred in areas related to machine learning and artificial intelligence [6].

In this project, we attempt to reproduce the results by Iondola et al. from their SqueezeNet paper [9]. The authors present a solution to recreate the accuracy of AlexNet using 50 times fewer parameters which results in a model, named SqueezeNet, which is less than 5 MB.

To reproduce the results, we used the CIFAR-10 image dataset [13]. This is different from the original paper, which uses the ImageNet dataset[3]. However, CIFAR-10 was used for this study due to computational constraints.

We divide our analysis into three main parts: reproducibility, ablation studies and experiments. We first attempt to reproduce the results from the original paper, followed by ablation studies and experiments to improve upon the results. The main goal is to be able to have parity between SqueezeNet and AlexNet. We also explore methods to improve upon the default SqueezeNet implementation via our experiments, especially for the CIFAR-10 dataset.

Our reproducibility tests show that AlexNet with a model size of 88.8 MB has a higher predicting accuracy of 75.42%, while SqueezeNet with a model size of 2.95 MB achieved its best accuracy of 70.76% using default parameters. The size reduction is 30x, less than the original 50x reduction. Another important detail is the training time of particular models. SqueezeNet takes much longer to train than AlexNet due to longer runtimes per epoch as well as more epochs that are required for convergence. Per epoch, on average, AlexNet took 17.45 seconds whereas SqueezeNet took 25.26 seconds.

In order to explore and improve upon the SqueezeNet architecture, we applied ablations and other experiments including residual connections, addition and removal of layers, modifying the squeeze ratio, changing the percentage of 3x3 filters as well as trying different learning rates. Our experiments allowed us to improve the accuracy of the SqueezeNet model to match the accuracy from AlexNet, while reducing size by 64x.

## 2 Related Work

Convolutional Neural Networks (CNN) have over the last decade found great success in image classification tasks. AlexNet is a CNN designed by Alex Krizhevsky et al. It is one of the most influential CNN architectures and it won the ImageNet competition in 2012. This deep CNN has been trained to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. It contains eight learned layers — five convolutional, some of which are followed by max-pooling layers and three fully-connected layers with a final 1000-way softmax [14]. It is the main CNN that is used as the benchmark against SqueezeNet.

SqueezeNet, another CNN, was released in 2016. SqueezeNet was developed by researchers at DeepScale at University of California, Berkeley and Stanford University. In designing SqueezeNet, the goal of the authors was to create a smaller neural network with fewer parameters that can more easily fit into computer memory and can more easily be transmitted over a computer network [5].

The main building block of SqueezeNet is a Fire module. A common layer configuration is as follow:

$\textit{Input} \rightarrow [[Conv \rightarrow \{BN + S\}_{opt} \rightarrow ReLU]^{N} \rightarrow \text{ß}Pool]^{M} \rightarrow (FC||GAP) \rightarrow \textit{Softmax}$

The number of kernels in each convolutional layer is increased after each pooling layer. The general idea behind this is that the network needs more kernels for the complex features at the end of the network than it needs for the basic features in the beginning. Since the number of parameters in a layer is dependent on both the input and output channels as $C_{in}\ x\ C_{out}$, one way to reduce the number of parameters in the network is to alternate between either keeping $C_{in}$ or $C_{out}$ relatively low. This idea is realized by the Fire module featured in the SqueezeNet architecture, which also exploits the kernel size to reduce the complexity further. A Fire module is made up of two layers: squeeze and expand. Within a squeeze layer, there are only 1x1 filters and in the expand layer, there are combined 1x1 and 3x3 filters [9].

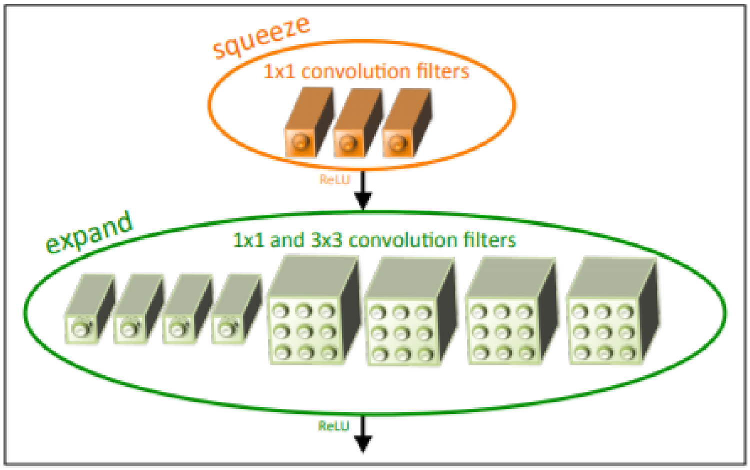

Figure 1: Organization of filters in the Fire module. There is a squeeze and expand layer; the squeeze layer contains 1x1 filters and the expand layer combines 1x1 and 3x3 filters.

Ablation studies are a way to test the importance of different parts of a network. Richard Meyes et al. discuss how ablation influences deep learning and present relevant work regarding the use of ablation and its possible applications and outcomes. They state: "We found that, in general, the larger the ablated network portion, the stronger the effect on the classification performance. However, this effect greatly varies across different aspects of the network." [17]

There have been ablation experiments with SqueezeNet comparing CIFAR-10 [22] conducted on TensorFlow [1] with accuracy reaching 75% on the test set. Although SqueezeNet is an architecture that tries save space by reducing the number of trainable parameters while pertaining accuracy close to AlexNet, there have been more recent architectures such as EfficientNet [21] that reduce the number of parameters while improving accuracy on existing CNNs. EfficientNet is the current State-of-the-Art neural network model for image classification on ImageNet [3] and can be scaled with up to 21x fewer model parameters than existing CNNs. It tries to address the problem of scaling within a neural network model by proposing a *compound scaling method*. In conventional practice, models are scaled arbitrarily. EfficientNet uniformly scales all 3 parameters with a fixed scaling coefficient which can be found through reinforcement learning [20].

Hyperparameter optimization can also lead to changes in performance. Domhan et al. discuss the significance of hyperparameter optimization and describe how it affects performance of neural networks. Deep neural networks (DNNs) show very strong performance on many machine learning problems, but they are very sensitive to the setting of their hyperparameters [4].

## 3 Dataset and Setup

The original paper uses ImageNet to benchmark SqueezeNet against other models such as AlexNet. Due to computing constraints which makes it difficult to train a model on the full ImageNet dataset, we opted to use the CIFAR-10 dataset instead. The reproducibility tests were conducted using Pytorch [18] models instead of Caffe [11] on which the results were originally produced by the paper. Implementations of SqueezeNet and AlexNet convey the same high level description of the internal structure. We hope to achieve comparable results regardless of platform usage.

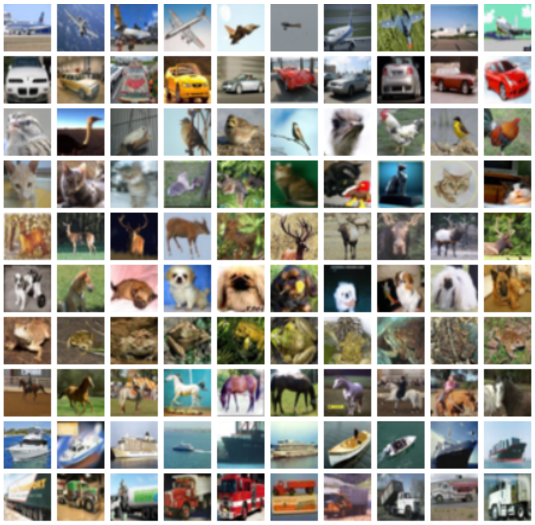

Figure 2: Example of images from the CIFAR-10 dataset.

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. We split the dataset into 80% for training, 10% for validation and 10% for testing. All the images were used in their original format for training and prediction. A batch size of 50 was used for the network input.

## 4 Proposed Approach

There were three main steps in our approach. First, we attempted to reproduce the results from the original paper, comparing AlexNet and SqueezeNet. The only difference between the original paper and our method was the usage of the CIFAR-10 dataset instead of ImageNet due to computing constraints. Second, we ran ablation studies to introduce and remove details from the original SqueezeNet model to test for the efficacy of each factor. Third, We ran our own experiments to attempt to improve the overall accuracy compared to AlexNet.

For all three parts, the CIFAR-10 dataset was split into a training set of 80% (50000 images), a validation set of 10% (5000 images) and a final testing set of 10% (5000 images).

In term of rigorous testing, each of our members ran the same experiments across three different initial seeds. We have averaged our results across all seeds in the results section below.

### 4.1 Reproducibility

The original SqueezeNet paper claims the same accuracy between SqueezeNet and AlexNet on the ImageNet dataset - we ran our experiment on CIFAR-10 instead. The same training and validation data was used on both models to compare performance and accuracy.

The default hyperparameters for our experiments were set to have the same values as the original experiments given by the source code. However, the mentioning of optimizer and learning rate wasn't clear in their setup. As with the time constraint on our experiments, we've opted for a more efficient adaptive optimizer called Adabound [15] instead of trying to fine tune generic optimizers such as Adam [12] and SGD [2]. Adabound is an adaptive method at the beginning of training similar to Adam. It then gradually transforms to SGD which gives a theoretical proof of convergence at a faster speed.

### 4.2 Ablation Studies

The SqueezeNet architecture is made up of Fire modules. Each Fire module is a combination of squeeze and expand layers. As part of the SqueezeNet architecture, 8 Fire modules are used, with a normal convolutional layer preceding and one succeeding the Fire modules. In order to test the importance of the Fire modules and the effects they have on the results, we conducted two main ablation studies: the addition and removal of Fire modules.

For the addition of the Fire modules, duplicate modules were created after the first and second maxpool layers. As a result, after the additions, there were 10 Fire modules. Similarly, for the removal of the Fire modules, two modules were removed after the first and second maxpool layers. The reduction resulted in 6 total Fire modules.

### 4.3 Experiments

Following the ablation studies, the goal was to experiment with different parameters and network architectures to achieve the best accuracy possible from SqueezeNet on the CIFAR-10 dataset. In general, we will be experimenting with batch normalization [10], residual connections, squeeze ratio and percentage of 3x3 filters given by the Fire module, starting convolution kernel size and learning rates.

#### 4.3.1 Batch Normalization

Because of training, a certain layer in a neural network has many parameters that depend on the changes from the previous layer. This requires careful initialization for the model to learn nonlinearities. Batch normalization is applied to reduce the covariate shift of each layer and has proven to speed up the training process by a factor of 14 as well as acting as a regularizer for the neural network. Given the time constraints of our project, we have decided to apply batch normalization on most of our experiments. These batch normalizations are applied within Fire modules as well as in the layers outside of the Fire modules after every convolutional layer.

#### 4.3.2 Residual Connections

As mentioned in the SqueezeNet paper, residual skip connections were added, motivated by the ResNet model architecture [7]. These addition operations were added as shown in Figure 3. As the default replication comparison, we will try to show an increase in accuracy. Experiments will be set up such that multiple residual layers can be added by replicating Fire layers with same number of input and output connections as well as extending the final Fire layer beyond 512 connections to further improve accuracy.

#### 4.3.3 Squeeze Ratio

The Fire modules are composed of squeeze and expand layers. A squeeze layer is composed of 1x1 filters whereas an expand layer has a combination of 1x1 and 3x3 filters, by default 50% each. The squeeze ratio is defined as "the ratio between the number of filters in squeeze layers and the number of filters in expand layers." [9] We experiment with different levels of squeeze ratio. As the ratio is increased, so is the size of the overall model. We hope to reproduce the same improvements as seen in the original paper with higher squeeze ratios.

#### 4.3.4 Percentage of 3x3 filters

Another hyperparameter for tuning within the Fire modules is the percentage of 3x3 expand filters. The default percentage of 3x3 expand filters compared 1x1 expand filters is set at a 1:1 ratio. We plan to experiment with a 12.5%, 25% and 75% 3x3 expand filters. As the percentage is increased, so is the size of the overall model. We hope to reproduce the same improvement as seen in the original paper with higher percentage of 3x3 expand filters.

#### 4.3.5 Starting Convolution Kernel Size

The SqueezeNet architecture first starts with a traditional convolution layer, which has its own starting kernel size that is independent of the 1x1 and 3x3 filters used as part of the Fire modules. The default value as part of SqueezeNet is a

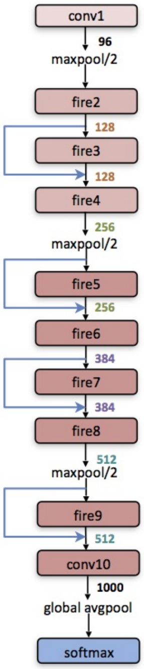

Figure 3: Visualization of the residual additions.

7x7 filter. Different sized filters were tried: 1x1, 3x3 and 5x5. It is important to note that SqueezeNet was originally trained on ImageNet, which has images of higher resolution than those from CIFAR-10. Therefore, it is possible for different sized filters to have varying levels of success depending on the dataset.

### 4.3.6 Learning Rates

Another modification we made was manipulating the learning rate. Five different rates were used for the Adabound optimizer. We found that the learning rate parameter proposed in the original paper, 0.04, yielded a high accuracy, however we found out that a learning rate of 0.1 yielded a notably higher accuracy. This could be due to the Adabound optimizer that is used.

# 5 Results

## 5.1 Reproducibility

Overall, the reproducibility tests show that AlexNet with a model size of 88.8 MB has a higher predicting accuracy of 75.42%, while SqueezeNet with a model size of 2.95 MB achieved its best accuracy of 70.76% using default parameters. The size reduction is 30x, less than the original 50x reduction. Training time of AlexNet took 17.45 seconds per epoch, whereas SqueezeNet took 25.26 seconds per epoch. SqueezeNet also needed 84 epochs to converge, compared to 18 epochs for AlexNet. When adding batchnorm, both models yielded better results.

| Model | Accuracy | Epochs | Time per Epoch (seconds) | Size (MB) |
|---|---|---|---|---|
| AlexNet | 75.42% | 18 | 17.45 | 88.78 |
| AlexNet, batchnorm | 79.22% | 15 | 20.12 | 88.78 |
| SqueezeNet | 70.76% | 30 | 29.95 | 2.95 |
| SqueezeNet, batchnorm | 73.72% | 52 | 37.19 | 2.95 |

Table 1. Results from reproducibility tests.

## 5.2 Ablation Studies

For the ablation studies, the addition and removal of single Fire modules had insignificant impact on the original accuracy. All tests resulted in approximately the same accuracy of 70%. When adding layers, the model sizes increased as well as training time. The removal of layers, in contrast, decreased both the model size and training times. If the decrease in accuracy is within acceptable measures, removing layers can be a way to reducing size. The removal of layers also showed less overfitting over more epochs. This is expected as the smaller network is inherently less complex. It is also important to note that without batchnorm, when adding layers, there was no convergence. The accuracy was at 10%, indicative of always predicting a single label.

| Model | Accuracy | Epochs | Time per Epoch (seconds) | Size (MB) |
|---|---|---|---|---|
| Fire layers added | 10% | 10 | 25.15 | 3.4 |
| Fire layers added, batchnorm | 70.12% | 54 | 26.47 | 3.48 |
| Fire layers removed | 70.34% | 80 | 18.85 | 1.95 |
| Fire layers removed, batchnorm | 70.44% | 96 | 19.29 | 2.0 |

Table 2. Results from ablation studies.

### 5.3 Experiments

Finally, experiments allowed us to improve the accuracy of the SqueezeNet model to match the accuracy from AlexNet, with the best set of hyperparameters containing residual connections with batchnorm and two layers removed as well as a starting convolution size of 5x5. The details of the experiments are below.

#### 5.3.1 Batch Normalization

Batch normalization improved the total training runtime and accuracy of most of our experiments and helped with convergence. The default SqueezeNet with batch normalization in every layer attained a 73.72% accuracy compared to the original 70.76%. On AlexNet, the accuracy improved to 79.22% from 75.42%. This can be observed in Table 1.

Applying batch normalization outside of the Fire module made no difference in terms of accuracy while applying batch normalization only inside increased the accuracy to 73.4%. Due to the improved results from batch normalization, all experiments henceforth include batch normalization.

#### 5.3.2 Residual Connections

Adding residual connections improved the accuracy substantially to a 74.84% as default. Testing of adding and removing layers was also conducted with residual connections. Some layers were added and removed, starting from the Fire modules before the output layer. These added layers had increments of 128 in terms of input size, with two added layers increasing the size of the input to 768. As for the addition of Fire module added by replicating Fire layers with same number of input and output connections, 2 and 4 replications were tested.

The best accuracy with the smallest model size was achieved with the removal of the last 2 Fire layers. Its accuracy is 75.42% with a model size of 1.4MB. Detailed results are shown in Table 3 below.

| Model | Accuracy | Epochs | Time per Epoch (seconds) | Size (MB) |
|---|---|---|---|---|
| Residual | 74.84% | 59 | 37.71 | 2.95 |
| Residual (2 layers removed) | 75.42% | 30 | 29.95 | 1.4 |
| Residual (4 layers removed) | 73.60% | 33 | 23.21 | 0.563 |
| Residual (2 layers added) | 75.44% | 34 | 48.55 | 8.62 |
| Residual (2 layers replicated) | 75.36% | 22 | 40.30 | 3.15 |
| Residual (4 layers replicated) | 74.72% | 34 | 45.47 | 4.35 |

Table 3. Results from residual connections experiment.

#### 5.3.3 Squeeze Ratio

The squeeze ratio (SR) was modified to observe how accuracy increases or decreases based on the overall size of the model. This is because a higher squeeze ratio signifies more filters from the squeeze layer of the Fire module. We found that increasing the squeeze ratio does indeed increase accuracy, however, it also drastically increases the size of the model, which goes against the purpose of SqueezeNet, that being of lightweight utility.

| Model | Accuracy | Epochs | Time per Epoch (seconds) | Size (MB) |
|---|---|---|---|---|
| SR 0.125 (default) | 73.72% | 52 | 37.19 | 2.95 |
| SR 0.25 | 76.48% | 59 | 54.21 | 5.65 |
| SR 0.5 | 77.6% | 94 | 53.29 | 11.14 |
| SR 0.75 | 77.92% | 81 | 35.18 | 16.63 |

Table 4. Results from squeeze ratio experiment.

#### 5.3.4 Percentage of 3x3 filters

The results of tuning the percentages of 3x3 filter are shown in Table 5. To our surprise, increasing the percentage of 3x3 filter does not show consistent results with the SqueezeNet paper's findings. The accuracy decreases after 25%.

| Model | Accuracy | Epochs | Time per Epoch (seconds) | Size (MB) |
|---|---|---|---|---|
| 12.5% 3x3 | 72.62% | 41 | 34.49 | 1.49 |
| 25% 3x3 | 74.08% | 32 | 34.29 | 1.96 |
| 50% 3x3 (default) | 73.72% | 52 | 37.19 | 2.95 |
| 75% 3x3 | 73.02% | 41 | 34.21 | 3.84 |

Table 5. Results from percentage of 3x3 filters experiment.

#### 5.3.5 Starting Convolution Kernel Size

The default starting kernel size for SqueezeNet is 7x7. We experimented with different values for the starting kernel size as it would have the most downstream impact on the rest of the layers of the network. The best results were from a 5x5 starting filter with an accuracy of 75.74%. It must be noted that the CIFAR-10 dataset contains 32x32 resolutions images - much smaller than ImageNet. This could be a reason why 5x5 performs better than the default 7x7 starting kernel size.

| Model | Accuracy | Epochs | Time per Epoch (seconds) | Size (MB) |
|---|---|---|---|---|
| 1x1 starting kernel size | 64.78% | 12 | 33.80 | 2.85 |
| 3x3 starting kernel size | 75.32% | 66 | 34.29 | 2.86 |
| 5x5 starting kernel size | 76.74% | 80 | 34.57 | 2.87 |

Table 6. Results from starting kernel size experiment.

#### 5.3.6 Learning Rates

The default learning rate of SqueezeNet is 0.04. We experimented with different learning rates using the Adabound optimizer. As presented below, the learning rate has a considerable impact on model performance. Results have been presented below in Table 7.

| Learning Rate | Accuracy | Epochs | Time per Epoch (seconds) | Size (MB) |
|---|---|---|---|---|
| 0.01 | 59.02% | 30 | 26.10 | 2.95 |
| 0.05 | 43.14% | 29 | 25.07 | 2.95 |
| 0.1 | 68.98% | 43 | 24.28 | 2.95 |
| 0.2 | 52.40% | 26 | 24.82 | 2.95 |
| 0.5 | 14.76% | 3 | 24.76 | 2.95 |

Table 7. Results from changing the learning rates with the Adabound optimizer.

Results above have been presented only for the Adabound optimizer. Adam and SGD optimizers have also been tested, however, probably due to the lack of batch normalization, models were not able to train and the accuracy did not exceed 14.76%.

#### 5.3.7 Best Results

Based on the experiments stated above, the best results were attained by two separate model variations: the residual network with 2 layers removed as shown in Table 3 and a starting kernel size of 5x5 as shown in Table 6. For our final model configuration, we combine these two experiments together and run it on our held out test set. The validation accuracy during training was 78.12% and the final accuracy on the held out test set was 76.34%. This configuration converges in 89 epochs with 27.94 seconds per epoch. It has a model size of 1.37 MB.

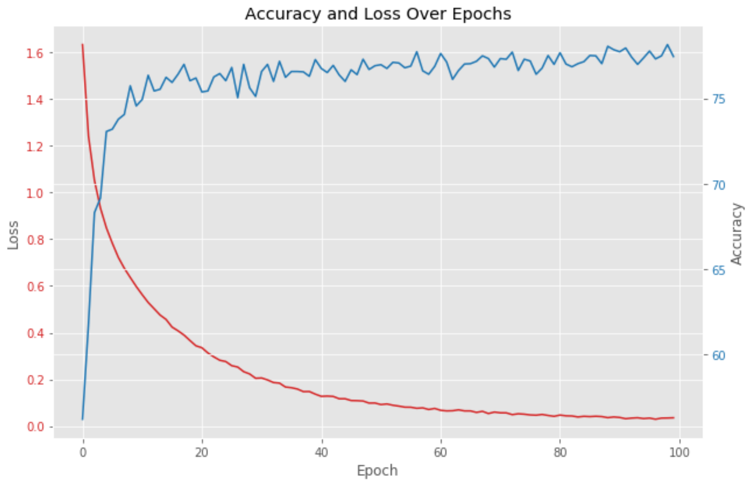

Figure 4: The loss and accuracy over epochs for the final model configuration.

## 6 Discussion and Conclusion

In conclusion, our findings did not completely replicate the results given by the original SqueezeNet paper [9]. we were, however, able to improve the SqueezeNet model with the addition of experiments such as residuals and filter sizes and have able to attain parity with the base AlexNet model. The original SqueezeNet paper did not make much mention of the much higher training times required for SqueezeNet compared to AlexNet, but we highlight the runtimes for all our analyses.

A possible hypothesis regarding our first results in terms of disparity can come from the fact that ImageNet image input resolution is 224x224 in the original experiment compared to the 32x32 resolution from CIFAR-10. The results shown from the decrease of our SqueezeNet layers and kernal size support evidence that a smaller network is better at classifying smaller images, although further testing and investigation is needed for verification of this hypothesis.

Other differences regarding the code implementation and platforms, random initialization and other unknown hyperparameters that weren't completely disclosed by the paper can affect the outcome of our reproducibility experiment.

For future investigation, given enough time and computing resources, we would like to solidify our investigation by conducting more rigorous reproducibility methodologies such as 5-fold cross validation [19] and multiple random seed settings to achieve significant results within a confidence interval of low error. We could also try to replicate the same experiments shown in this paper on ImageNet and other image datasets and see how ablation of other components such as additional squeeze and expand layers with different filter sizes would affect the results.

The general domain of reproducibility in machine learning research is one that must improve as machine learning is adopted more in the mainstream. Deep learning models, with their large architectures inherently introduce a lot of complexities. The discourse of the black box of machine learning, trust and interpretability will garner more focus, especially in fields such as medicine as radiology uses machine learning solutions more and more [16], which makes reproducibility in vision based models such as AlexNet and SqueezeNet increasingly vital. Techniques such as reinforcement learning also introduce many issues. Factors due to the non-deterministic nature of reinforcement learning make reproducibility complicated and difficult to interpret. Henderson et al. describe what is needed in such cases, including "proper experimental techniques, and reporting procedures." [8]

## 7 Statement of Contributions

Jizhou worked the default implementations of AlexNet and SqueezeNet as well as experiments to improve accuracy. Nahiyan worked the default implementations of AlexNet and SqueezeNet as well as experiments to improve accuracy. Adam worked on reproducibility of AlexNet and SqueezeNet as well as experiments to improve accuracy. Everyone took part in the writing of the paper.

## References

[1] Martın Abadi et al. “Tensorflow: A system for large-scale machine learning”. In: *12th $\{$USENIX$\}$ Symposium on Operating Systems Design and Implementation ($\{$OSDI$\}$ 16)*. 2016, pp. 265–283.

[2] Léon Bottou. “Large-scale machine learning with stochastic gradient descent”. In: *Proceedings of COMPSTAT’2010*. Springer, 2010, pp. 177–186.

[3] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[4] Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. “Speeding Up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves.” In: *IJCAI*. Ed. by Qiang Yang and Michael J. Wooldridge. AAAI Press, 2015, pp. 3460–3468. ISBN: 978-1-57735-738-4. URL: `http://dblp.uni-trier.de/db/conf/ijcai/ijcai2015.html#DomhanSH15`.

[5] Abhinav Ganesh. “Deep Learning Reading Group: SqueezeNet”. In: (2018-04-07).

[6] Odd Erik Gundersen and Sigbjørn Kjensmo. “State of the Art: Reproducibility in Artificial Intelligence”. In: Feb. 2018.

[7] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[8] Peter Henderson et al. “Deep reinforcement learning that matters”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[9] 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`.

[10] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).

[11] Yangqing Jia et al. “Caffe: Convolutional Architecture for Fast Feature Embedding”. In: *arXiv preprint arXiv:1408.5093* (2014).

[12] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[13] Alex Krizhevsky, Geoffrey Hinton, et al. *Learning multiple layers of features from tiny images*. Tech. rep. Citeseer, 2009.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. 2012, pp. 1106–1114. URL: `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks`.

[15] Liangchen Luo, Yuanhao Xiong, and Yan Liu. “Adaptive Gradient Methods with Dynamic Bound of Learning Rate”. In: *International Conference on Learning Representations*. 2019. URL: `https://openreview.net/forum?id=Bkg3g2R9FX`.

[16] The Lancet Respiratory Medicine. “Opening the black box of machine learning”. In: *The Lancet Respiratory Medicine* 6.11 (Nov. 2018), p. 801. DOI: `10.1016/s2213-2600(18)30425-9`. URL: `https://doi.org/10.1016/s2213-2600(18)30425-9`.

[17] Richard Meyes et al. “Ablation Studies in Artificial Neural Networks”. In: *arXiv preprint arXiv:1901.08644* (2019).

[18] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[19] Mervyn Stone. “Cross-validatory choice and assessment of statistical predictions”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 36.2 (1974), pp. 111–133.

[20] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*. Vol. 2. 4. MIT press Cambridge, 1998.

[21] Mingxing Tan and Quoc V Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *arXiv preprint arXiv:1905.11946* (2019).

[22] zshancock. $\mathit{SqueezeNet}_v s_C IFAR10$. `https://github.com/zshancock/SqueezeNet_vs_CIFAR10`. 2019.