
EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

Abstract

In this paper, different characteristics of CNN Architecture were manipulated to evaluate accuracy scores on how well the CNN architect was performing on class classification of images from CIFAR10. Some characteristics include the amount of layers, pooling type [average and max] and different activation functions [Relu, LeakyRelu, Tahn, and EIU]. The optimization was done by Adam or SDG while optimizing each architecture to compare how each performs. After the dropout and batching were added to the worst performing model, based on the performance on the activation function accuracy average. CIFAR has 10 classes, with a total of 6000 images per class, with a total count of 60000 images [32x32 color images]. 50000 of the images are for training and 10000 are for testing. All of the models' accuracies were done in two trials to report effective findings.

1. Introduction

For this project, the CIFAR10 ([CIFAR-10 and CIFAR-100 datasets \(toronto.edu\)](#)) dataset was used to evaluate how well different CNN architectures perform when manipulating characteristics. The dataset was normalized and transformed into a tensor. The device that was used to compute the CNN was cuda. The CIFAR dataset has 10 classes ['plane', 'car', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', and 'truck'], with 6000 images per class, with a total count of 60000 images [32x32 color images]. The dataset was pre-split where 50000 of the images are for training and 10000 are for testing. The testing accuracy score was used to measure how well each architecture performs. After obtaining each of the models overall accuracy, the worst activation average accuracy was obtained to investigate how drop out and batching normalization to manipulate the accuracy of the models.

For each CNN there were 3 convolution layers, 3 pooling layers [max or average], 2 fully

linear connected layers, and 4 activations [ReLU ([ReLU — PyTorch 2.2 documentation](#)), ELU ([ELU — PyTorch 2.2 documentation](#)), LeakyReLU ([LeakyReLU — PyTorch 2.2 documentation](#)), and Tanh ([Tanh — PyTorch 2.2 documentation](#))]. The loss function for all the models was cross-entropy ([CrossEntropyLoss — PyTorch 2.2 documentation](#)) to maintain consistency. The optimizers included Adam ([Adam — PyTorch 2.2 documentation](#)) and SGD ([SGD — PyTorch 2.2 documentation](#)). The code for this project was written using Python, using torch, matplotlib, and numpy. The developmental environment to write the python was done in a Jupyter notebook.

2.1 Methodology

2.1.1 Convolution

The convolution layers for all the models were the same; there were 3 layers. The first layer's inc channels was 3 and the out channels were 128. With a 3x3 kernel size. The second layer had 128 in channels and 256 out channels, with a 3x3 kernel size. The third layer had 256 in channels and 512 out channels, with a 3x3 kernel size.

2.1.2 Pooling

The pooling size for every layer was the same for each model. There were 3 pooling layers. The pooling size was a 2x2 kernel - for each layer. There were two types of pooling methods used: Max pooling and Average pooling. Max pooling helps to preserve the features, while average pooling down samples.

2.1.4 Flattening layer

There was one flatten layer added that was used to make the high dimensional array into a one dimensional array. This will play a

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

role in helping the fully connected layer to be used.

2.1.5 Fully Connected Layer

There were two linear fully connected layers. The first layer was a 2048x1024 and the second layer was a 1024x64. This was used to optimize class scores.

2.1.6 Dropout

For the worst performing activation accuracy average, a dropout layer was added in order to adjust if there is overfitting. This will help the model to prevent it from relying heavily on certain neurons. The dropout layer was set to 0.5 and 0.2 meaning that half of the outputs will be set to a value of 0 during training for 0.5.

2.1.7 Batching normalization

This was used for faster convergence and to regularize the mini batches. This is used to make each dimension zero-mean unit-variance.

2.1.8 Activation Function

There were a total of 4 different types of activation functions being used. This included ReLU, LeakyRelu, ELU, and Tanh. Each function was given the same parameters in order to maintain consistency to evaluate how well each performed. There are advantages and disadvantages when using each respective activation function.

2.1.9 Loss Function

The loss function used was cross entropy loss function - a common loss function to use. Cross entropy loss is useful when observing the

loss of information in classification problems, such as what CNN is doing.

2.1.10 Optimizer

There were 2 optimizers used, Adam and SGD each had a learning rate of 0.001 to maintain consistency. SGD updates parameters based on the gradient of the loss function, while Adam adjusts the learning rate relative to the history of the gradients.

2.2 General Approach

There was one CNN with the same architecture, the only difference was the activation function [Relu, LeakyRelu, Tahn, and ELU], pooling [max and average], and the optimizer [Adam and SGD]. The models were trained using one of the activation functions and ran through an SGD and Adam, later the model's accuracy was tested using the test data. Where the accuracy score was recorded for the model and for all the classes [10 classes]. This was done twice to ensure there were accurate accuracies from the models. After this the worst performing activation average accuracy were noted in order to evaluate how batch normalization and dropout can impact the accuracy of the model [without having to change the dimensions].

2.2.1 ReLU with Average Pooling

In the first trial the average between both Adam and SGD was 63% accuracy. Adam had a 71% accuracy, while SGD had a 55% accuracy. Results in image 1.

In the second trial the accuracies were 63.50% for the average of both Adam and SGD. Adam had an accuracy score of 72%, while

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

SGD had an accuracy of 55%. Results in image 2.

The scores between both trials are very similar meaning that ReLU with Average pooling is consistent with trials, giving a confident average score of ~71.50% accuracy when using Adam on ReLU with average pooling. While SGD average accuracy score was ~ 55%.

2.2.2 ReLU with Max Pooling

In the first trial the average between Adam and SGD the accuracy score was 69%. Where Adam's accuracy score was 71% and SGD was 67%. Results in image 1.

In the second trial the average accuracy score between Adam and SGD was 68%. Where Adam had an accuracy of 69%, while SGD had an accuracy of 67%. Results in image 2.

The scores between both trials are very similar meaning that ReLU with Max pooling is consistent with trials, giving a confident average score of ~70% accuracy when using Adam on ReLU with Max pooling. While SGD average accuracy score was ~ 67%.

2.2.3 ELU with Average Pooling

In the first trial the average between Adam and SGD the accuracy score was 10%. Where Adam's accuracy score was 10% and SGD was 10%. This could indicate that there was some type of overfitting going on. When looking at all the classes' accuracy score it clearly shows that there was over-fitting going on. Adam was overfitting on ships with an accuracy score of 100%. SGD was overfitting on frogs with an accuracy score of 100%. Results in image 1.

In the second trial the average accuracy score between Adam and SGD was 56%. Where Adam had an accuracy of 62%, while SGD had an accuracy of 50%. The layers were not manipulated in any way. This could indicate that the model overfits when it comes across a certain neuron. Making this inconsistent. Results in image 2.

The scores between both trials are not similar, meaning that ELU with average pooling is not consistent with trials, giving a low average score of ~36% accuracy when using Adam on ELU with average pooling. While SGD average accuracy score was ~ 30%.

2.2.4 ELU with Max Pooling

In the first trial the average between Adam and SGD the accuracy score was 39%. Where Adam's accuracy score was 10% and SGD was 68%. Results in image 1.

In the second trial the average accuracy score between Adam and SGD was 66.50%. Where Adam had an accuracy of 64%, while SGD had an accuracy of 67%. Results in image 2.

The scores between both trials are very similar meaning that ELU with Max pooling is consistent with trials, giving an average score of ~37% accuracy when using Adam on ELU with Max pooling. While SGD average accuracy score was ~ 67%.

These scores could indicate that Adam is relying heavily on certain neurons causing it to over fit, when they are not there, causing it to have better accuracy.

2.2.5 Leaky ReLU with Average Pooling

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

In the first trial the average between both Adam and SGD was 47.50% accuracy. Adam had a 40% accuracy, while SGD had a 55% accuracy. Results in image 1.

In the second trial the accuracies were 55.50% for the average of both Adam and SGD. Adam had an accuracy score of 57%, while SGD had an accuracy of 54%. Results in image 2.

The scores between both trials are similar, meaning that Leaky ReLU with Average pooling is consistent with trials, giving an average score of ~48.5% accuracy when using Adam on Leaky ReLU with average pooling. While SGD average accuracy score was ~ 54.5%.

2.2.6 Leaky ReLU with Max Pooling

In the first trial the average between Adam and SGD the accuracy score was 50%. Where Adam's accuracy score was 33% and SGD was 67%. Results in image 1.

In the second trial the average accuracy score between Adam and SGD was 52.50%. Where Adam had an accuracy of 38%, while SGD had an accuracy of 67%. Results in image 2.

The scores between both trials are very similar meaning that Leaky ReLU with Max pooling is consistent with trials, giving a confident average score of ~35.50% accuracy when using Adam on Leaky ReLU with Max pooling. While SGD average accuracy score was ~ 67%.

2.2.7 Tanh with Average Pooling

In the first trial the average between both Adam and SGD was 47% accuracy. Adam had a 44% accuracy, while SGD had a 50% accuracy. Results in image 1.

In the second trial the accuracies were 46% for the average of both Adam and SGD. Adam had an accuracy score of 40%, while SGD had an accuracy of 52%. Results in image 2.

The scores between both trials are similar with Average pooling is consistent with trials, giving an average accuracy score of ~42% accuracy when using Adam on Tanh with average pooling. While SGD average accuracy score was ~ 51%.

2.2.8 Tanh with Max Pooling

In the first trial the average between Adam and SGD the accuracy score was 56.50%. Where Adam's accuracy score was 45% and SGD was 68%. Results in image 1.

In the second trial the average accuracy score between Adam and SGD was 53.50%. Where Adam had an accuracy of 40% while SGD had an accuracy of 67%. Results in image 2.

The scores between both trials are similar meaning that Tanh with Max pooling is consistent with trials, giving an average score of ~42.5% accuracy when using Adam on Tanh with Max pooling. While SGD average accuracy score was ~ 67.5%.

2.2.9 ELU with Average Pooling with Batch Normalization and Dropout 0.5

In the first trial the average between Adam and SGD the accuracy score was 65.50%.

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

Where Adam's accuracy score was 69% and SGD was 62%. Results in image 3.

In the second trial the average accuracy score between Adam and SGD was 65.50%. Where Adam had an accuracy of 63% while SGD had an accuracy of 68%. Results in image 4.

The scores between both trials are similar, meaning that ELU with average pooling, batch, and dropout of 0.5 is consistent with trials, with minor fluctuations, giving an average score of ~66% accuracy for Adam. While SGD's average accuracy score was ~65%.

2.2.10 ELU with Max Pooling with Batch Normalization and Dropout 0.5

In the first trial the average between Adam and SGD the accuracy score was 68.50%. Where Adam's accuracy score was 70% and SGD was 67%. Results in image 3.

In the second trial the average accuracy score between Adam and SGD was 68.50%. Where Adam had an accuracy of 70% while SGD had an accuracy of 67%. Results in image 4.

The scores between both trials are similar, meaning that ELU with max pooling, batch, and dropout of 0.5 is consistent with trials, giving an average score of ~70% accuracy for Adam. While SGD's average accuracy score was ~67%.

2.2.11 ELU with Average Pooling with Batch Normalization and Dropout 0.2

In the first trial the average between Adam and SGD the accuracy score was 70%. Where Adam's accuracy score was 70% and SGD was 70%. Results in image 5.

In the second trial the average accuracy score between Adam and SGD was 69%. Where Adam had an accuracy of 72% while SGD had an accuracy of 66%. Results in image 6.

The scores between both trials are similar, meaning that ELU with average pooling, batch, and dropout of 0.2 is consistent with trials, giving an average score of ~71% accuracy for Adam. While SGD's average accuracy score was ~68%.

2.2.12 ELU with Max Pooling with Batch Normalization and Dropout 0.2

In the first trial the average between Adam and SGD the accuracy score was 69.50%. Where Adam's accuracy score was 69% and SGD was 70%. Results in image 5.

In the second trial the average accuracy score between Adam and SGD was 69.50%. Where Adam had an accuracy of 69% while SGD had an accuracy of 70%. Results in image 6.

The scores between both trials are similar, meaning that ELU with max pooling, batch, and dropout of 0.2 is consistent with trials, giving an average score of ~69.25% accuracy for Adam. While SGD's average accuracy score was ~70%.

3. Discussion

The dimensions of the models were kept the same to maintain consistency, to avoid increasing the accuracy because of the dimensions. The accuracies of all the models with respect to their activation function, pooling type, and optimizer were recorded using a cross entropy loss function which was used to create a graph for each accuracy. This was used to create

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

a table of all the accuracies, while showing an image in the graph in the table with the trial number. Using this to capture the average accuracy for each activation function, pooling function, and optimizer over 2 trials.

After finding the ELU gave the overall closest accuracy score, which was done because of overfitting from both optimizers.

ELU was used to perform batch normalization and dropout with rates 0.5 and 0.2. When doing so they performed exceptionally well compared to the original ELU model with no batch normalization and no dropout rate. For trial 1 on a dropout rate of 0.2 the average accuracy score for this model was 70% for both Adam and SGD when max pooling, and when average pooling the average accuracy score was 69.50%. For trial 2 on a dropout rate of 0.2 the average for average pooling for both optimizers was 69%, while when max pooling it was 69.50. This shows it was consistent over trials, while without we see how the model jumped from a 10% accuracy score to ~60%. When it was 10% accurate it was overfitting for both SGD and Adam. For trial 1 on a dropout rate of 0.5 the average accuracy score when average pooling was 65.50%, while when max pooling it was 68.50%. For trial 2 on the dropout rate of 0.5 the average accuracy score when average pooling was 65.50%, while when max pooling it was 68.50%. We see that dropout rate of 0.2 outperformed the dropout rate of 0.5/

4. Conclusion

From these results it was shown that ReLU with average pooling and an Adam as the optimizer outperformed the rest of the

activation functions, yielding an average accuracy score for ReLU with pooling of 71%-72%; this may have multiple implications. This could indicate that Adam performs well when giving the downsampled information by being able to adapt using the downsample information. It was also found that adding batch normalization and dropout can help prevent overfitting. The accuracy score looks to be consistent over trials and even when manipulating the dropout rate it increases the accuracy score. Adding batch normalization and dropout prevented the model from overfitting and from being inconsistent. When doing a dropout rate of 0.2 it increased ELU's performance from 42.88% to 69.50% (when averaging both poolings average for both trials). This yielded a performance to a point of ReLU with average pooling (71.5%), with a difference of 2%. This could later be investigated to see how to increase the accuracy score, for models that overfit, in order to yield consistent high accuracies.

Some improvements for this could be to adjust the out channels that can help the activation functions work. Manipulating the dimensions in the model can impact the accuracies, potentially some the dimensions [3x128, 128x256, 256x512, 2048x1024, and 1024x64] are not compatible for specific activation functions or optimizers. This can potentially explain why we see ReLU with average pooling using Adam optimizer to perform the best.

















EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

5. References

1. *Adam*¶. Adam - PyTorch 2.2 documentation. (n.d.).
<https://pytorch.org/docs/stable/generated/torch.optim.Adam.html>
2. CIFAR-10 and CIFAR-100 datasets. (n.d.).
<https://www.cs.toronto.edu/~kriz/cifar.html>
3. *Convolutional Neural Networks cheatsheet star*. CS 230 - Convolutional Neural Networks Cheatsheet. (n.d.).
<https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks#layer>
4. *Crossentropyloss*¶. CrossEntropyLoss - PyTorch 2.2 documentation. (n.d.).
<https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html>
5. *Elu*¶. ELU - PyTorch 2.2 documentation. (n.d.).
<https://pytorch.org/docs/stable/generated/torch.nn.ELU.html>
6. *Final project*. UCSD COGS 181 Winter 2024. (n.d.).
<https://sites.google.com/view/ucsd-cogs-181-winter-2024/assignments/final-project?authuser=0>
7. *Leakyrelu*¶. LeakyReLU - PyTorch 2.2 documentation. (n.d.).
<https://pytorch.org/docs/stable/generated/torch.nn.LeakyReLU.html>
8. *Relu*¶. ReLU - PyTorch 2.2 documentation. (n.d.).
<https://pytorch.org/docs/stable/generated/torch.nn.ReLU.html>
9. *Sgd*¶. SGD - PyTorch 2.2 documentation. (n.d.).
<https://pytorch.org/docs/stable/generated/torch.optim.SGD.html>
10. *Tanh*¶. Tanh - PyTorch 2.2 documentation. (n.d.).
<https://pytorch.org/docs/stable/generated/torch.nn.Tanh.html>

















EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

Image 1:

Trial 1														
ROUNDS	Avg Pooling	Max Pooling	Relu	Elu	Leaky Relu	Tanh	Adam	SGD	Graph	Pooling Accuracy	Pooling Accuracy Average	Activation function Accuracy Average	Adam Average Accruacy	SGD Average Accruacy
1	Avg Pooling		Relu					SGD		55.00%	63.00%			
2	Avg Pooling		Relu				Adam			71.00%				
3		Max Pooling	Relu					SGD		67.00%				
4		Max Pooling	Relu				Adam			71.00%				
5	Avg Pooling			ELU				SGD		10.00%	10.00%			
6	Avg Pooling			ELU			Adam			10.00%				
7		Max Pooling		ELU				SGD		68.00%				
8		Max Pooling		ELU			Adam			10.00%				
9	Avg Pooling				LeakyReLU			SGD		55.00%	47.50%			
10	Avg Pooling				LeakyReLU		Adam			40.00%				
11		Max Pooling			LeakyReLU			SGD		67.00%				
12		Max Pooling			LeakyReLU		Adam			33.00%				
13	Avg Pooling					Tanh		SGD		50.00%	47.00%			
14	Avg Pooling					Tanh	Adam			44.00%				
15		Max Pooling				Tanh		SGD		68.00%				
16		Max Pooling				Tanh	Adam			45.00%				

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

Image 2:

Trial 2														
ROUNDS	Avg Pooling	Max Pooling	Relu	Elu	Leaky Relu	Tanh	Adam	SGD	Graph	Pooling Accrura	Pooling Accuracy Average	Activation function Accuracy Average	Adam Average Accruacy	SGD Average Accruacy
1	Avg Pooling		Relu					SGD		55.00%	63.50%			
2	Avg Pooling		Relu				Adam			72.00%				
3		Max Pooling	Relu					SGD		67.00%				
4		Max Pooling	Relu				Adam			69.00%				
5	Avg Pooling			ELU				SGD		50.00%	56.00%			
6	Avg Pooling			ELU			Adam			62.00%				
7		Max Pooling		ELU				SGD		69.00%				
8		Max Pooling		ELU			Adam			64.00%				
9	Avg Pooling				LeakyReLU			SGD		54.00%	55.50%			
10	Avg Pooling				LeakyReLU		Adam			57.00%				
11		Max Pooling			LeakyReLU			SGD		67.00%				
12		Max Pooling			LeakyReLU		Adam			38.00%				
13	Avg Pooling					Tanh		SGD		52.00%	46.00%			
14	Avg Pooling					Tanh	Adam			40.00%				
15		Max Pooling				Tanh		SGD		67.00%				
16		Max Pooling				Tanh	Adam			40.00%				
											53.50%	49.75%	40.00%	59.50%

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

Image 3:

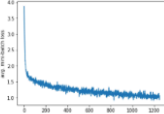
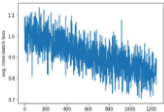
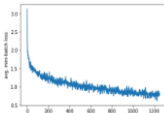
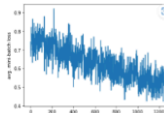
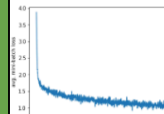
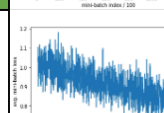
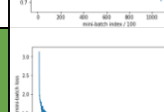
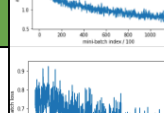
Trial 1											
With Dropout 0.5 and Batch Normalization											
ROUNDS	Avg Pooling	Max Pooling	Elu	Adam	SGD	Graph	Pooling Accuracy	Pooling Accuracy Average	Activation function Accuracy Average	Adam Average Accruacy	SGD Average Accruacy
1	Avg Pooling		ELU		SGD		62.00%				
2	Avg Pooling		ELU	Adam			69.00%	65.50%			
3		Max Pooling	ELU		SGD		67.00%				
4		Max Pooling	ELU	Adam			70.00%	68.50%	67.00%	69.50%	64.50%

Image 4:

Trial 2											
With Dropout 0.5 and Batch Normalization											
ROUNDS	Avg Pooling	Max Pooling	Elu	Adam	SGD	Graph	Pooling Accuracy	Pooling Accuracy Average	Activation function Accuracy Average	Adam Average Accruacy	SGD Average Accruacy
1	Avg Pooling		ELU		SGD		63.00%				
2	Avg Pooling		ELU	Adam			68.00%	65.50%			
3		Max Pooling	ELU		SGD		67.00%				
4		Max Pooling	ELU	Adam			70.00%	68.50%	67.00%	69.00%	65.00%

EVALUATING DIFFERENT CNN ARCHITECTURE'S CHARACTERISTICS – DEEP LEARNING

Image 5:

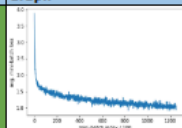
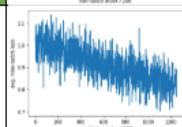
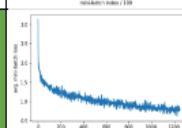
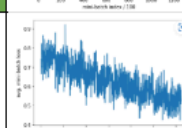
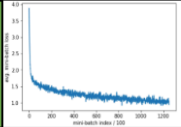
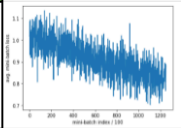
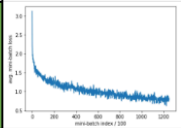
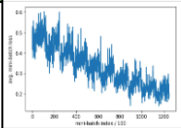
Trial 1											
With Dropout 0.2 and Batch Normalization											
ROUNDS	Avg Pooling	Max Pooling	Elu	Adam	SGD	Graph	Pooling Accuracy	Pooling Accuracy Average	Activation function Accuracy Average	Adam Average Accruacy	SGD Average Accruacy
											
1	Avg Pooling		ELU		SGD		70.00%				
											
2	Avg Pooling		ELU	Adam			70.00%	70.00%			
											
3		Max Pooling	ELU		SGD		70.00%				
											
4		Max Pooling	ELU	Adam			69.00%	69.50%	69.75%	69.50%	70.00%

Image 6:

Trial 2											
With Dropout 0.2 and Batch Normalization											
ROUNDS	Avg Pooling	Max Pooling	Elu	Adam	SGD	Graph	Pooling Accuracy	Pooling Accuracy Average	Activation function Accuracy Average	Adam Average Accruacy	SGD Average Accruacy
1	Avg Pooling		ELU		SGD		66.00%	69.00%			
2	Avg Pooling		ELU	Adam			72.00%				
3		Max Pooling	ELU		SGD		70.00%	69.50%			
4		Max Pooling	ELU	Adam			69.00%	69.50%	69.25%	70.50%	68.00%