Evaluating Capsule Network Performance on Multiple Datasets

ANANT BHARGAVA (ABHARGA7)[1], KATIE LINK (KLINK2)[1], AND JIN YOUNG SOHN (JSOHN16)[1]

[1] *Johns Hopkins University, Baltimore, MD 21218, USA*

Compiled December 19, 2017

Since the early 2010's, convolutional neural network approaches have shown revolutionary performance on computer vision tasks; however, they show limitations that ultimately hamper their performance and flexibility. A new type of deep learning architecture, called capsule networks, confront these issues in convolutional neural networks by introducing several new features that attempt to loosely recreate our parallelized, attentional visual system. By encoding feature information in a vector rather than a scalar, the layers of the network can encode both visual feature and pose information in the magnitude and direction, respectively, of the neuron. Multiple capsules are trained to extract general feature information and are dynamically routed to the appropriate classifying capsule. Because of their sensitivity to pose and general features of a class, they are reported to perform better on classifying tasks of objects with different orientations or obscured objects, for example, two tasks that convolutional neural networks have traditionally struggled with. In this paper, we present an implementation of the capsule network for multiple different datasets: MNIST, Fashion-MNIST, CIFAR-10, and CIFAR-100. Along with testing different parameters of the model, such as learning rate, epochs, and augmentation, we also experimented slight modifications to the architecture as well as tried a new loss function for the more complex datasets, CIFAR-10 and CIFAR-100, based on intuition of the capsule network. We have compared these results with the original paper, other capsule network implementations, and/or with the state-of-the-art convolutional neural networks, as well as between datasets to determine limitations of the implementation and capsule networks in general.

OCIS codes:

https://github.com/jysohn23/CapsuleNetwork/

1. BACKGROUND

Computer vision scientists have long struggled with vision tasks that come deceptively naturally to humans. From a young age, children are capable of discriminating among large numbers of objects, learning their names and functions quickly, as well as recognizing unique faces such as those of their parents. From a neuroscientific standpoint, we can attribute these amazing visual feats to our large visual cortex, which is comprised of multiple parallel processing pathways. These pathways extract visual features from outputs of the retina by processing them through many layers of neurons.

Because of incredible capabilities of the human visual system, computer scientists have long tried to imitate the structure of the visual cortex in order to achieve similar performance on difficult tasks. The single layer perceptron, and later the multilayer perceptron were among the first imitations, but were limited by the extreme number of parameters needed to trained for an image. In the early 2010's, increased graphics processing speed from GPUs combined with huge datasets allowed the rapid development of convolutional neural networks, which have shown incredible, and on some tasks, superhuman capabilities. In general, these deep learning networks utilize convolutional layers, which learn a filter bank in decreasing abstraction, pooling layers, which pool together the representations and add some spatial orientation, and a series of fully connected layers, which result in a classification of the image. During training of the network, the error of the result of this classification is backpropagated through the network to update the weights of the parameters. This model was said to more strongly reflect the way our brains process information, as neuroscientists have observed similar filters being learned by neurons in our visual cortex.

Although convolutional neural networks have been revolutionary in computer vision for many tasks, they also have important limitations and drawbacks that hamper their efficacy and robustness. For example, the filters learned by the convolutional layers care little about spatial placement of features; thus, as long as the image has the right "parts", it could be misclassified. In addition, although it does better than older models with different poses, lighting conditions, and other variations in objects, general convolutional neural networks are still highly sensitive to these conditions; in general, their architectures must be specialized in order to perform well on these tasks. Finally, it is unclear why the max pooling layer, which introduces spatial position information back into the model, helps the network perform much better, since it also forces the network to lose much information in the process.

These drawbacks of convolutional neural networks have inspired the recent work of Geoffery Hinton, a leading figure in the world of deep learning, which centers around a new type of deep learning architecture. This architecture, called capsules or the capsule network, combines intuition about how the visual system parallelizes visual information processing to route information to the correct areas of the cortex with more traditional convolutional neural network structures. It also tries to imitate specifically how we represent geometric object in our imaginations: upon seeing new objects, we deconstruct them into hierarchical representations which we can then relate to our representations of learned objects.

2. ARCHITECTURE

The architecture of the network is comprised of several main points, which we will describe briefly here. As described in Sabour et al. (2017), in general, the image is passed into a traditional convolutional layer with a ReLU activation function. The outputs of this convolutional layer are then fed into a set of capsule layers called Primary Capsules, which are then routed to the classifier capsule layer. The weights between the Primary Capsules and the classifier capsules are then dynamically routed so that the connections between the Primary Capsule and the classifier capsule that it has the strongest association with. These weights are iteratively updated per training batch. The outputs of these classifier capsules are then passed into a decoder made up of fully connected layers. The loss from the reconstructions made from the output of this layer is combined with the margin loss computed from the outputs of the classifier capsules to yield the total loss that is backpropagated throughout the network.

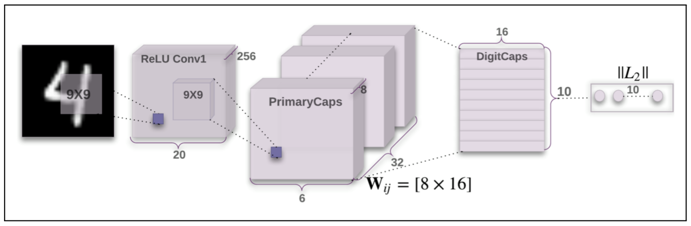

Fig. 1. Overview of the Capsule Network architecture [Sabour et al. 2017].

Another major difference between other neural networks and the capsule network is that the values of the neurons of the capsules are represented as vectors, where the length of the vectors represents the strength of the feature while the orientation of the vector encodes the orientation of the feature. In addition, instead of a ReLU or sigmoid activation, the activation function is a squash function, which is a novel way to introduce vector-to-vector non-linearity by squashing the response vector magnitude to one, while maintaining the direction of the vector.

3. METHODS

We decided to implement the capsule network in Python 2.7 with the Python deep learning package PyTorch. In general, we aimed to keep the network the same for each dataset in order to be able to compare their performances among each other. When augmentation was used, the following parameters were used: Probability of augmentation is 0.7. If the image is chosen to be augmented the scale is between 0.7 and 1.3, translation in the x and y coordinates is between -3 and 3 pixels, rotation is between -30 and 30 degrees. All these are uniformly distributed. Additionally left right flip was done for CIFAR dataset with probability of 0.35. For all datasets, we used the Adam optimizer. The training and testing of these models were run on NVIDIA K80 GPUs provided by Google Cloud. To decide the other parameters such as initial learning rates and number of epochs, we looked at other initial implementations. We did, however, make the following changes per dataset:

A. MNIST Dataset

Because this dataset was the one used in the paper, we wanted to replicate the network as closely as possible with the network described there. Thus, we made no changes to the architecture for MNIST. We did optionally include augmentation, which 70% of the time (randomly) included some combination of scaling, rotation, and translation. We also always normalized each image, which is important for computing the reconstruction loss as one input is the images themselves; if the images are not normalized, this loss will be orders of magnitude greater than the margin loss, thus completely overtaking it and allowing it very little contribution to the overall loss. Thus, the model will train on only the reconstruction loss, which is not very effective. We began with a learning rate of 0.01 and 30 epochs, as MNIST is a relatively easy classification task.

B. Fashion MNIST Dataset

These images of different classes of clothing/shoes (e.g. shirts, pants, dresses) represent a more difficult classification task than MNIST. Because these images have the exact same dimensions as MNIST and also had 10 classes, we kept the network the same throughout. We did not include augmentation as the images are all very well aligned. We did include the normalization, again to make sure the margin loss does contribute to the overall loss.

C. CIFAR 10 Dataset

While these images are also only classified into 10 classes, these images are larger in dimension, thus the initial convolutional layers were changed to accommodate this change in dimension. Due to early runs which gave low accuracies, we also decided to expand the Primary Capsule layer. Because the Primary Capsule layer is thought to train to produce representations of low level features, we expanded this layer to include 36 capsules instead of 16 capsules in order to include more low level features in its representation of these classes because they are more complicated. We also did not include the reconstruction loss in the total loss as this loss requires highly specific reconstructions of the data with neutral backgrounds. While MNIST and Fashion MNIST both have black backgrounds, the background of CIFAR-10 is highly varied. We also tried switching MSE (Mean Squared Error) loss for BCE (Binary Cross-Entropy Loss) for the margin loss term. We trained the model with either no augmentation or augmentation, as described above.

D. CIFAR 100 Dataset

Similar to CIFAR-10, these images are more complicated and represent a harder classification task than MNIST and Fashion MNIST. CIFAR-100 also has few examples per class as compared to CIFAR-10 which makes it a harder task. However, due to memory constraints of our Google Cloud Virtual Machines, we had to reduce the number of Primary Capsule units from 16 to 8. We also tried both MSE and BCE loss, as well as augmentation vs. no augmentation. Since this dataset was the first to have more than 10 classes, we also needed to expand the classifier capsule layer from 10 to 100 units.

4. RESULTS

For evaluation of the models, the total loss value was saved every two iterations through the network. To evaluate the efficacy of the neural network, ROC curves were used. The combined ROC curve is obtained by doing micro averaging. For MNIST, Fashion MNIST, and CIFAR-10 ROC curves for each class were also plotted to inspect if there was difference in the difficulty for classification of classes.

A. MNIST

MNIST showed near state-of-the-art performance (99.55% train accuracy/99.343% test accuracy) with 20 epochs, a learning rate of 0.01, and augmentation. Sabour et al. (2017) reported a slightly higher test accuracy of 99.75% in their implementation of the Capsule Network. Comparing to the state-of-the-art convolutional neural network, these methods show very comparable performance, as the top performing MNIST network(1) resulted in a test accuracy of 99.79%. We found that decreasing the learning rate to 0.001 and increasing the number of epochs to 40 and removing the augmentation parameter showed a very slight decrease in performance (99.990% train accuracy/99.234% test accuracy). We can see from these results that augmentation is necessary to prevent overfitting of the data, as the training accuracy is extremely high. We also saw in the reconstructions that the MNIST reconstructions with augmentation showed a more general representation of the digits. These figures are available in the supplementary data.

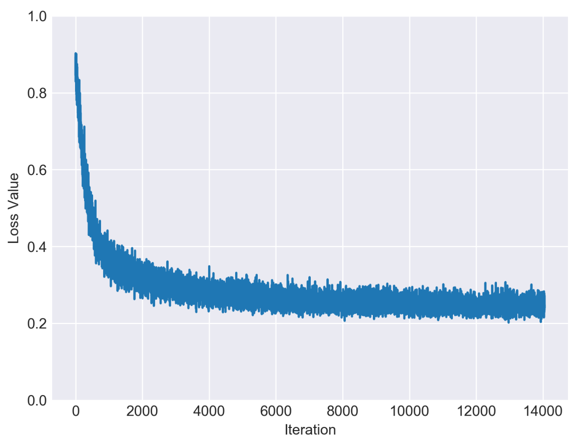

Fig. 2. MNIST Loss (30 Epochs, Learning Rate: 1E-2, Augmentation, MSE Loss)

B. Fashion MNIST

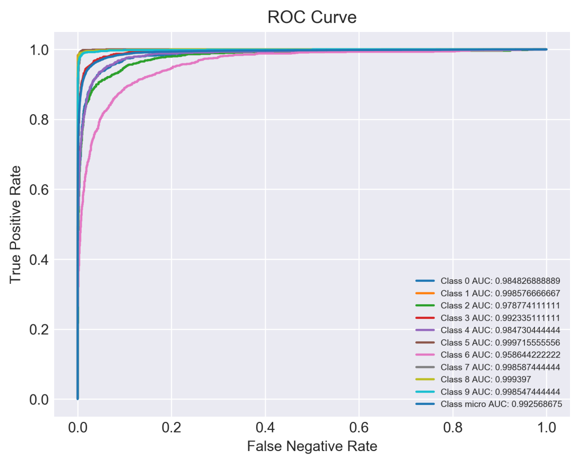

Fig. 3. ROC Curve For Fashion MNIST(Epochs 30, Learning Rate: 1E-2, MSE Loss Function)

Fashion MNIST also showed very strong performance with train/test accuracies of 99.59%/90.54% respectively after 30 epochs and a learning rate of 0.01. This model performed slightly better than a model that was trained over 40 epochs and a learning rate of 0.001, which is surprising due to the increased complexity of the Fashion MNIST

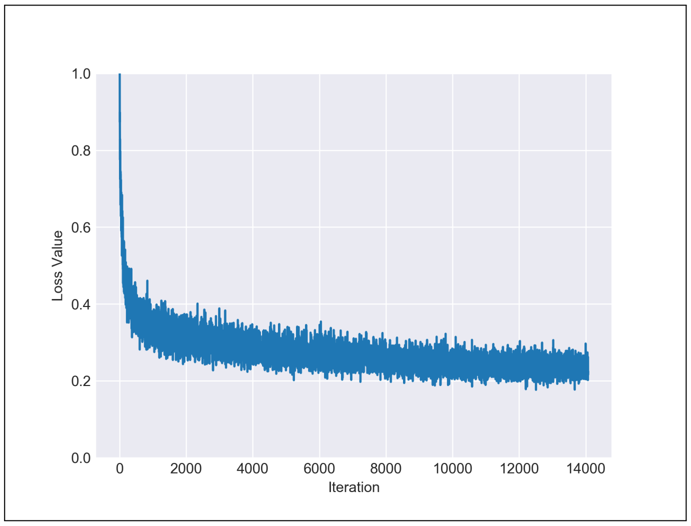

Fig. 4. Fashion MNIST Loss (30 Epochs, Learning Rate: 1E-2, MSE Loss)

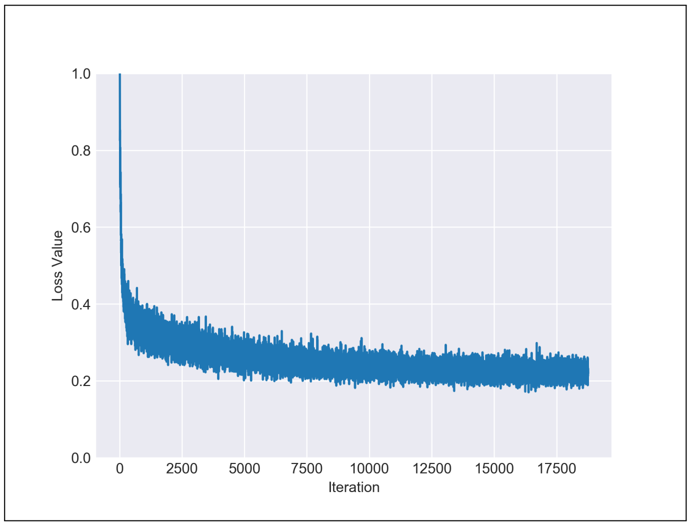

Fig. 5. Fashion MNIST Loss (40 Epochs, Learning Rate: 1E-3, MSE Loss)

dataset compared to the MNIST dataset, which would imply that a lower learning rate and more epochs would be needed. Although there is no paper yet on classifying Fashion MNIST with the capsule network, we found a implementation on Github which reports a test accuracy of 93.62%(5). We did perform better than another capsule network implementation on the website Kaggle(6), which reported a test accuracy of 76.8%. Our results are still below that of the highest performing convolutional neural network, which had a test accuracy of 96.9%(7). In the reconstructions, we can see that the internal representations of the final capsule layer removes much of the detail of the clothing, especially the sandals. These figures are available in the supplementary data.

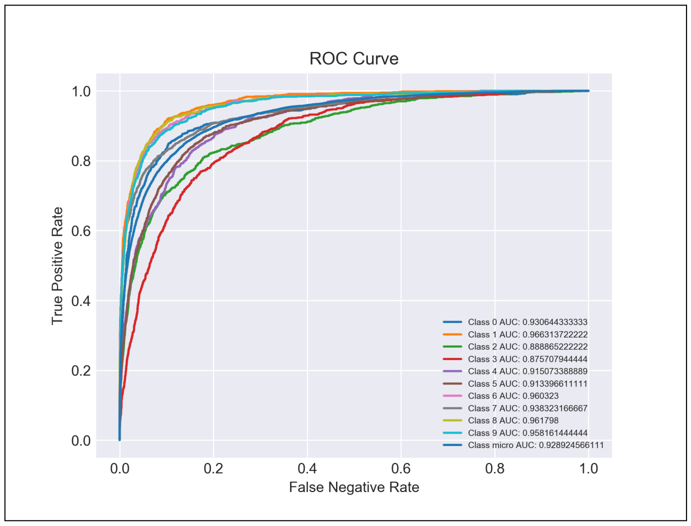

Fig. 6. ROC Curve For CIFAR-10(20 Epochs, BCE Loss Function, Learning Rate: 1E-4)

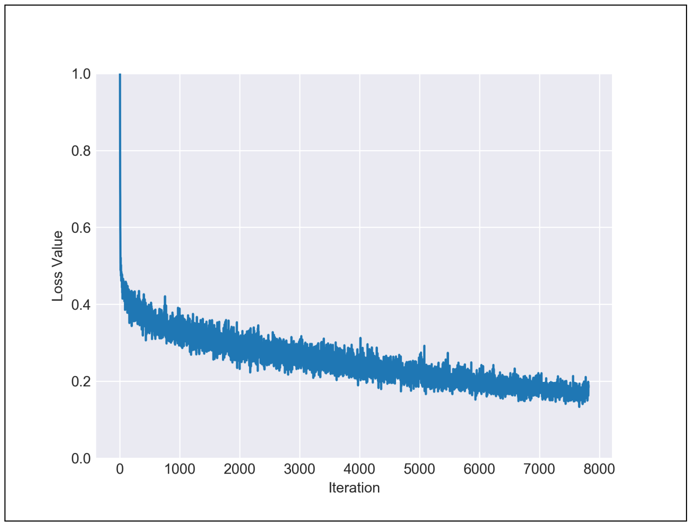

Fig. 7. CIFAR 10 Loss (20 Epochs, Learning Rate: 1E-4, MSE Loss)

C. CIFAR-10

With train/test accuracies of 93.70%/65.36%, CIFAR-10 showed lower accuracy as compared to Fashion MNIST, due to the dataset having more inputs along with a much harder classification. In comparison, Sabour et al. (2017) reported a test accuracy of 89.40%, although they do not specify their specific model for this dataset. The loss curve for BCE loss with augmentation show convergence around 15 epochs. However, without augmentation the loss is still going down after 20 epochs. The accuracy for without augmentation was also higher both on the test set and the training set by large amount. This might be due to the augmentation parameters being too stringent. The ROC curves for CIFAR-10 showed wide distribution of the ROC curves for individual classes. Class 3 corresponds to birds which consistently showed the worst ROC curve

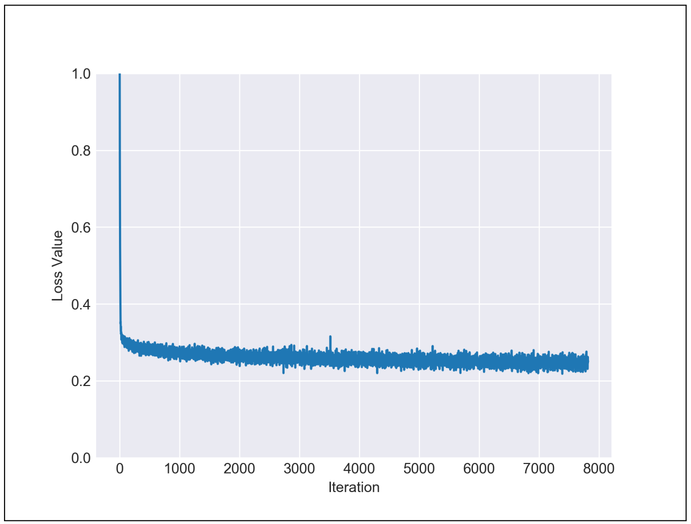

Fig. 8. CIFAR 10 Loss (20 Epochs, Learning Rate: 1E-4, MSE Loss)

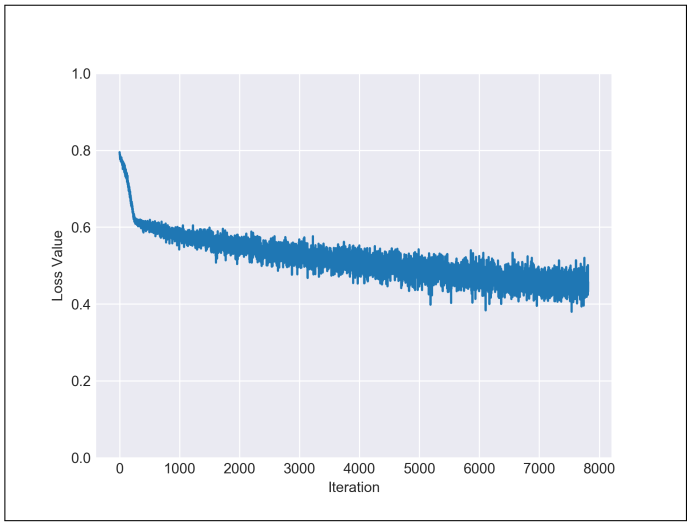

Fig. 10. CIFAR 100 Loss (10 Epochs, Learning Rate: 1E-4, MSE Loss)

across neural networks trained with loss function and with/without augmentation.

D. CIFAR-100

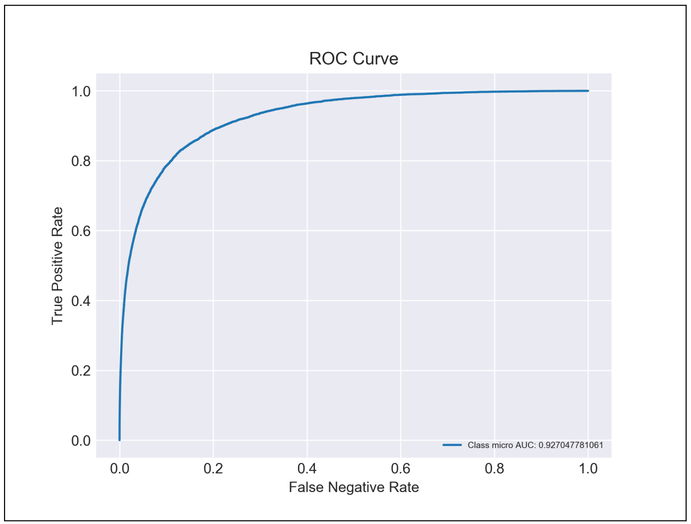

Fig. 9. ROC Curve For CIFAR-100 (10 Epochs, MSE Loss Function, Learning Rate:1E-4)

CIFAR-100 had the lowest accuracy; this might be due to variety of factors. The total number of samples available are much less, as such the network requires further optimization or more epochs would be required to increase the accuracy. This trend can be seen with the contінous near linear decrease in loss of CIFAR-100 neural network, both for MSE and BCE loss functions. From the graphs it appears that more epochs would have increased the accuracy for the neural network. Each epoch for CIFAR-100 took approximately 1 hours, as such it was unfeasible to run it for longer epochs.

5. DISCUSSION

In general, we found our implementation of the Capsule Network performed well for the simpler datasets, MNIST and Fashion MNIST, while we still need to make more updates and increase the number of the epochs to achieve better results for the more complicated datasets, CIFAR-10 and CIFAR-100. The lowest error obtained for MNIST was 0.657 which is near to the current lowest error of 0.21%(1). For Fashion MNIST the lowest error we achieved was 9.46% which is slightly higher than 6.38%(5) using capsule networks. We hypothesize that adding additional Primary Capsule layers for Fashion MNIST and/or increasing the number of epochs might increase the accuracy. Best accuracy for CIFAR 10 was 65.36%, which is much lower than 96.53%(2) . Similarly for CIFAR-100 the best accuracy of 35.29% is a lot lower than the 75.72%(3). The neural net architecture needs to be optimized for CIFAR-10 and CIFAR-100 for better classification. More epochs and better augmentation parameters would also help to achieve better classification. This was not possible for us to do given the limitation posed by long epoch times and capped google cloud GPU credits. Other optimizers such as Adadelta can also be used to remove hyperparameter tuning required for Adam optimizer(4). While conceptualizing the project we thought of using it on CARS dataset, we initially started to scale up from MNIST to CIFAR-10 to CIFAR-100. During this process we ran out of credits. We also aimed to explore how well capsule network can take advantage of hierarchy, but this would have required many more Google Cloud credits, as the models for CIFAR-100 took more than 10 hours to run for 10 epochs.

In this paper, we introduce an implementation of the novel Capsule Network that has been modified to accommodate and test multiple common datasets for deep learning in order to compare their performance among datasets as well as with other models and other capsule network implementations, such as Sabour et al. (2017). Because the Capsule Network is reported to have advantages over traditional convolutional neural networks as it uses an inverse graphics approach to represent hierarchies of features and images, we wanted to expand the testing of the Capsule Network in order to see where the strengths and weaknesses of the network lie in practice. We found that while our implementation of the Capsule Network does work well for simpler datasets like MNIST and Fashion MNIST and performs comparably to other capsule network and convolutional neural network implementations, the models for CIFAR-10 and CIFAR-100 in general fall short of the results produced by convolutional neural networks (as very few to no capsule network implementations have been reported). This is likely because the number of units present in our implementation of the Capsule Network was not enough to represent all of the variation of the objects in these datasets. Thus, we find that Capsule Networks do need large modifications from the original network in order to perform well on more complex datasets. Oftentimes, these modifications may result in a much slower training phase of the model, which is major drawback of Capsule Networks compared to convolutional neural networks. It would also be interesting to capture the representations of each Primary Capsule to view their internal representations of the objects and use that information to inform our final number of Primary Capsules.

6. ACKNOWLEDGEMENTS

We'd like to thank Google Cloud for provide us with GPU credits for NVIDIA K80 GPUs. These were integral to us being able to run these models.

A. Supplementary Data

Table 1. Best Test Accuracy Comparisons

Dataset	Ours	Best Capsule Network
MNIST	99.34	99.75
Fashion MNIST	90.54	93.62
CIFAR-10	65.36	89.60
CIFAR-100	35.29	N/A

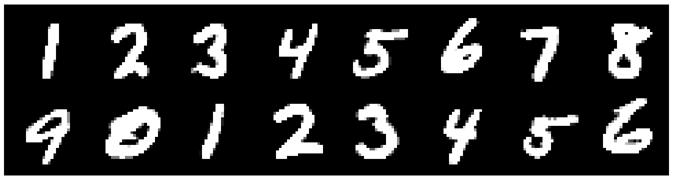

Fig. 11. MNIST True Image (10 Epochs, Learning Rate: 1E-4, MSE Loss)

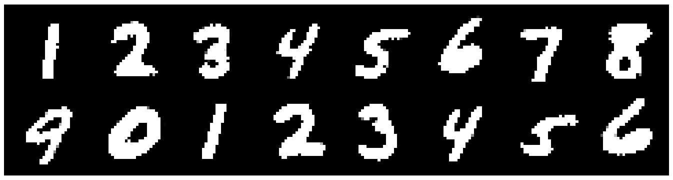

Fig. 12. MNIST Reconstruction Image (30 Epochs, Learning Rate: 1E-2, MSE Loss)

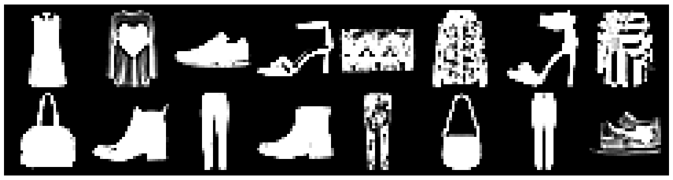

Fig. 13. Fashion MNIST True Image (10 Epochs, Learning Rate: 1E-4, MSE Loss)

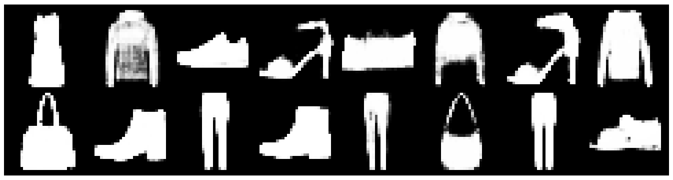

Fig. 14. Fashion MNIST Reconstruction Image (30 Epochs, Learning Rate: 1E-3, MSE Loss)

7. REFERENCES

1. Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., & Fergus, R. (2013, February). Regularization of neural networks using dropconnect. In International Conference on Machine Learning (pp. 1058-1066).
2. Graham, B. (2014). Fractional max-pooling. arXiv preprint arXiv:1412.6071.
3. Clevert, D. A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289.
4. Zeiler, M. D. (2012). ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.
5. Xifeng Guo. "CapsNet Fashion MNIST". https://github.com/XifengGuo/CapsNet-Fashion-MNIST
6. Kevin Mader. "Capsule Network on Fashion MNIST". https://www.kaggle.com/kmader/capsulenet-on-fashion-mnist
7. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. Han Xiao, Kashif Rasul, Roland Vollgraf. arXiv:1708.07747
8. Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.
9. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.