CapsuleNet vs CNN : A Performance Based Study

CS 725: Course Project

by

A.V.S Bharadwaj 193050010
Pankaj Kumar 193050065
Raushan Raj 193050073
Kaushik Ganorkar 193050078

under the guidance of

Prof.Sunita Sarawagi

Department of
Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai 400 076

Contents

1 Project Objective

Geoffrey Hinton , one of the pioneers of Deep Learning has come up with a new architecture for computer Vision called as Capsule Networks which are based on the fundamental units called Capsules.This latest architecture possesses the quality of EQUIVARIANCE which is not present in CNNs and utilises this to its benefit. This ground-breaking architecture claims to outperform the traditional CNN based architectures on many standard datasets . We would like to test this hypothesis by implementing the original Capsule Network as given by Hinton along with a CNN architecture and compare their results on the two standard datasets of MNIST and CIFAR10.

CNNs can handle translational invariance but not rotational invariance.Layers need to communicate with each other – Maxpooling layers works like a messenger between two layers of a CNN and transfers the activation information from one layer to the next layer. It tells the layers about the presence of a part, but not the spatial relation between the parts. The MaxPooling layer strips off this information to create translational invariance – the ability of a network to detect an object even wherever it lies in the image.

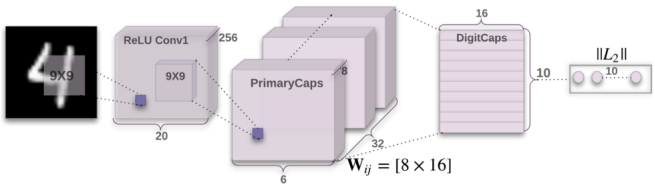
Capsule Networks use Dynamic Routing between Capsules in layers to pass on the information from one layer to the next layer (in place of MaxPooling layer in CNN) Also, Maxpooling does not provide 'ViewPoint Invariance" – the ability to make the model invariant to changes in viewpoint. Capsule Networks outperform everything else when it comes to problems involving viewpoint invariance.

The main goal of this project is to test this hypothesis and compare the results obtained in proctored settings for CNN as well as for CapsuleNet

2 Litreture Survey

Geoffrey Hinton and his team published two papers that introduced a completely new type of neural network based on so-called capsules. In addition to that, the team published an algorithm, called dynamic routing between capsules, that allows to train such a network.(1)
Capsules are a new concept which can contains more information about each "object". Capsules are a vector (an element with size and direction) specifying the features of the object and its likelihood. These features can be any of the instantiation parameters like "pose" (position, size, orientation), deformation, velocity, albedo (light reflection), hue, texture, etc(2)
CNN's detect features in images and learn how to recognize objects with this information. Layers near the start detecting really simple features like edges and layers that are deeper can detect more complex features like eyes, noses, or an entire face. It then uses all of these features which it has learned, to make a final prediction. Herein lies the flaws of this system — there is no spatial information that is used anywhere in a CNN and the pooling function that is used to connect layers, is really really inefficient.(1) In the process of max pooling, lots of important information is lost because only the most active neurons are chosen to be moved to the next layer. This operation is the reason that valuable spatial information gets lost between the layers. To solve this issue, Hinton proposed that we use a process called "routing-by-agreement". This means that lower level features will only get sent to a higher level layer that matches its contents. If the features it contains resemble that of an eye or a mouth, it will get to a "face" or if it contains fingers and a palm, it will get send to "hand".(4) Information about properly trained features gathered in one position can be spread out to other positions;this functionality has become advantageous for image interpretation. In contrast, CapsNet replaces the scalar-output feature detectors from CNNs with vector-outputs, it also replaces the max-pooling sub-sampling technique with routing-by-agreement, so it enables the duplication of learned knowledge across space. In this new CapsNet architecture, only the first layer of cap-sules, also known as primary capsules, includes groups of convolutional layers. Fol-lowing traditional CNN ideas, higher-level capsules cover more extensive regions of the image. However, the information about the precise position of the entity within the region is preserved in contrary to standard CNNs. For lower-level capsules, the location information is "place-coded" by the active capsule. As we ascend in the hierarchy, more and more of the positional information is "rate-coded" in the real-valued components of the output vector of a capsule. All these ideas

imply that the dimensionality of capsules must increase as we move up in the hierarchy. To summarize, the Capsule Networks give us the opportunity to take full advantage of intrinsic spatial relationships and model the ability to understand the changes in the image, and thus to better generalize what is perceived(3)
Hilton et al also described an enhanced version of Capsule Networks where each capsule is described using a logistic unit to represent the presence of the entity and a matrix to describe the properties of the entity (pose matrix), which is a 4x4 matrix to represent different characteristics like spatial coordinates, orientation of the object and other characteristics of a feature. The connection between the lower capsule and parent capsule is a transformation matrix (also 4x4). They use the Expectation Maximization (EM) routing to group appropriate capsules in a lower layer to the higher layer to form a part-whole relationship(5) .



3 Approaches:

As this is a study based course project our team first built an CNN model for obtaining baseline values for two different datasets. We started off with CIFAR-10 dataset. We applied the concepts taught in the class and developed an CNN model having different hyperparameters. We were stuck on deciding the striding and padding sequences and the convolutional layers were also checked for higher accuracy .Then we worked on the pooling layer, its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. Pooling layer operates on each feature map independently. The most common approach used in pooling is max pooling. The convolutional and pooling layers are followed by a dense fully connected layer that interprets the features extracted by the convolutional part of the model. A flatten layer is used between the convolutional layers and the dense layer to reduce the feature maps to a single one-dimensional vector.
a) CNN Architecture:: For baseline CNN architecture,we use 2 layer convolution model of 32, 64 channels. First channel has 55kernel and stride of 1. The last convolution layer is followed by one fully connected layer which maps the kernel to the output class probabilities.

b) CapsuleNet-(Dynamic Routing):: For Dynamic Routing-based CapsNet architecture , we use a the architecture followed by the original paper. It consists of 2 convolution layers and one fully connected layer. The first convolution acts like a initial feature extractor and are passed as inputs to the primary capsules. The first convolution layer has 99 convolution kernels with a stride of 1 followed by ReLU activation. The primary capsules is a 32 channels of 8D convolution capsules and generate [3266] capsule output vectors of 8 dimensions. Dynamic Routing mechanism is introduced between the primary and secondary capsules. Each capsule vector of the lower layer is connected to the capsules on the higher layer. The final Digit Capslayer is mapped to the 16D capsule per class and trained on reconstruction loss and l2 logits loss

4 Experiments:

4.1 Code Description :

4.1.1 CNN:

We used python and different libraries such as Tensorflow,Keras,Matplotlib etc. Total lines of code is 198. We added 7 convolution layers . The input is (32,32) vector .The activation functions used are Relu in the convolutional layers and did a batch normalisation and max pooling and also used Dropout to prevent overfitting. Then we trained the CIFAR-10 and MNIST CNN models.Then we plot the summary and generate the comparisions.
We used this reference to get started
:https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/
:https://towardsdatascience.com/cifar-10-image-classification-in-tensorflow-5b501f7dc77c

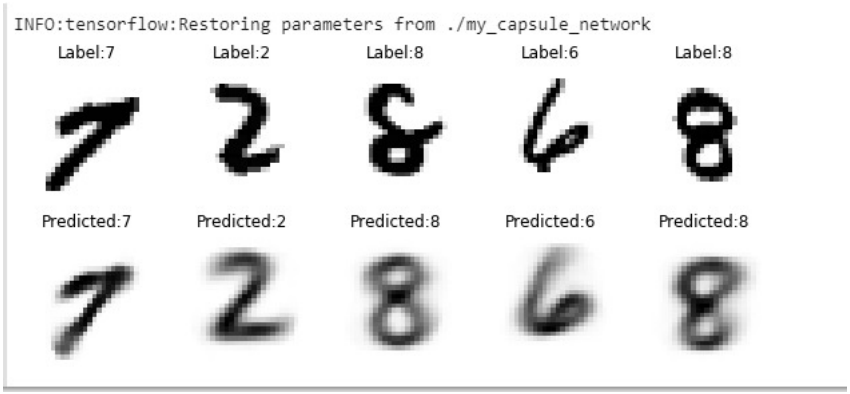
4.1.2 CapsuleNet:

To implement the CapsNet model, the first layer is a standard convolutional,we took this design from Sabouret al.and used it in the same way for all our tests. This first convolutional layer produces 256 feature maps,using a 9x9 kernel and valid padding. For the convolution within the primary capsules we also kept a kernel size of 9x9, x1 and x2 entirely depend on the size of the input image. G1xG2 are the dimensions of each capsule, and these values are computed automatically based on x1 and x2. Other parameters we tuned for different datasets including D1 and D2(the dimensions of the output vectors in primary and routing capsules),F(the number of channels in the primary capsule layer), and C (the number of classes).To test different hyper parameter options we tested several batch sizes for CapsNet, from 10 to 50 to fit GPU RAM. Here we were able to only test the MNIST model , because of computational difficulties and the results were getting a bit ambiguous
We used this reference to get started
https://mc.ai/a-tutorial-for-implementing-capsulenet-using-tensorflow/

4.2 Resources

We did most of our initial computations using Pycharm IDE , but it was a bit slow , so we shifted to google colab workspace (Python workspace on cloud).using Pycharm, it was taking 25 mins for one epoch.

4.3 Results:



5 Efforts:

5.1 Project Estimation:

We spent most of the time researching about the CapsuleNets.Then considerable amount of time was spent on development of CNN models. The most critical point of the project was the implementation of CapsuleNet.Different challenges were encountered.

5.2 Challenges:

During this course of our implementation of the models,we encountered multiple issues:

• **Sensitivity to hyperparameters** : Hinton have not released their official code base for Matrix Capsules. Any open-source implementation we tried and implemented ourselves suffered from severe hyper-parameter sensitivity. The sensitivity affected the convergence and overall accuracy of the model. We suspect this might be due to some underlying design decisions which were not mentioned in their paper

• **Computationally expensive** : CapsNet and Matrix Capsules are quite computationally expensive and slow.We were forced to curtail the number of epochs for

convergence

5.3 Work Allocation:

The team worked in two groups. A.V.S Bharadwaj and Raushan Raj worked on scaling and understanding the CapsuleNets and Kaushik Ganorkar and Pankaj Kumar worked on implementing the CNNs and comparisions.So the work was evenly split at the onset of the project.

References

[1] Tomer Eldor *Capsule Neural Networks – Part 2: What is a Capsule?* . https://towardsdatascience.com/capsule-neural-networks-part-2-what-is-a-capsule-

[2] Ashutosh Kumar *Image recognition by Neural Networks.* . https://analyticsindiamag.com/why-do-capsule-networks-work-better-than-convolutio

[3] Rinat Mukhometzianov and Juan Carrillo *CapsNet comparative performance evaluation for image classification* https://arxiv.org/pdf/1805.11195.pdf

[4] Aryan Mishra *Capsule Networks: The New Deep Learning Network.* . https://towardsdatascience.com/capsule-networks-the-new-deep-learning-network-bd9

[5] Sara Sabour, Nicholas Frosst, Geoffrey Hinton *Dynamic Routing Between Capsules* https://arxiv.org/pdf/1710.09829.pdf