Empirical evaluations of learning datasets with weight-sharing and locality

Michael Zhang
msz@mit.edu

Abstract

Convolutional neural networks have been widely used for image processing due to their effectiveness, and previous work has suggested that this potency arises from the weight-sharing and locality assumptions that CNNs make. In this project, we empirically investigate performance differences between making these assumptions with the model architecture versus attempting to learn them. To do so, we develop specialized neural network architectures to empirically evaluate the difficulty of learning weight-sharing versus the difficulty of learning locality in datasets that satisfies both properties. Specifically, the datasets examined include a simple contrived dataset, MNIST, and CIFAR10. Robustness of these architectures that make weight-sharing and/or locality assumptions are also studied. All the code for this project can be viewed at `https://github.com/michaelszhang/6.860-final-project`.

1 Introduction

Convolutional neural networks (CNNs) have proven to be effective at image classification tasks [4], in part due to the weight-sharing and locality assumptions they impose on image data. One particular advantage CNNs provide is the small number of necessary parameters due to these weight-sharing and locality assumptions, which allows for efficient model training [7]. Relatively low storage space, even for large models, is another added benefit that is utilized in many applications, such as autonomous driving [3]. The literature also suggests that there may be further efficiency improvements with more advanced weight-sharing architectures [1].

In addition to efficiency benefits, there is also evidence for computational advantages of CNNs over fully-connected networks (FCNs), despite the fact that sufficiently large FCNs can exactly mimic the parameters of CNNs. Previous work [5] has shown that there are tasks solved by CNNs that are hard to learn for FCNs specifically due to the lack of locality in FCNs. Others [6] also claim that weight sharing is crucial to optimization as well.

These points provide the motivation for this project. In particular, we seek to empirically evaluate the difficulty of learning weight-sharing and/or locality using model architectures that do not assume those properties.

2 Binary tree dataset

2.1 Dataset description

A simple contrived dataset satisfying both weight-sharing and locality is first examined; we initially select a simple dataset in order to draw conclusions regarding the number of samples required for various architectures to train well, rather than the accuracy of the architectures on more complex datasets.

6.860/9.520[J]: Statistical Learning Theory and Applications (Fall 2021).

Architecture	Parameters in k^{th} layer	Action of k^{th} layer
`WeightNet`	2^{7-k}	Computes $\mathbf{z}' = (\langle w, C(\mathbf{z}, 0)\rangle, \langle w, C(\mathbf{z}, 2)\rangle, \ldots, \langle w, C(\mathbf{z}, 2^{7-k}-2)\rangle)$, where $C(\mathbf{z}, c)$ is the cyclic shift of $\mathbf{z}$ by c (e.g. $C(\mathbf{z}, 2) = (z_3, z_4, \ldots, z_2)$) for $w \in \mathbb{R}^{2^{7-k}}$.
`LocalNet`	2^{7-k}	Computes $\mathbf{z}' = (\langle w_1, (z_1, z_2)\rangle, \langle w_2, (z_3, z_4)\rangle, \ldots, \langle w_{2^{6-k}}, (z_{2^{7-k}-1}, z_{2^{7-k}})\rangle)$ for $w_1, w_2, \ldots, w_{2^{6-k}} \in \mathbb{R}^2$.
`OracleNet`	2	Computes $\mathbf{z}' = (\langle w, (z_1, z_2)\rangle, \langle w, (z_3, z_4)\rangle, \ldots, \langle w, (z_{2^{7-k}-1}, z_{2^{7-k}})\rangle)$ for $w \in \mathbb{R}^2$.
`FCNet`	2^{13-2k}	Computes $\mathbf{z}' = W\mathbf{z}$ for $W \in \mathbb{R}^{2^{6-k} \times 2^{7-k}}$.

Table 1: Description of `WeightNet`, `LocalNet`, `OracleNet`, and `FCNet` layers.

We now proceed to describing the contrived distribution. Data points $(\mathbf{x}, y)$ with $x \in \mathbb{R}^{64}$ and $y \in \mathbb{R}$ in this dataset are drawn independently from

$$p(\mathbf{x}, y) = p(x)p(y|\mathbf{x}), \quad p(\mathbf{x}) \sim \mathcal{N}(\mathbf{0}, I_{64}), \quad p(y|\mathbf{x}) \sim \mathcal{N}(f_\theta(\mathbf{x}), \epsilon),$$

where ϵ is a small noise parameter and if $\mathbf{x} = (x_1, x_2, \ldots, x_{64})$, then $f_\theta(\mathbf{x})$ is described by

$$\begin{aligned} f_\theta(\mathbf{x}) &= f_{6,\theta}(x_1, x_2, \ldots, x_{64}) \\ &= \theta_{61} \tanh(f_{5,\theta}(x_1, x_2, \ldots, x_{32})) + \theta_{62} \tanh(f_{5,\theta}(x_{33}, x_{34}, \ldots, x_{64})) \\ &= \theta_{61} \tanh(\theta_{51} \tanh(f_{4,\theta}(x_1, x_2, \ldots, x_{16})) + \theta_{52} \tanh(f_{4,\theta}(x_{17}, x_{18}, \ldots, x_{32}))) \\ &+ \theta_{62} \tanh(\theta_{51} \tanh(f_{4,\theta}(x_{33}, x_{34}, \ldots, x_{48})) + \theta_{52} \tanh(f_{4,\theta}(x_{49}, x_{50}, \ldots, x_{64}))) \\ &= \ldots. \end{aligned}$$

Intuitively, the evaluation of $f_\theta(\mathbf{x})$ can be viewed as passing values up a binary tree, where all edges pointing towards left children on a given layer share a weight, and all edges pointing towards right children on a given layer also share a weight.

2.2 Model architectures

Experiments with four different model architectures, `WeightNet`, `LocalNet`, `OracleNet`, and `FCNet`, are conducted on the contrived dataset. In order to evaluate the difficulty of learning weight-sharing and locality, `WeightNet` assumes weight-sharing only, `LocalNet` assumes locality only, `OracleNet` assumes both, and `FCNet` assumes neither. To facilitate accurate computation of $f_\theta(\mathbf{x})$, all architectures have 6 layers, the latent space representation of $\mathbf{x}$ after the k^{th} layer has dimension 2^{6-k}, and the $\tanh$ activation function is used after every layer except the last. Furthermore, no biases are used in any architecture. Descriptions of the architectures are given in table 1.

2.3 Experiments

Varying training set sizes n_{train} and error parameters ϵ were tested; multiple trials of each combination of $(n_{\text{train}}, \epsilon) \in \{64, 128, 256, 512, 1\,024, 2\,048, 4\,096\} \times \{0, 0.1, 0.3\}$ were completed. During each trial, the parameters $\theta = (\theta_{11}, \theta_{12}, \ldots, \theta_{62}) \in \mathbb{R}^{12}$ were first sampled from the uniform distribution $\mathcal{U}(0, 2)^{12}$. Then, n_{train} training points were sampled independently from $p(x, y)$ *with* ϵ noise, and $n_{\text{text}} = 1\,024$ test points were sampled independently *without* ϵ noise. All four model architectures were then trained on the same training set, and evaluated using the same test set.

2.4 Training details

For training, a batch size of 32 was used, and the trials using 64, 128, 256, 512, 1 024, 2 048, and 4 096 training examples were trained for 800, 700, 600, 500, 400, 300, and 200 epochs, respectively. The Adam optimizer and mean-squared error loss function were used. In addition, training began with learning rate equal to `1e-3` with the learning rate decreasing to `1e-4` after half of the training epochs were completed.

All experiments were done using the Pytorch library on Google Colaboratory with a Tesla P100 GPU.

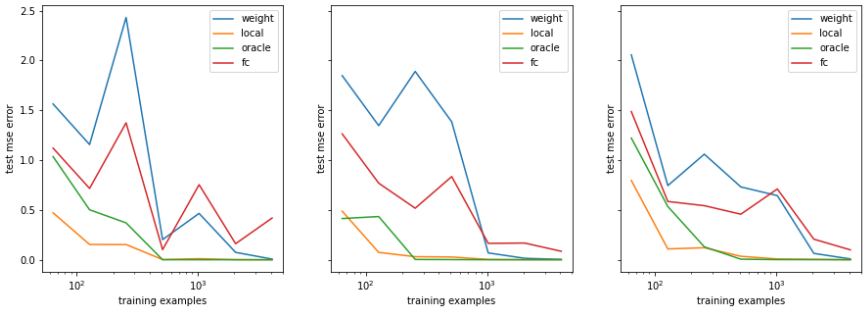

Figure 1: Average test mean-squared error versus number of training examples for $\epsilon = 0$ (left), $\epsilon = 0.1$ (middle), $\epsilon = 0.3$ (right).

2.5 Results and discussion

The average test mean-squared error across all trials versus number of training examples is plotted in figure 1. There is a negative correlation between test error and number of training examples, which is as expected. The reason why error is not monotonically decreasing can be attributed to random noise caused by sampling a new θ every trial: larger values of θ parameters typically leads to large y values, which results in greater mean-squared error.

Furthermore, `LocalNet` and `OracleNet` outperform `WeightNet` and `FCNet` for all values of n_{train} and ϵ, which suggests that weight-sharing is easier to learn than locality. However, for larger values of n_{train}, `WeightNet` also approaches zero error, indicating that locality can indeed be learned for this contrived distribution, but doing so requires more training examples.

The architectures that assume locality also appear to be more robust to random noise in the training set. In particular, for $\epsilon = 0.3$, both `LocalNet` and `OracleNet` still achieve low test error within hundreds of training examples, while `WeightNet` and `FCNet` require at least 2048 to do so. Intuitively, this may be because architectures assuming locality only solve a two-variable regression-like problem for their weights, and the true regression mean becomes fairly accurate after a few hundred examples even for large noise. On the other hand, architectures that do not need much more examples when noise is large since they must see enough data to eliminate many dependencies.

The specific learned weights of `WeightNet` and `LocalNet` can also be examined and compared to the true weights parameters of θ. Histograms of first-layer weights for both `WeightNet` and `LocalNet` are displayed in figure 2 for one trial of 512 training examples and another trial of 4096 training examples.

The true parameters for the $n_{\text{train}} = 512$ trial are $\theta_{11} = 0.6248$ and $\theta_{12} = 0.4908$. If `WeightNet` trained perfectly, we should expect to see one weight at 0.6248 in the θ_{11} histogram (top, first column) and the rest at 0, thus learning locality. For `LocalNet`, we wish to see all weights in its θ_{11} histogram (top, third column) be 0.6248. This does not appear to be the case for `WeightNet` at all, and `LocalNet` has a distribution centered around the true θ_{11} and θ_{12} but with high variance.

When the number of training examples increases to $4\,096$, the learned weights become closer to the true parameters, indicating that both weight-sharing and locality can be learned given enough training examples. The true parameters for this trial are $\theta_{11} = 0.4973$ (first and third histograms) and $\theta_{12} = 1.8477$ (second and fourth histograms).

3 Benchmark datasets

3.1 Dataset description

We now evaluate weight-sharing and locality on the common benchmark datasets MNIST and CIFAR10. MNIST contains 28×28 black-and-white images of handwritten digits, with $60\,000$ training examples and $10\,000$ test examples, while CIFAR10 contains 32×32 color images of 10 classes of objects, with $50\,000$ training examples and $10\,000$ test examples.

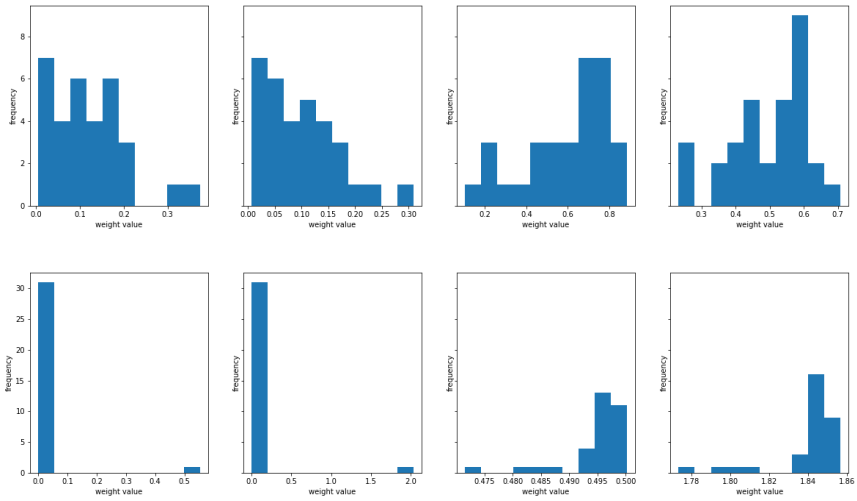

Figure 2: Learned weights in the first layer of `WeightNet` (left half) and `LocalNet` (right half) on trials with $n_{\text{train}} = 512$ (top), $n_{\text{train}} = 4\,096$ (bottom), and $\epsilon = 0$.

3.2 Model architectures

As with the binary tree dataset, we wish to test models with that assume weight-sharing only, locality only, both, and neither. For the case of assuming both, `ConvNet`, a vanilla CNN, is used. `ConvNet` contains two convolutional layers of kernel size 5, each followed by a max-pool of kernel size 2, transforming the image to 6 and then 16 channels. Then, the features are flattened into a vector (256-dimensional for MNIST, 400-dimensional for CIFAR10) and three fully-connected layers transform the latent dimension to 120, 84, and finally 10, where the final 10-dimensional output serves as a prediction for the 10 classes in both datasets. In addition, the ReLU activation function is used after every layer.

To construct the other architectures, we modify the `ConvNet` architecture while maintaining the same number of parameters for the latent representation after each convolutional+pooling layer is applied. For instance, on the MNIST dataset, an image will be transformed into a $6 \times 12 \times 12$ representation after the first convolutional+pooling layer, so the three other architectures will also have 864 parameters in their first latent representation. All three architectures also keep the final three fully-connected layers and the ReLU activation function.

As before, the three other architectures are named `FCNet`, `WeightNet`, and `LocalNet`. `FCNet` simply replaces the convolutional layers with fully-connected layers of the proper dimension, and assumes neither weight-sharing nor locality. `WeightNet` assumes weight-sharing, and implements this by replacing the kernel-size 5 convolutional layers with a convolutional layer that has kernel size equal to the input width/height, with appropriate padding to maintain the correct latent dimensions. Finally, `LocalNet` replaces each convolutional layer with many convolutional layers that act only on one specific kernel of the imag to assume locality only.

3.3 Experiments

Varying training set sizes n_{train} were tested; multiple trials of each value of $n_{\text{train}} \in \{10, 30, 100, 300, 1000, 3000, 10000, 30000\}$ were completed. For each trial, a random balanced size-n_{train} subset of the training dataset is selected, and all four architectures are trained on that subset. To evaluate test accuracy, the entire 10 000-example test set was used.

In addition, to test model robustness, the fast gradient sign method (FGSM) [2] was used to generate adversarial examples. For both datasets, each of the 10 000 testing images were converted to adversarial examples using maximum L^∞ pertubations of 0.1 for MNIST and 0.01 for CIFAR10, and the adversarial accuracy was subsequently evaluated.

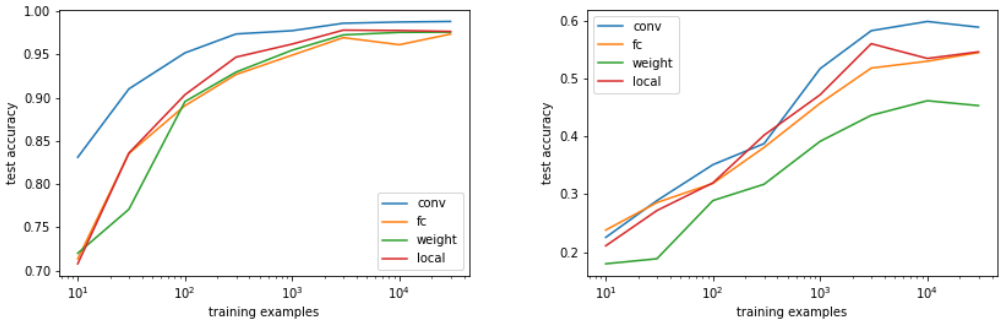

Figure 3: Average test accuracy versus number of training examples for MNIST (left) and CIFAR10 (right).

Dataset	n_{train}	Architecture	Original accuracy	Adversarial accuracy	Change
MNIST	1 000	`WeightNet`	0.9549	0.7777	-18.56%
MNIST	1 000	`LocalNet`	0.9620	0.7879	-18.09%
MNIST	10 000	`WeightNet`	0.9755	0.8373	-14.17%
MNIST	10 000	`LocalNet`	0.9777	0.8464	-13.43%
CIFAR10	1 000	`WeightNet`	0.3914	0.2989	-23.64%
CIFAR10	1 000	`LocalNet`	0.4719	0.2829	-40.04%
CIFAR10	10 000	`WeightNet`	0.4616	0.3469	-24.85%
CIFAR10	10 000	`LocalNet`	0.5346	0.3241	-39.37%

Table 2: Average adversarial accuracies of selected trials after applying FGSM.

3.4 Training details

For both datasets, a batch size of 4 was used, and the trials using 10, 30, 100, 300, 1 000, 3 000, 10 000 and 30 000 training examples were trained for 40, 40, 20, 20, 10, 10, 5, and 5 epochs, respectively. The SGD optimizer with a learning rate of `1e-3` and momentum 0.9 were used along with the cross-entropy loss function.

All experiments were done using the Pytorch library on Google Colaboratory with a Tesla P100 GPU.

3.5 Results and discussion

The results of the accuracy experiments are summarized in figure 3. As expected, there was a positive correlation between number of training examples and average test accuracy for all architectures and both datasets. For MNIST, we see that `ConvNet` performs best, and the performance of the other three models, as evaluated by accuracy, are comparable. On the other hand, for CIFAR10, `WeightNet` performs substantially worse than the other architectures for all values of n_{train}, suggesting that it may be more difficult to learn locality. Here, we note that although the performance of `FCNet` is comparable to the other two architectures, it also has much more parameters with which to fit the data, so it is not necessarily the case that `FCNet` has learned locality.

In addition, the results of applying FGSM to determine adversarial accuracies are displayed in table 2. For MNIST, the robustness of `WeightNet` and `LocalNet` are approximately the same, and robustness increases as number of training examples increases. However, for CIFAR10, `LocalNet` is far less robust than `WeightNet`, and increasing the number of training examples does not appear to affect robustness.

Overall, neither of these architectures appear to be very robust. Samples from MNIST and CIFAR10 before and after applying FGSM are provided in the appendix, and in particular, the FGSM changes on CIFAR10 are indistinguishable to the human eye but impact model performance greatly.

4 Conclusion and future work

All in all, experiments on the contrived binary tree dataset seem to support the hypothesis that locality is harder and takes more examples to learn than weight-sharing, and that architectures assuming locality tend to be more robust. Both `WeightNet` and `LocalNet` were able to learn the correct true weights with enough training examples. On the contrary, results from the standard benchmark datasets MNIST and CIFAR10 indicate that while locality may be more difficult to learn, it is much less conducive to robustness.

Conducting these experiments has also led to ideas for future exploration. These include obtaining less noisy results for the binary tree dataset by fixing θ, testing more complex architectures for benchmark datasets that may be more conducive to correctly learning weight-sharing and/or locality, training on adversarial examples, and theoretical work for explaining some of the binary tree dataset results.

References

[1] Shubhra Aich et al. *Multi-Scale Weight Sharing Network for Image Recognition*. 2020. arXiv: 2001.02816 [cs.CV].

[2] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015. arXiv: 1412.6572 [stat.ML].

[3] Sorin Grigorescu et al. "A survey of deep learning techniques for autonomous driving". In: *Journal of Field Robotics* 37.3 (Apr. 2020), pp. 362–386. ISSN: 1556-4967. DOI: 10.1002/rob.21918. URL: http://dx.doi.org/10.1002/rob.21918.

[4] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[5] Eran Malach and Shai Shalev-Shwartz. *Computational Separation Between Convolutional and Fully-Connected Networks*. 2020. arXiv: 2010.01369 [cs.LG].

[6] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. *Weight Sharing is Crucial to Succesful Optimization*. 2017. arXiv: 1706.00687 [cs.LG].

[7] Mingxing Tan and Quoc V. Le. *EfficientNetV2: Smaller Models and Faster Training*. 2021. arXiv: 2104.00298 [cs.CV].

5 Appendix

5.1 Samples after FGSM

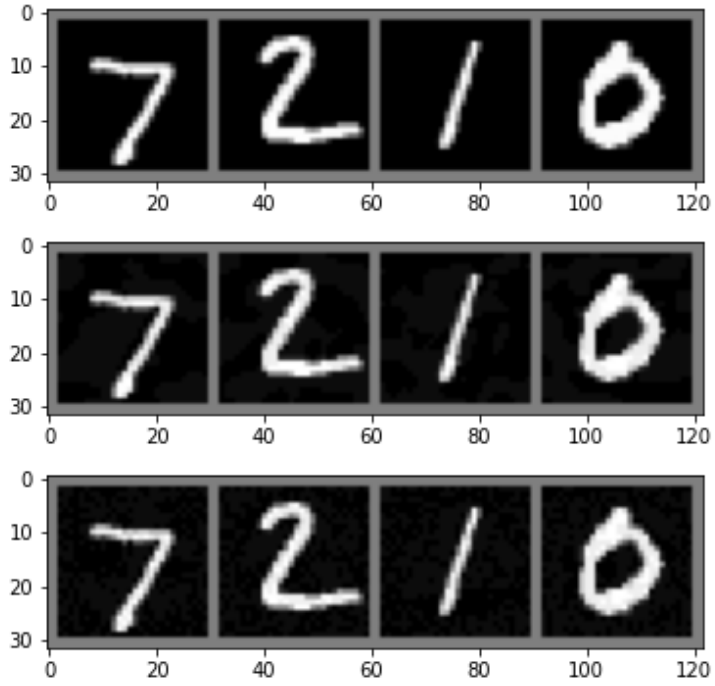

Figure 4: Original MNIST samples (top), samples after FGSM on a `WeightNet` trained on 10 000 examples (middle), samples after FGSM on a `Local` trained on 10 000 examples (bottom).

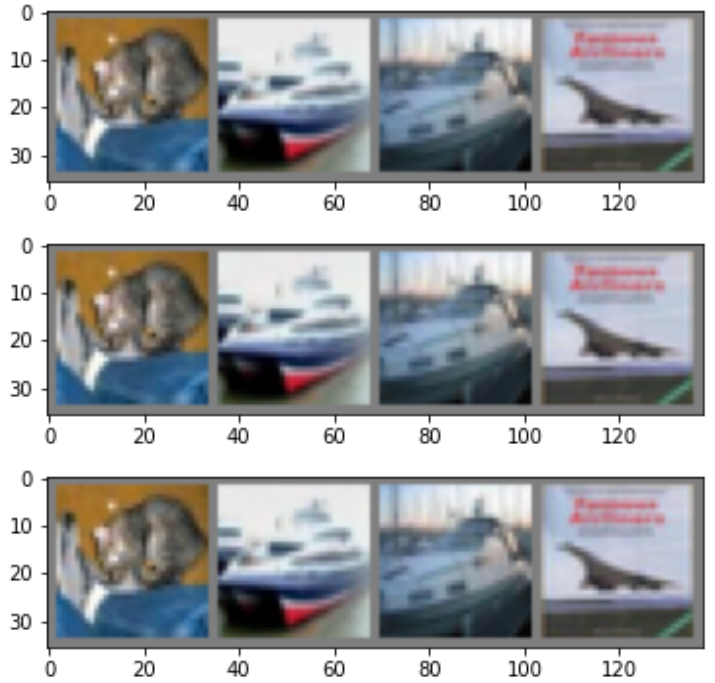

Figure 5: Original CIFAR10 samples (top), samples after FGSM on a `WeightNet` trained on 10 000 examples (middle), samples after FGSM on a `Local` trained on 10 000 examples (bottom).