Regularization Techniques for Image Classification Tasks

Aashka Trivedi (aht323), Natalia Zubkova (nz2094)

1 Project Description

This project aims to empirically understand the effects of introducing the following four regularization techniques- Batch Normalization [1, 2], Dropout [3], Data Augmentation [4] and Adaptive Gradient Clipping [5]- to a Resnet-50 Model. We evaluate the performance of the model on an Image Classification task using the CIFAR-10 Dataset.

The major goal of this project is to determine a way to use the above mentioned techniques to reduce overfitting and improve the accuracy of a self-implemented Resnet. A second goal of our project is to study whether Adaptive Gradient Clipping [5] can be used as an effective replacement of Batch Normalization.

2 Approach

There are two phases of this project- the Research phase, and the Implementation Phase. The approach to both these phases are discussed below.

Research Phase

This phase involves exploring different combinations of regularization parameters to obtain the best configuration to reduce overfitting. Here we study the effects of the following regularization techniques and their respective parameters on a Resnet-50 model implemented from scratch:

1. Batch Normalization: We mainly observe how the different metrics are affected when we introduce a batch normalization layer to different Resnet blocks.

2. Dropout: We introduce dropout layers after different Resnet blocks, and test the effect that varying dropout probabilities have on the metrics. Specifically, we define two types of models - those with *Symmetric Dropout*, which have the same dropout probability across all layers, and those containing *Asymmetric Dropout* and have different dropout probabilities across the different layers.

3. Adaptive Gradient Clipping: We seek to test Brock et. al.'s [5] use of adaptive gradient clipping in Normalization-Free Resnets. For the same, we conduct a thorough ablation study on the effects of the clip value threshold and batch size on performance. After obtaining the best parameter values, we compare the performance of a Normalization Free model with adaptive gradient clipping to other models with varying batch-norm, dropout, and other regularization methods.

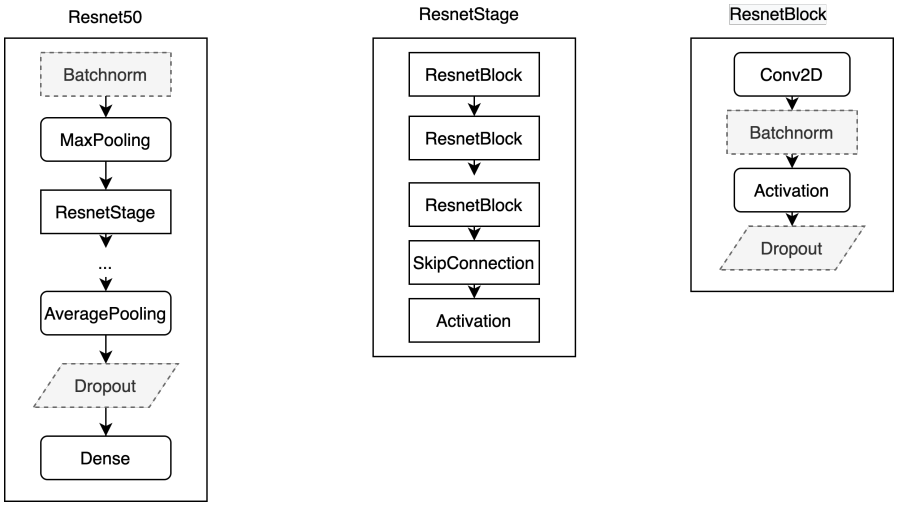


Figure 1: Resnet architecture sketch. Gray boxes indicate added/removed regularization layers. ... indicates omission of repeating Resnet stages.

The scheme of our Resnet implementation with optionally added Batch Normalization and Dropout layers is depicted in Figure 1. The performance of different models is measured by their accuracy on the test set, training time for 200 epochs and time to reach 87% training accuracy. We choose the best model configurations from each training pool to analyze in the implementation phase.

Implementation Phase

Here, we implement a Resnet-50 model from scratch, train it till convergence using the best configurations from the research phase, and evaluate its performance by analyzing the training accuracy, testing accuracy and time to train. In this phase, we seek to empirically study the following key points:

1. Adaptive Gradient Clipping as an alternative to Batch Normalization.

2. The effect of varying the dropout probability across layers.

3. Data Augmentation Techniques (specifically, image transformations and cutout regularization) to effectively increase the dataset size and provide regularization.

4. Comparison of overfitting in a baseline model and a model with regularization techniques applied.

The best models are evaluated on the CIFAR-10 Dataset [6].

3 Implementation Details

A brief overview of the implementation details are as follows.

Resnet 50 Implementation: Our Resnet 50 implementation consists of Identity blocks and Convolution blocks which forms the basis of out Resnet Blocks. To maintain the modular approach of the original Resnet Models, we have maintained that once we set the parameters (e.g., number of dropout and batchnorm layers, dropout probability, etc) for the Identity and Convolution blocks, these parameters will not change with different instantiations in the Resnet blocks.

Regularization Techniques: The different parameters tested to find the best candidates for Batch Normalization, Dropout and Adaptive Gradient Clipping is given in Table 1. The number of layers are an indication of how many ResnetBlocks in Figure 1 have a Batchnorm and/or Dropout Layer.

Regularization Technique	Parameter	Candidates
Batch Normalization	Number of Layers BN is Applied To	0,1,2,3
Dropout	Number of Layers Dropout is Applied To Symmetric Dropout Probability Asymmetric Dropout Probability	0,1,2,3 0.2,0.5,0.8 [0.1,0.8] (continuous range)
Adaptive Gradient Clipping	Clip Threshold Batch Size	0.01,0.02,0.04 64, 128, 256
Cutout Regularization	Cutout Images per Batch	2,8,6

Table 1: Parameters tested to Analyze different Regularization Techniques

Framework and Hardware: The entirety of this project has been coded using Tensorflow. Every model has been trained on a Nvidia Tesla V100 GPU, on the Deep Learning VM solution of Google Cloud.

4 Experimental Results

The experiments performed to understand the effects of each regularization technique, as discussed in Table 1, followed by a brief discussion on the observations are given in this section.

For every configuration, we analyzed the test accuracy and overall training time, as well as time to reach 87% training accuracy.

Dropout

Our first experiment concerned how many dropout layers per ResNet block to use in the model. For each value 0.2, 0.5 and 0.8 we ran experiments increasing the number of dropout layers with that value. Figure 2 illustrates the trade-off between the *level of dropout* and *test accuracy*. From here we see that a dropout probability of 0.2 outperforms almost every case of varying Batchnorm and Dropout Layer. Figure 3, in turn, indicates that the best performance for 0.2 is in the case of 2 layers, and for larger probabilities the fewer layers - the better.

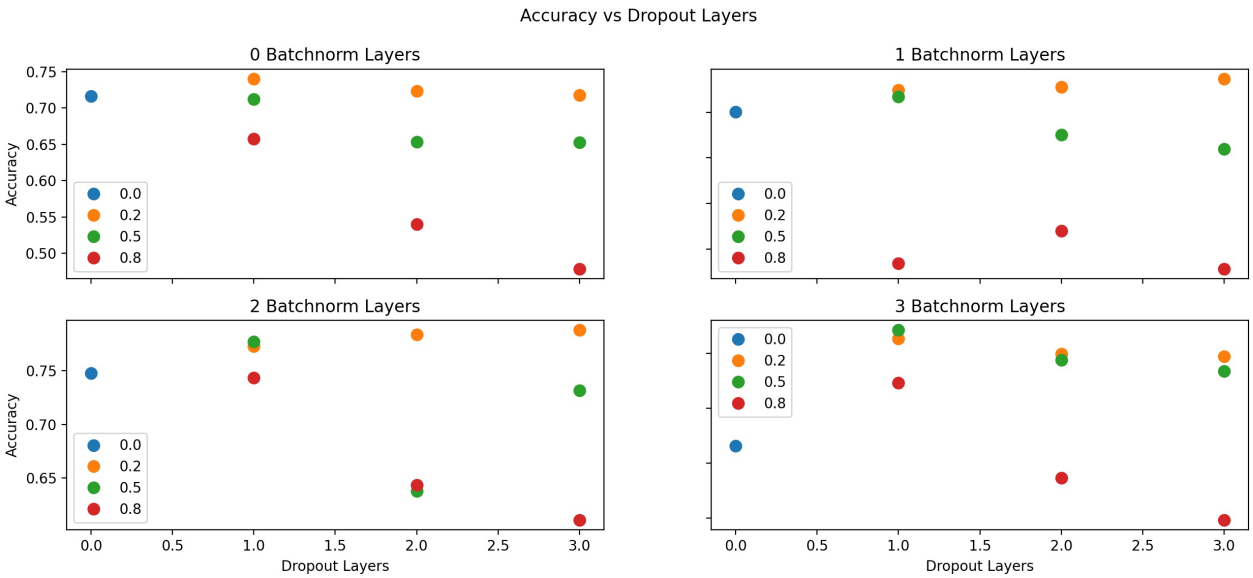


Figure 2: Accuracy vs Dropout Probabilities

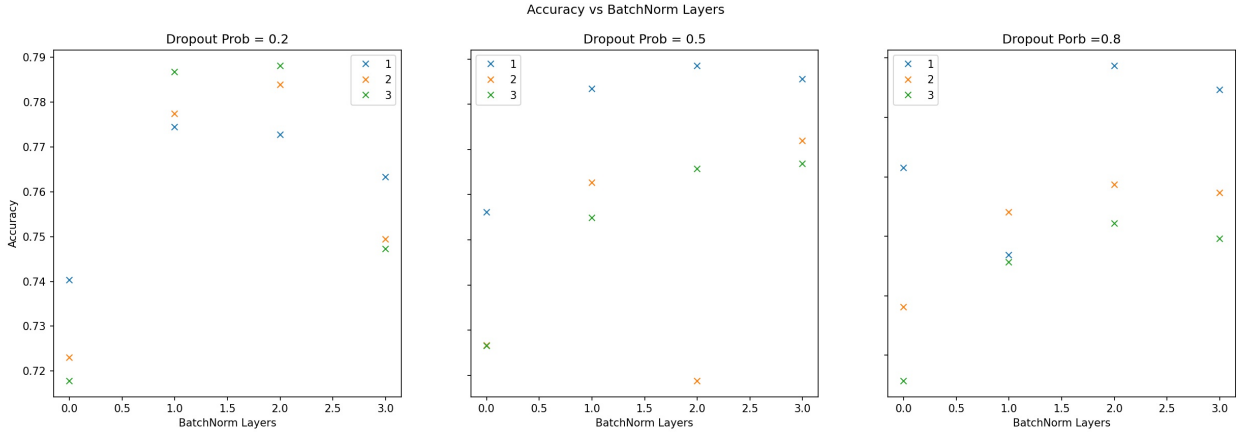
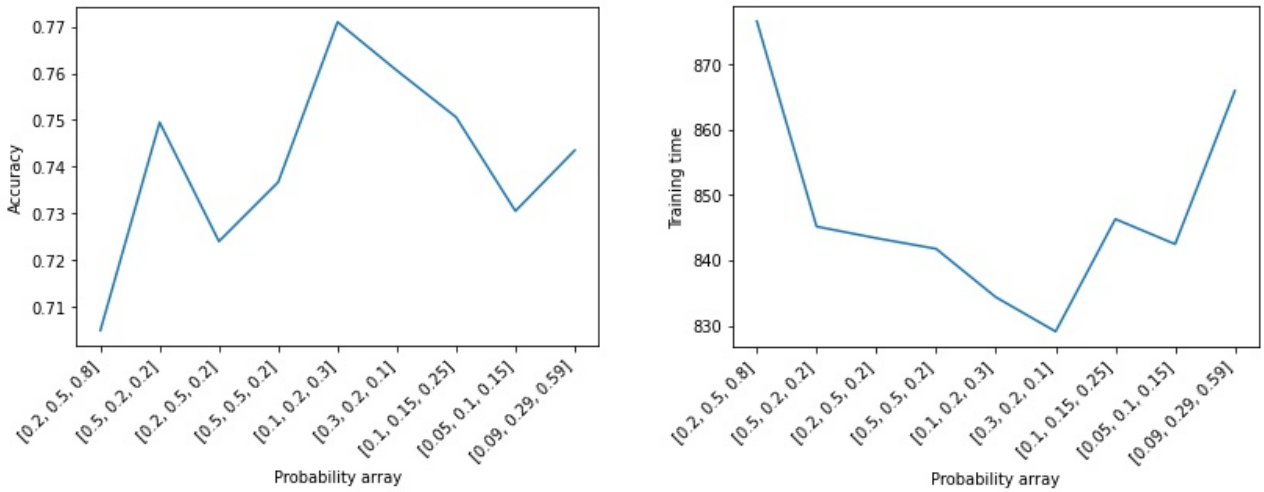


Figure 3: Accuracy vs Batch normalization layers with different Dropout Probabilities

Next, we wanted to examine the appropriateness of *different combinations* of dropout probabilities. Figure 4 demonstrates examples of different probability arrays used in our experiments. These combinations were selected using manual grid search, and adjusted based on produced metrics. The element *probability_array[i]* refers to the dropout probability give to the *i-th* Resnet Block, as shown in Figure 1.



(a) Accuracy vs Dropout Probabilities

(b) Training time vs Dropout Probabilities

Figure 4: Asymmetric Dropout Performance

Batch Normalization

We ran similar set of experiments for Batch Normalization. Here we only experiment with number of layers and how this number combines with previously discussed dropout configurations. From Figure 5 we can see that the best performance is produced by BN=2. Data depicted in Figure 6 confirms the ability of batch normalization to accelerate training: with the increase of batch normalization layers, it takes less time to achieve training accuracy of 87%.

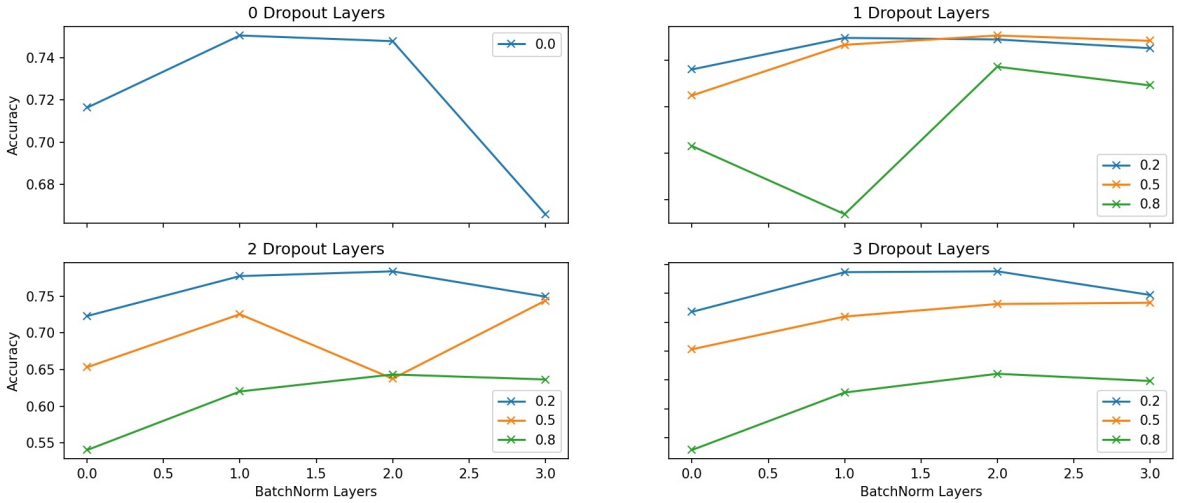


Figure 5: Accuracy vs Batch Normalization layers

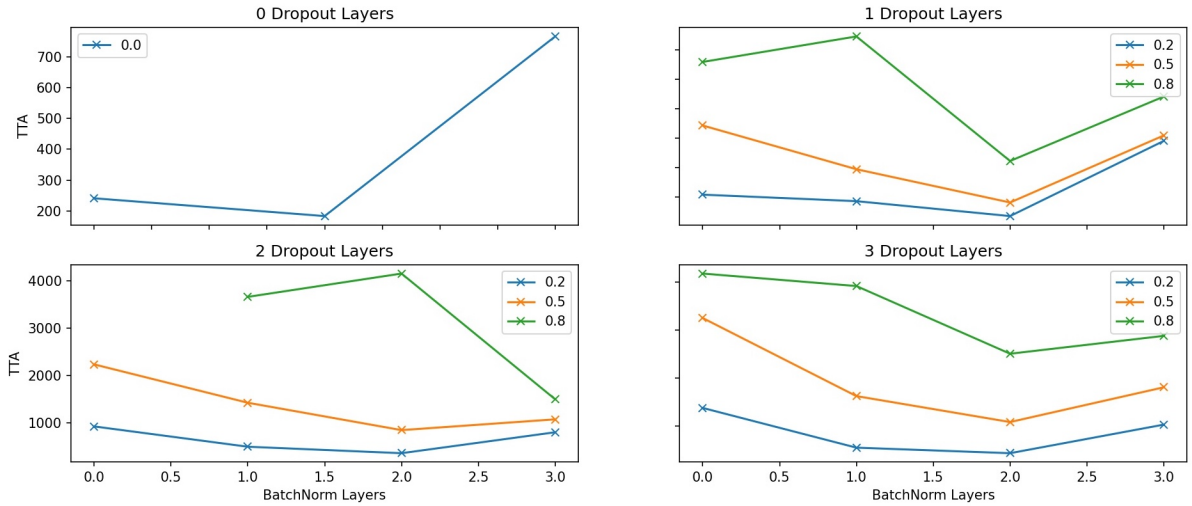


Figure 6: TTA vs Batch Normalization layers

Adaptive Gradient Clipping

For this technique, we first wanted to study the impact of its hyperparameters in application to our case. The authors of [5] mention two: *batch size* and *clip value*. As we increased batch size, models took less time to train. This is not attributed to this method specifically, but that is why we chose the largest tested batch size **256**. Figure 7 shows that for almost all cases, clip value of **0.02** produced better accuracy scores than all other values. This was in accordance to the recommendations of [5], where it is obeserved that a large batch size and a lower clip value helps stabilize training and improves performance. We observe that time was not affected by varying clipvalue.

Since Adaptive Gradient Clipping (AGC) is a technique that is aimed to replace batch normalization, we wanted to compare the performance of our models with these 2 approaches. Figure 8 shows accuracy, training time and time to accuracy of 5 models: with 0 to 3 batch normalization layers in Resnet block, and with AGC. We can see that in our case AGC took more time to train and slightly more time to achieve 87% training accuracy than models with batch normalization (0 to 2 layers). As for accuracy, AGC's performance is comparable to others in the case of *0 dropout layers*. This brings the intuition that AGC can be an effective way to replace normalization if dropout is not present. What is more, in the paper the authors suggested using other, more rigorous means of regularization, e.g. L2 regularization. To further improve performance of models with AGC one could experiment with optimizers and learning rate decay.

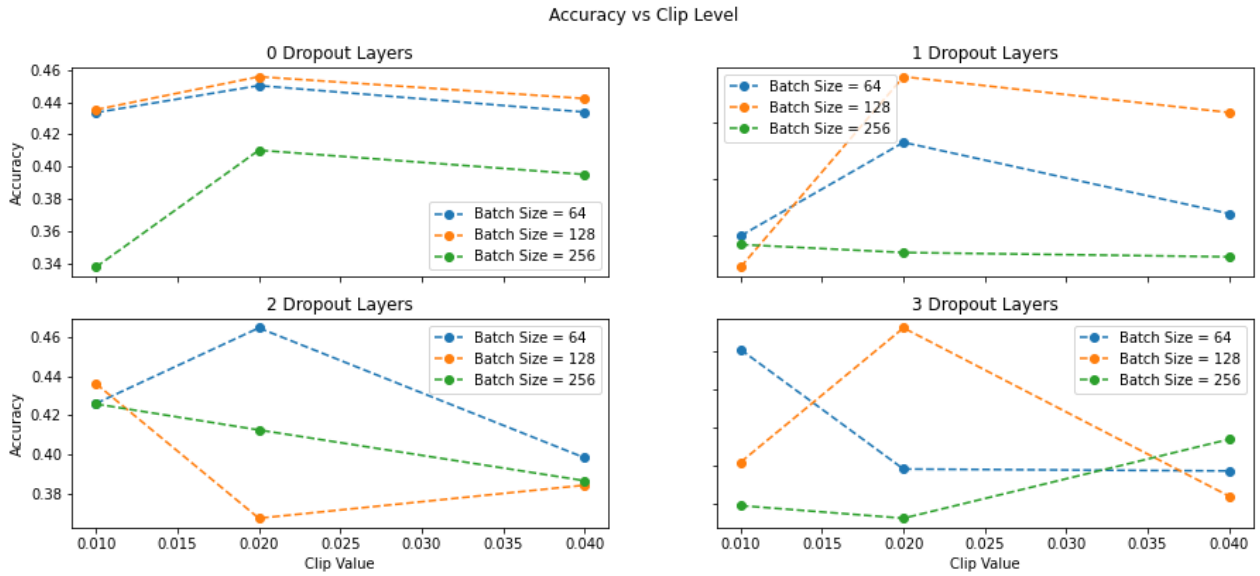


Figure 7: Accuracy vs Clip Level

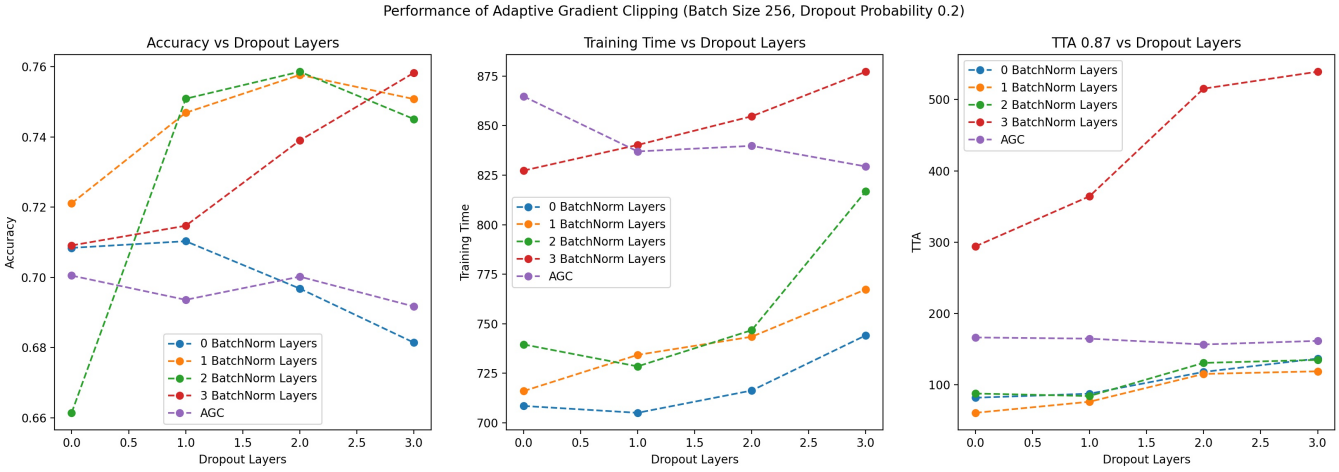
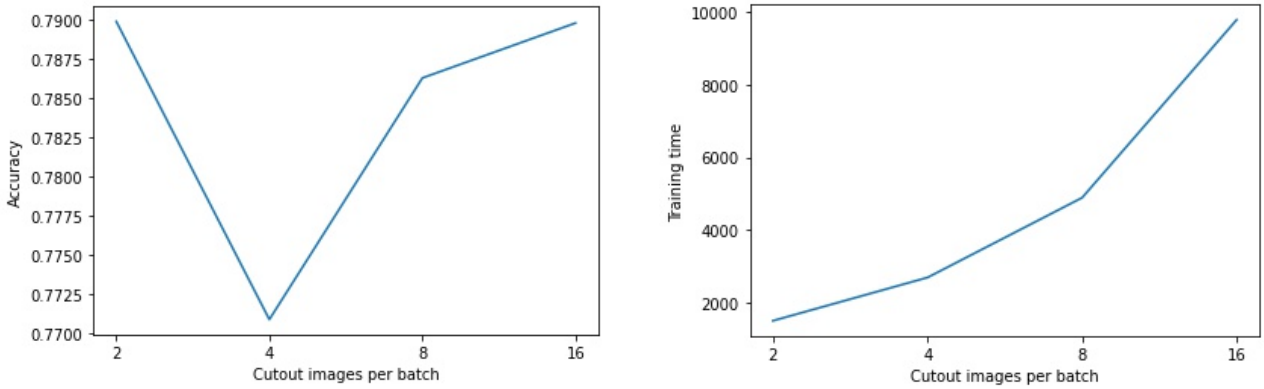


Figure 8: AGC performance compared to Batch Normalization

Data Augmentation

For cutout augmentation we first wanted to establish the *number of cutout images* per batch: we ran a short training cycle of 100 epochs with 2, 4, 8 and 16 new images per batch with a cutout of 12x12 black square. Due to hardware limitations we had to stop at 16: our training environment would run out of memory if a batch size became too large. Based on data from Figure 9, we selected 3 values: **2, 8** and **16** (taking into consideration both accuracies and training time of those parameters).



(a) Accuracy vs # of Cutout Images

(b) Training time vs # of Cutout Images

Figure 9: Selecting Cutout Parameter

The overall comparison of different data augmentation techniques is presented in Figure 10.

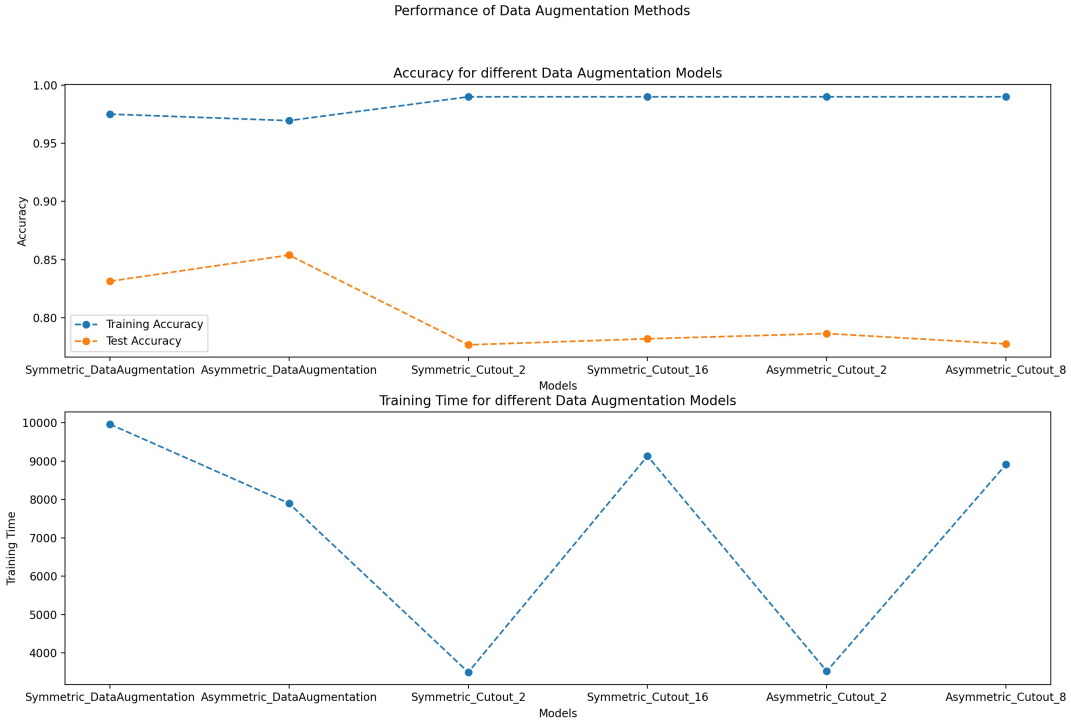


Figure 10: Overall performance with different Data Augmentation Techniques
The method that decreased overfitting the most is regular data augmentation with simple transformations. It also took less time than some of the cutout examples, with 8 and 16 cutout images per batch.

5 Observations

A comparison of the Resnet50 Models with the best parameter candidates is shown in Figure 11. Here, we compare the following 6 models:

1. The baseline model with no regularization (Baseline).

2. A model with only batch normalization and symmetric dropout (BatchNorm_Symmetric_Dropout). This contains 2 batch normalization layers and 3 dropout layers with a dropout probability of 0.2 across all layers.

3. A model with symmetric dropout and data augmentation (Symmetric_DataAugmnetation). Image transformations are used to augment the dataset, which contains 2 batchnorm layers and 3 dropout layers with a 0.2 dropout probability across layers.

4. A model with asymmetric dropout and data augmentation (Asymmetric_DataAugmentation). Image transformations are used to augment the dataset, which contains 2 batchnorm layers and 3 dropout layers with a 0.1, 0.2 and 0.3 dropout probability across each layer.

5. A model with symmetric dropout and cutout regularization (Symmetric_Cutout_2). The model contains 2 batchnorm layers and 3 dropout layers with a 0.2 dropout probability across layers. The number of cutput images per batch is 2.

6. A model with asymmetric dropout and cutout regularization (Asymmetric_Cutout_2). The model contains 2 batchnorm layers and 3 dropout layers with a 0.1, 0.2 and 0.3 dropout probability across each layer. The number of cutout images per batch is 2.

Here, we can see that we achieve the best accuracy with the model with asymmetric dropout and data augmentation. It can also be seen that this model has the lowest overfitting, as it has the least difference between the training and testing accuracy.

6 Conclusion

Using rigorous experiments, we are able to obtain a model that obtains an 85% test accuracy on the CIFAR-10 dataset using only batch normalization, dropout and data augmentations. This is an almost 15% increase as compared to base methods.

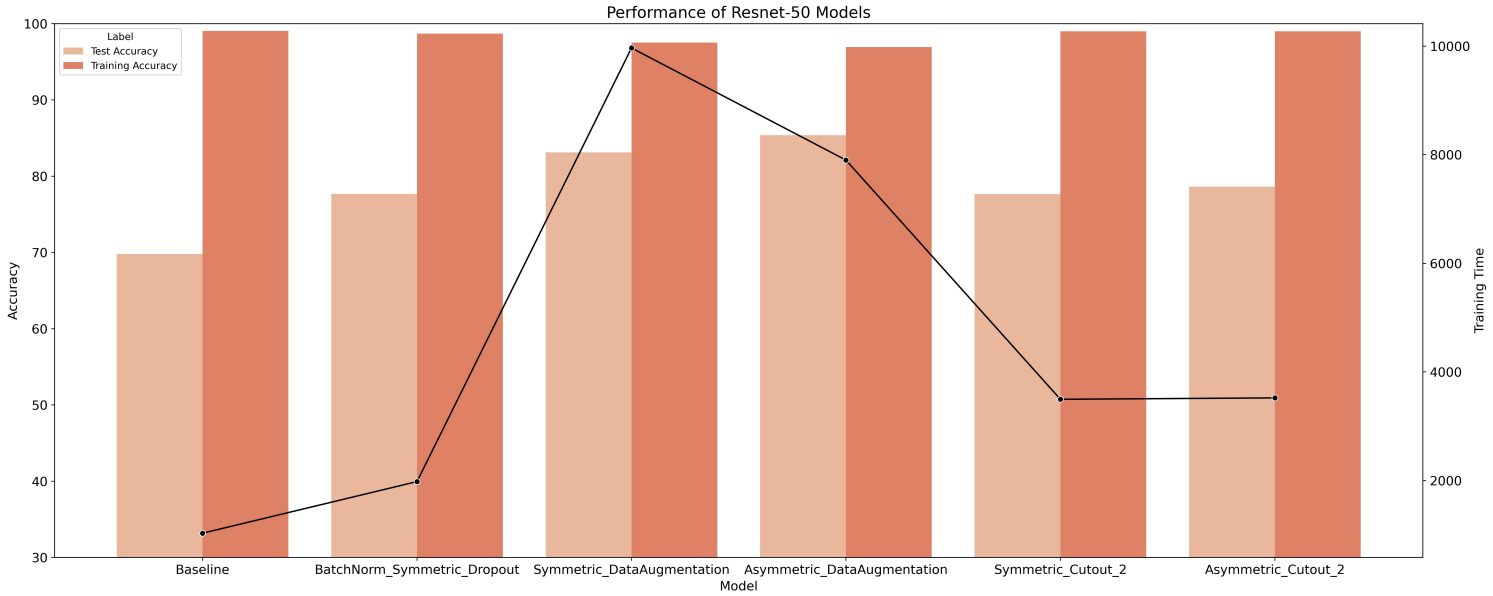


Figure 11: Comparison of Final Models

The Baseline model gives the worst performance with a 69% test accuracy, while model with asymmetric dropout of [0.1, 0.2, 0.3], 2 batch normalization layers and simple data augmentation achieves an 87% accuracy with least overfitting.

This study also concludes that while Adaptive Gradient Clipping cannot replace Batch Normalization entirely, it may be prudent to experiment on AGC with a model with more robust regularizers, and better optimizers.

References

[1] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.

[2] J. Bjorck, C. P. Gomes, and B. Selman, "Understanding batch normalization," *CoRR*, vol. abs/1806.02375, 2018. [Online]. Available: http://arxiv.org/abs/1806.02375

[3] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[4] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," 2017.

[5] A. Brock, S. De, S. L. Smith, and K. Simonyan, "High-performance large-scale image recognition without normalization," 2021.

[6] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.