# Final Report

Mohamed Matar, Dingqing Yang

April 6th, 2018

**Abstract**

Capsule network is a new proposed neural network as an enhancement to Convlutional Neural Network(CNN) and potentially could replace CNNs in different application that require more accurate detection of objects that tends to change in terms of graphical properties such as position, orientation and thickness. In this work, we study the architecture of a capsule network, demonstrate its bottlenecks by profiling its operation and finally we propose potential solutions that can take advantage of patterns we have observed that can be implemented as a hardware accelerator.

## 1 Motivation

CNNs have been the standard method for computer vision. In general, CNNs work by detecting the presence of features in an image, and using the knowledge of these features to predict whether an object exists. However, if we look at figure 1 ,a CNN would classify both images as faces because they both contain the elements of faces.
The reason for CNNs failure to deal with such images is the way CNNs pass information from layer to layer which is max pooling. Max pooling examines a grid of pixels, and takes the maximum activation in that region. This procedure essentially detects whether a feature is present in any region of the image, but loses spatial information about the feature.
Capsule networks attempt to address the limitations of CNNs by capturing part-whole relationship. For example, in a face, each of the two eyes is part of a forehead. In order to understand these part-whole relationship, capsule networks need to know more than just whether a feature exists, it needs to know the feature location, orientation, and other pose information. Capsule networks succeed in ways that CNNs have not because they can successfully capture this information from parts to their wholes.

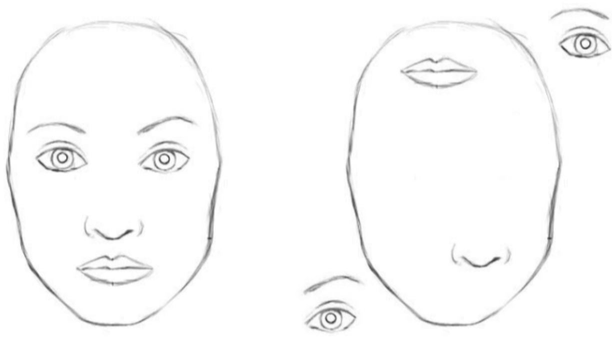

Figure 1: False positive example for CNN inference

## 2 Routing Algorithm: EM for GMMs

Deciding on activating high layer capsules given the low layer capsule is done using expectation maximization algorithm for Gaussian mixture models, in which, low layer capsules are treated as data-points and high layer capsules are the Gaussian models that these data-points should belong to. In order to visualize this, we can take a look at figure 2, the EM tries to see if each data-point *(i)* is a good fit for a model *(j)*, so we can see that the red data-point (let's call it capsule (M) in layer L) is a good fit for Gaussian model 1 (first on the left, parent capsule (N) in layer L+1) hence, capsule (M) will vote strongly for capsule (N). On the other hand, capsule (M) will not vote for the second Gaussian model since it doesn't fit for its model. This operation is done iteratively until data-points vote strongly for their models and lower votes are pruned gradually until they disappear.

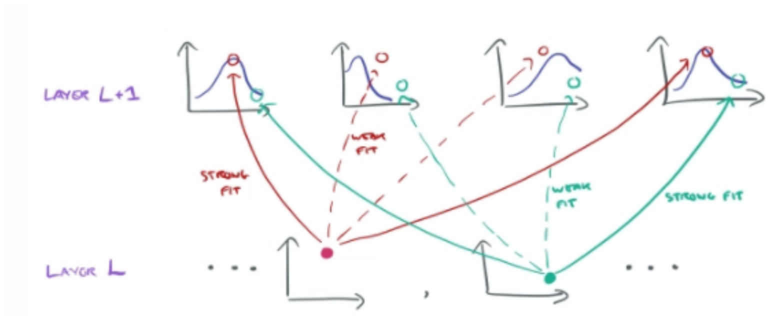

Figure 2: Visualizing Expectation Maximization iterative operation

## 3 Evaluation

In this section, we show several properties that we observed during analyzing the capsule network architecture. We will talk about each of them as follow:

### 3.1 Activation Sparsity

The first observation we have seen is that activations tends to have a sparse pattern after each layer in the network. In figure3, we show activations of smallNORB dataset, and we can see that activations tend to be more sparse as we go deeper in the network. This property shows a potential of skipping a lot of operations based on the zero-valued activations. This behavior looks similar to CNN activation behavior and the authors of [AJH+16] proposed this solution on CNN accelerator.

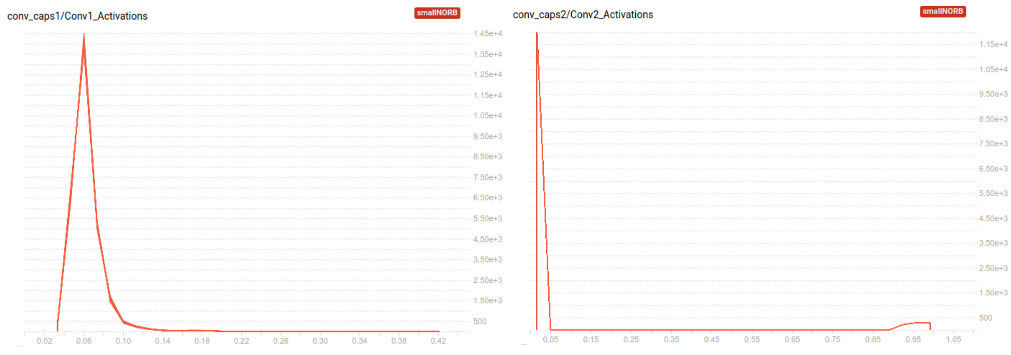

Figure 3: Activations histogram of two caps convlutional layers

### 3.2 Pose Distribution

Another interesting property about capsule is its pose matrix. In essence, the pose matrix is a description matrix for the object in terms of graphical properties. During inference of capsule networks, we analyze the pose matrix of each layer capsules and as seen in figure 4, the pose tends to have a specific pattern of a certain distribution. Also, it tends to stretch with datasets that have orientation nature such as smallNORB.

A good way to make use of this distribution is non-linear quantization, in this way frequent values can have high resolution quantization and less frequent values will have lower resolution which will eventually save space and computation time, as we will see in figure 5 most of the memory is consumed in the first layers since it has more capsules hence more pose matrices.

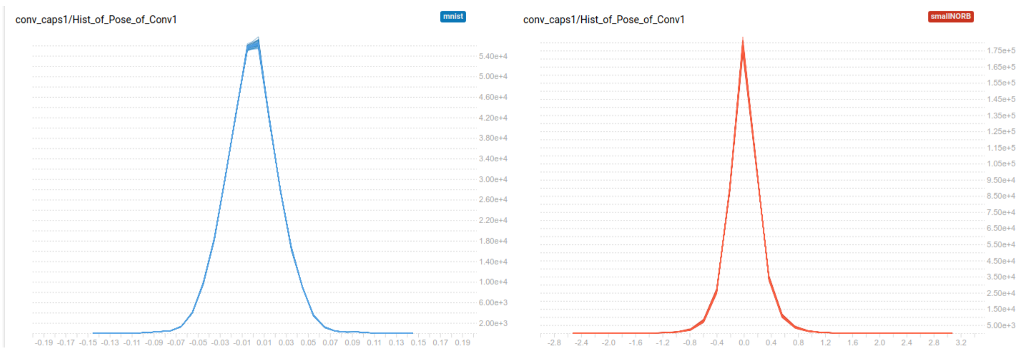

Figure 4: a) Pose matrix histogram of SmallNORB b) Pose matrix histogram of MNIST

## 3.3 Capsule Network Computational Graph

Based on the network architecture proposed in [GEH18], we implemented this architecture using tensorflow [AAB$^+$15] framework.In figure 5, we present the breakdown of memory utilization and execution time for each layer in the capsule network architecture. We can observe that memory utilization is maximized in the first capsule convlutional layer which contains most of the biggest number of capsules, hence, a lot of pose matrices are stored in this stage. In 5b) we can see that execution time get increased as we go deeper towards the output layer which behaves similarly to CNN network. Based on these observations, we will motivate our optimizations to the network operations.

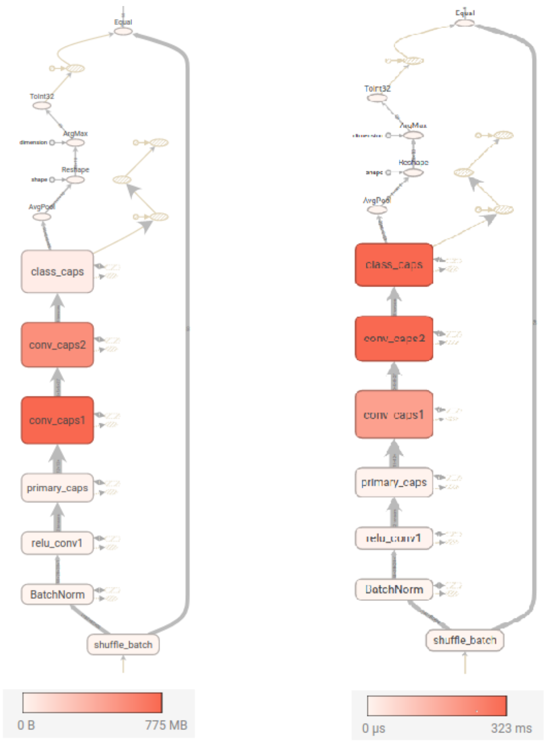

Figure 5: a) Memory utilization for each layer in the network b) Time consumption for each layer

## 3.4 Evaluation Accuracy

In figure 6, we report test accuracy for different datasets trained during this experiment. The figure shows high accuracy for both datasets: MNIST and Fashion-MNIST, smallNORB dataset is less accurate since its hyper-parameters are different than the first two datasets and extra study is needed to identify these parameters.

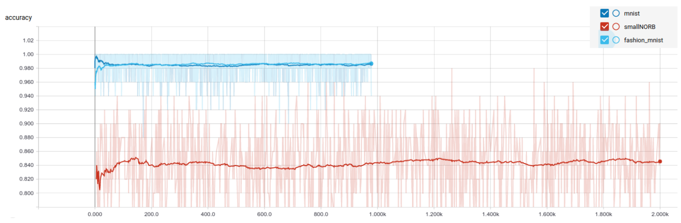

Figure 6: Test Accuracy for different datasets trained

## References

[AAB$^+$15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[AJH$^+$16] J. Albericio, P. Judd, T. Hetherington, T. Aamodt, N. E. Jerger, and A. Moshovos. Cnvlutin: Ineffectual-neuron-free deep neural network computing. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pages 1–13, June 2016.

[GEH18] Nicholas Frosst Geoffrey E Hinton, Sara Sabour. Matrix capsules with EM routing. *International Conference on Learning Representations*, 2018.