Rotation and Scale Equi-variance for MNIST Image Dataset

A Project Report

submitted by

S NARAYANA KRISHNAN, EE14B124

in partial fulfilment of requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

JUNE 2018

THESIS CERTIFICATE

This is to certify that the thesis titled **Rotation and Scale Equi-variance for MNIST Image Dataset**, submitted by **S Narayana Krishnan, EE14B124**,to the Indian Institute of Technology Madras, for the award of the degree of **Bachelor of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Aravind R
Project Guide
Professor
Dept. of Electrical Engineering
IIT Madras, 600036

Place: Chennai

Date: 12th June 2018

ABSTRACT

KEYWORDS: Computer Vision, Deep Learning, Machine Learning, Capsule Nets, equi-variance, Neural Networks,CNN

In-variance to view points and affine transformations is one of the most sought after feature in computer vision. General affine transformations like scaling, illumination, rotation and translations should not come in the way of image classification/recognition. Convolutional Neural Networks or CNN's have become the established architecture for computer vision as they are able to provide translational equi-variance. They are able to learn features from the image which were up until then were hand-engineered. But CNN's are not very efficient in generating features which are equi-variant to general transformations other than translation.

In this project an attempt was made to understand and improve upon two existing solutions for achieving scale and rotational equi-variance on MNIST Dataset, Harmonic Networks and Capsule Networks. Harmonic Networks strive to achieve rotation equi-variance by replacing regular CNN filters with rotational harmonics. The filters in Harmonic Nets were further tuned in an attempt to induce equi-variance to scale.

The ideas proposed for use of Capsule Nets for MNIST classification were applied for rotated and scaled MNIST digits to observe the performance and an altercation was made to improve the performance and also to encourage the Capsule Net to learn transformation equi-variant representations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Although CNN's have become the go-to architecture for computer vision problems, they still face several short comings. They are unable to learn equi-variant features when the images are scaled or rotated. Many existing methods which employ CNN's either use data augmentation or hand designed image pre-processing methods to achieve rotation and scale equi-variance. These methods are unable to provide elegant and general solutions to the problem at hand. The work done in this project is mainly on two separate approaches with aim achieve generalized solutions to the above stated problems.

In the first part of the project the concept of harmonic networks was deeply studied and implemented. The paper on Harmonic Nets is an attempt to provide rotational equi-variance to CNN's. The main idea in the paper is to restrict the filters in a CNN to a general class of radial harmonic functions. This ensures that the feature maps generated by HNets are equi-variant to rotation and therefore inherently induce a CNN to learn rotation invariant feature map representations. This method works very well and beats the state of the art in classification of rotated images. A similar approach derived from the same core idea was applied to this network to modify and to try and achieve equi-variance for scale. The filters of this network were further constrained by radial profile weight sharing. A considerable improvement in performance on rotated and scaled images was achieved.

In the second part the idea of Capsule Nets was explored. This paper provides a novel approach to the problem by representing information in the form of vectors rather than feature maps as in a regular CNN and applied dynamic routing in between capsule layers to achieve part to whole relationships between two consecutive layers of capsules. This architecture has now delivered state of the art performance on mnist classification. The network was then applied and tuned to classify rotated and scaled images to observe performance. The network managed to achieve high accuracy on scaled and rotated mnist images. This indicated that then vector representation of features may lead to

some sort equi-variance to rotation and scale and therefore further attempts were made to extract the values of rotation and scale parameters from the output of the Capsule Net in order to verify this claim.

1.1 Literature Review

Given below is the list of papers which are related to the problem statement , along with a brief gist on the core ideas proposed in the paper.

- Polar Transformer Networks (PTN's)[5] propose a 3 stage end to end architecture for classification. First, a polar origin predictor a CNN[1] that extracts feature maps which are used to find the centre of the image. Second stage performs the Log-polar transformation on the image and the final stage is a CNN classifier used to predict the image class. Due to the Log-polar image transformation, rotation and scaling of images are converted into translation, a transformation that CNN's are very good at identifying. But translations do not have an equi-variant representation in log polar domain which is the reason a separate origin predictor CNN module is used to deal with translations. Though this network is able to beat the state of the art on rotated MNIST dataset it is not scale-able and cannot make good predictions when multiple objects of interest are present in the same image.

- Spatial Transformer Networks(STN)[6] provide a new attachable spatial transformer module to a regular CNN which can be inserted at any layer. The transformer module generates multiple feature maps from the given feature maps by applying spatial transformations to them. When applied to the input layer of the network itself is able to learn to crop out, rotate and normalize the image which can lead to a simplified classification task for the CNN module, dramatically increasing its performance. This model also exhibits scale-ability able to provide competitive results on street view numbers and bird classification problems. The main drawbacks of this model is its requirement of significantly larger number of parameters.

- The paper Scale invariant pattern recognition[7] with logarithmic radial harmonic filters proposes filtering the image using a matched phase-only LRH(logarithmic radial harmonic) filter function and then use correlation at the origin in order to perform image classification.

- Circular harmonic phase filters[8] for efficient rotation-invariant pattern recognition also follows a very similar approach as used in the paper described above, using phase-only circular harmonics instead of LRH to achieve rotational equivariance. This paper along with the concept of steerable filters[3] forms the basic foundations on which the idea of Harmonic Networks is based.

- Harmonic Nets[9] as already discussed, places constrains on the CNN filters which cause it to generate feature maps which are equi-variant to rotation, this

paper produces the state of the art performance on rotated MMIST dataset, but only works for image rotations.

- Transforming auto encoders[2] lays the basic foundation on the idea of capsules which is later perfected in Capsule Nets. This paper highlights the main ideas about using vectorial representation of information instead of using feature maps as in traditional CNN's. It introduces ideas on part-whole relations between individual layers of capsules.It also published results on simple experiments to prove that the idea of using Capsules actually works. The experiments involve reconstructing transformed MNIST digits and 3D stereo images of different types of cars from outputs of the auto encoders.

- Finally the paper on Capsule Nets[4] discusses the idea about capsules, and the Dynamic routing algorithm to carry out the proposed part-to-whole transformations between adjacent capsule layers. This paper is able to beat the state of the art performance on MNIST data-set and at the same time able to encode transformation invariant features of the image in its vector output in sufficient detail to allow approximate reconstructions of the image. The video on Youtube by Aurélien Géron[1] also provides a deep and excellent explanation on the working of Capsule Nets.

1.2 Outline of the Problem

The problem statement of this project is to attempt to improve upon harmonic nets and classify the MNIST dataset which has been transformed through random 360° rotations(figure 1.2) and random scaling by a factor in range $[0.5, 2]$ (figure 1.3). For capsule nets the aim is to classify a much less rigorous dataset - MNIST transformed by random rotations through angles which are multiples $11.25°$. The images also scaled by 1 of 32 different scales in $[0.5, 1]$. Along with classification, an attempt is made to extract the rotation and scale parameters from the capsule network's output vectors.

1.2.1 MNIST Dataset

The MNIST database of handwritten digits[2] that has a training set of 55,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image. Each image has 28x28 pixel's which are converted into a single 784 dimensional input vector for training. A few sample MNIST images are shown in figure 1.1.

[1]Capsule Networks (CapsNets) - Tutorial,June 2018.
[2]MNIST Dataset

Figure 1.1: MNIST images

Figure 1.2: Rotated MNIST images

Figure 1.3: Rotated and scaled MNIST images

CHAPTER 2

Harmonic Nets

This chapter discusses about Harmonic Nets portion of the project.

2.1 equi-variance

Equi-variance is a useful property to have because transformations π of the input produce predictable transformations ψ of the features, which are interpretable and can make learning easier. Formally, we say that feature mapping $f : X \to Y$ is equi-variant to a group of transformations if we can associate every transformation $\pi \in \Pi$ of the input $x \in X$ with a transformation $\psi \in \Psi$ of the features; that is,

$$\psi[f(x)] = f(\pi[x]) \tag{2.1}$$

A special case of equi-variance is invariance, when $\Psi = I$, the identity.

2.1.1 Complex Circular Harmonics

Rather than having all the weights in the filter learnt as in a conventional CNN in harmonic nets the filters functions are restricted to a class of complex circular harmonics. The general filter equation is given by

$$W_m(r, \phi, R, \beta) = R(r)e^{i(m\phi+\beta)} \tag{2.2}$$

The learn-able parameters in this filter function is the radial profile $R(r)$ and the phase offset β. Such a filter function provides equi-variance under the operation of circular cross cross-correlation($*$), let the input feature be denoted as F then a version of this feature map rotated by θ is denoted by F^θ, then,

$$[W_m * F^\theta] = e^{im\theta}[W_m * F] \tag{2.3}$$

2.1.2 Equi-variance to Scaling

By using the same line of thought as in the above section,an attempt was made to come up with a general class of functions which provide equi-varaint feature maps on scaling. For this purpose one can look to the Fourier-Mellin transform, the magnitude of which is invariant to both rotation and scaling, it is formulated as,

$$M_f(u,v) = \frac{1}{2\pi}\int_0^{\infty}\int_0^{2\pi} f(r,\theta) r^{-iu-1} e^{-iv\theta} d\theta dr \tag{2.4}$$

Harmonic nets simply constraint their phase to be of the form $e^{im\theta}$ to achieve equi-variance to rotation. Following a similar line of thought one can constraint the radial profile to be of the form r^n, $n \in N$,resulting in the filter equation,

$$W_n(r,\phi,\Theta,\alpha) = \alpha r^n \Theta(\phi) \tag{2.5}$$

Such a filter output is equi-varaint with respect to scaling.

2.1.3 equi-variance to Rotation and Scaling

At this point one might be tempted to combine the ideas from both the above section and design a filter function as such,

$$W_{mz}(r,\phi,\alpha,\beta) = \alpha r^z e^{i(m\phi+\beta)} \tag{2.6}$$

Such a filter function indeed would output feature maps invariant to both rotation and scaling but the problem with such a filter function is that it has only 2 learn-able parameters α, β and therefore in order to fit the data set would require a huge number of feature maps, which is impractical.

There is a reduction in number of learn-able parameters per-filter from CNN to a HN and from HN to a network using filters as in 2.6. Therefore we need to make a trade-off between number of learn-able parameters per-filter and the amount of scale equi-variance.

We constraint the radial profile of the HN filters to have same values for pairs of adjacent rings, effectively stretching the profiles to provide a somewhat equi-variant

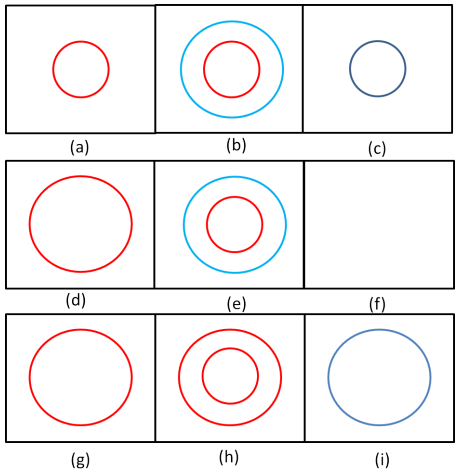

Figure 2.1: Input images,learned filters and outputs for a simple network designed to identify a circle.

response to feature maps even when they are scaled.The radial profiles are therefore *sharing* weights.

This idea is better explained though a simple example. Consider a simple network that is used to identify whether or not a given input image has a circle(figure 2.1(a)). We can expect the network to learn a filter as in figure 2.1(b). Assume that the network was trained only on un-scaled images,to such a network if an image scaled by a factor of 2 (figure 2.1(d)) is shown, its filter as in figure 2.1(e) would completely fail to recognize the larger circle.But to this filter if *radial weight sharing* was applied then its radial profile would appear something as in figure 2.1(h) and despite never having seen larger circle the filter would still output a high correlation for such an image(figure 2.1 (i)).

2.2 Implementation and Results

2.2.1 Test Networks

First, a shallow CNN was trained on the standard MNIST dataset. This CNN is only two layers deep and manages to get a classification accuracy of about 97%. This network is used as a bench-mark to test the performance of the harmonic nets.

In order to test the working of harmonic nets and fully understand its TensorFlow™ implementation, a shallow harmonic net also 2 layer with close to same number of parameters of the bench-mark CNN was implemented. When tested on the standard

MNIST dataset this model gave only 91% classification accuracy. In order to test if the harmonic net is able to learn rotation *"intuitively"* the same model was then tested with a test set containing random 360 ° rotations of the MNIST images, and the results are documented in the table 2.1.

Both networks perform poorly on the rotated test set. But the harmonic net is still able to outperform the bench-mark CNN, hence proving that the harmonic net is indeed a better architecture to learn image rotations.

Network	MNIST()	Rotated MNIST
CNN	0.9707	0.3552
HN	0.9115	0.4029

Table 2.1: Test results for MNIST and rotated MNIST on CNN and HN,values represent accuracies.

2.2.2 Rotated MNIST

In the next step the network mentioned in the paper with the same hyper-parameters was implemented. The network described in the paper was trained only on 10,000 rotated test images and was giving 100% test accuracy, implying that it was over-fitting the test-set. Though this network was giving state-of-the art results as claimed in the paper[9], such a network will most likely not generalize well when applied to both rotated and scaled images.

A new network (figure2.2) with half the number of feature maps as the old network was created and trained on standard MNIST dataset, but tested on rotated MNIST data-set. Similar to the test networks this network performed poorly giving about 60% classification accuracy.

In the next step a 1000 rotated images were added to the MNIST dataset and the network was retrained. Even the addtion of such a small number of rotated images dramatically increased the performance for the HN which now gave a classification accuracy of 85%.

Gradually more rotated samples were added until the network's accuracy increased to about 90%.The final training set therefore contained 55,000 unrotated images and

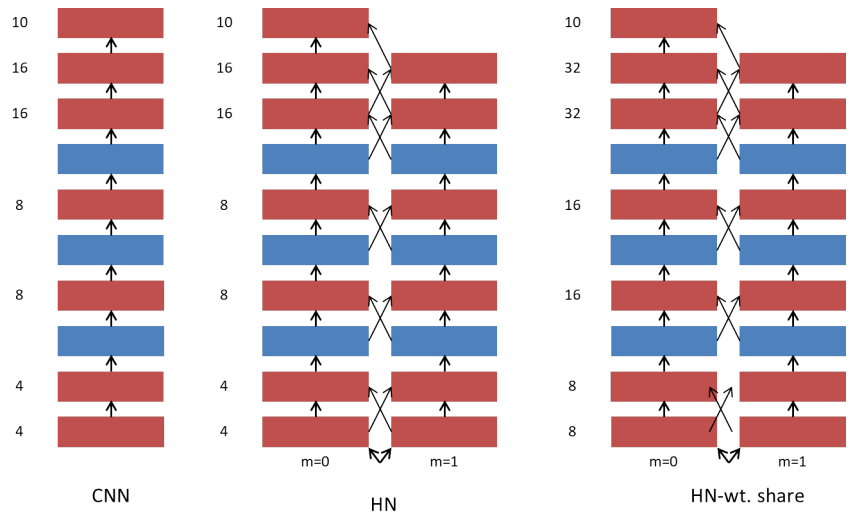

Figure 2.2: The different models used for comparision.Red blocks correspond to correlation and blue blocks correspond to pooling.

5000 rotated images.The final test set contained 10,000 rotated test images. This network shall from now on be referred to as HN.

In the table 2.2 the performance of the harmonic net is compared to that of a CNN with the same number of feature maps and layers. As expected the HN gives superior performance compared to the CNN on rotated images.

Network	MNIST	MNIST +5k rot images
CNN	0.3823	0.8405
HN	0.5607	**0.9052**

Table 2.2: Test accuracy for different Networks on different Training sets for a rotated test set.

2.2.3 Rotated and Scaled MNIST

Finally weight Sharing was applied to the filters in the harmonic nets and as the number of learn-able parameters are decreased per-filter, the total number of feature maps were doubled to keep the total number of parameters in the network same.

This network was then trained on the standard MNIST dataset along with 5000 rotated and scaled images. The test set contained 20,000 rotated and scaled images and the total calssification accuracy obtained was 86% about 4% higher than the HN which was trained and tested on the same dataset.

A bench-mark CNN model with the same number of feature maps as HN was also trained on the same datasets for comparison. The table 2.3 contained the detailed results obtained by different networks on different datasets.

Network	MNIST +5k rot images	MNIST +5k rot and sclaed images
CNN	0.3467	0.6153
HN	0.4439	0.8259
HN-wtshare	**0.7832**	**0.8616**

Table 2.3: Test accuracy for different Networks on different Training sets for a rotated and scaled test set.

2.3 Conclusion

From the results obtained in the previous section(table 2.3) it is clear that the weight sharing approach indeed improves performance in scaled datasets and the architecture can be further optimized to compete for state-of-the-art performance on rotated and scaled MNIST.

CHAPTER 3

Capsule Nets

The work done on Capsule Nets is discussed in this chapter.

3.1 Capsules

The concept of capsules was first introduced in the paper Transforming Auto-Encoders[2]. The paper claims that the widely used CNN filter banks are not a good method for dealing with complicated recognition tasks like facial identification, which require precise spatial information regarding high level parts like face and mouth. In a regular CNN these spatial relation are gradually lost due repeated pooling or sub-sampling operations, and though such a property is sometimes is desirable as it makes them invariant to spatial changes, it makes them incapable of computing exact spatial relationships.

The paper claims that rather than using a single neuron which outputs a scalar values ,networks should use "capsules" which perform computations in their local field of view and encapsulate the results in the form of a vector. The dimension of the vectors then learn to represent certain features in the capsules local region. For example a vector dimension could represent the probability for the existence of a particular feature, which is view-point invariant and its other dimensions could output view-point equi-variant representations of the said feature.

The other major advantages of capsules is their ability to form part-whole relationships between other capsules. A good example for this is provided in the paper itself,consider the problem of facial recognition. Assume two capsules that represent nose and mouth respectively.Assume they both are connected to a higher level capsule which represents a face. Both the mouth and face capsules give predictions about the representation for face capsule and if these predictions agree it means that the face is present in a manner following a linear manifold part whole relationship corresponding to the representation.

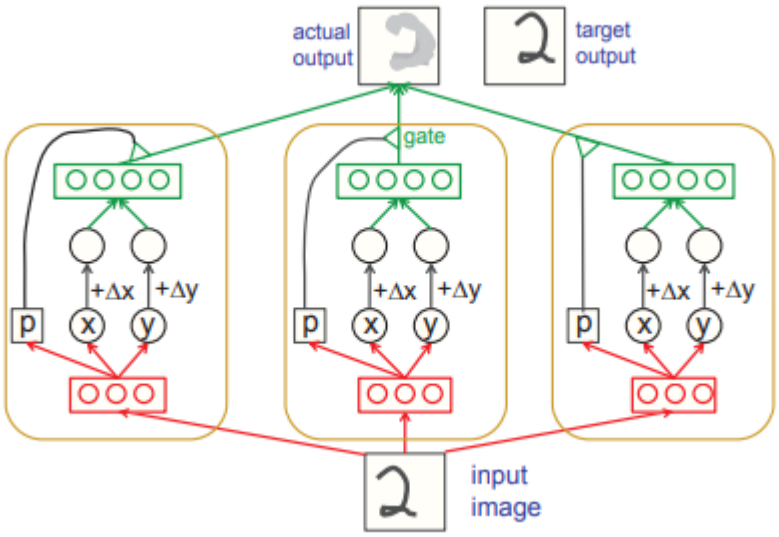

Figure 3.1: Three capsules of a transforming auto-encoder that models translations. Each capsule in the figure has 3 recognition units and 4 generation units. The weights on the connections are learned by backpropagating the discrepancy between the actual and target outputs.

3.2 Transforming auto encoders

The transforming auto encoders in the paper are trained to reconstruct a transformed image from the base image and the transformation matrix, after passing them through a capsule layer.

In essence the capsules are made to encode inforamtion in the image in a vector. This vector is used as input to another network which attempts to reconstruct the image. In effect these auto-encoders are able to achieve some sort of a viewpoint equi-variant information "compression" of the input. This experiment forms the core idea of the project, which is to explore how the capsule vectors manage to encode equi-varaint representations of features.

The auto-encoder demonstrated in the paper is shown in the figure3.1. [1]

[1]Image taken from Transforming Auto-Encoders.[2]

3.3 Caps-Net

The idea of using capsules in a neural network was finally successfully implemented in the paper Dynamic Routing between Capsules [4] . This paper describes the use of a Caps-Net and a routing algorithm between capsule layers, and is able to beat state of the art performance on the MNIST dataset. The following subsections cover important aspects of Capsule-Nets in brief.

3.3.1 Primary Capsules

The first layer of capsules also called the primary capsules are derived by reshaping feature maps extracted from a few convolutional layers as vectors. The first convolutional layer uses 256 kernels of size 9x9 with stride of 1 to generate the first set of feature maps. The second layers again uses 256 kernels of 9x9 size and stride 2 to generate 256 6x6 feature maps. These are then reshaped into 8-D vectors leading to a total of 1152 capsules.

3.3.2 Non-Linearity : Squash

The vectors of the these capsules are then passed through the squash function (eqn3.1), this function squashes the length of large vectors to slightly less than 1 and that of small vectors towards 0 while keeping their orientation unchanged. The length of a capsule after passing through the squash function encodes the probability for the existence of the entity represented by the capsule.

$$v_j = \frac{||s_j||^2}{1 + ||s_j||^2} \frac{s_j}{||s_j||} \tag{3.1}$$

3.3.3 Secondary Layer

The Secondary layer of capsules is used to predict probability of existence of each digit, therefore it has a total of 10 16-D capsules and the predicted label is extracted corresponding to the capsule with the largest length.

3.3.4 Prediction Vectors

Each of primary capsules is required to generate a prediction vectors for output of each of the secondary layer of capsules.These are used to calculate outputs for secondary capsules.These prediction vectors are obtained by multiplication via a unique and learnable 16x8 weight matrix W as shown in the eqn3.2.

$$u_{j|i} = W_{ij} u_i \tag{3.2}$$

3.3.5 Dynamic Routing

Once the prediction vectors are obtained the dynamic routing algorithm is implemented to generate the prediction vectors for the secondary layer of capsules. figure3.2.

Procedure 1 Routing algorithm.

1: **procedure** ROUTING($\hat{\boldsymbol{u}}_{j|i}$, r, l)
2: for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow 0$.
3: **for** r iterations **do**
4: for all capsule i in layer l: $\mathbf{c}_i \leftarrow \texttt{softmax}(\mathbf{b}_i)$ ▷ `softmax` computes Eq. 3
5: for all capsule j in layer $(l+1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij} \hat{\mathbf{u}}_{j|i}$
6: for all capsule j in layer $(l+1)$: $\mathbf{v}_j \leftarrow \texttt{squash}(\mathbf{s}_j)$ ▷ `squash` computes Eq. 1
7: for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}.\mathbf{v}_j$
return $\mathbf{v}_j$

Figure 3.2: Dynamic Routing Algorithm as described in the paper

Once we have the prediction vectors $u_{j|i}$, we initialize routing weights b_{ij} to 0. In the next step the weights are passed to a softmax(eqn3.3) to give coupling coefficients. These are multiplied with the prediction vectors and all predictions for a given secondary capsule are summed to get the output of a secondary capsule. The secondary capsules are then squashed and the routing weights are updated by adding the dot product of the output vector and the prediction vector to the routing weights. This process is repeated required number of times.

$$c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{ik})} \tag{3.3}$$

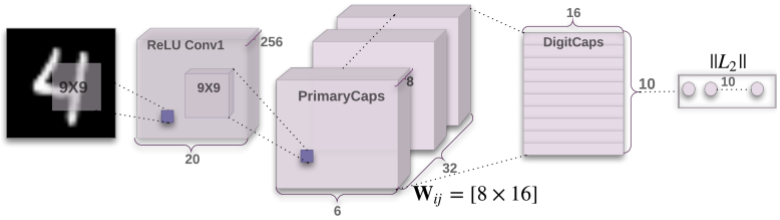

Figure 3.3: The capsule net model taken from Dynamic Routing between Capsules[4]

3.3.6 Margin Loss

The margin loss for existence of a digit is defined as in eqn 3.4, this loss ensures correct classification even when multiple digits are present

$$L_k = T_k max(0, m^+ - ||V_k||)^2 + \lambda(1 - T_k)max(m^-, ||v_k||)^2 \tag{3.4}$$

Where, $m^+ = 0.9$,$m^- = 0.1$,$\lambda = 0.5$.

3.3.7 Reconstruction Loss

The output of the capsule network is then sent to a feed forward neural network that reconstructs the image, the reconstruction loss is defined as the mean squared loss between the original and reconstructed image. This encourages the capsule to learn meaningful representations of each dimension.

3.3.8 Estimation Loss

In order to encourage the capsule to explicitly learn representations for angle and scale an estimation loss, computed by the squared error between the parameter value and the output of a feed forward neural network(which estimates these parameters)was also added to the total loss.

3.4 Implementation and Results

All the required functions for generating a generalized capsule network were first coded in TensorFlow. Then the model described in the paper was replicated on MNIST to obtain classification accuracy of 99.5%.

The same network was again trained on rotated and scaled images,surprisingly the network was able to give a very good classification accuracy of 91%. This meant either of two things, the capsules had learned transformation invariant features or that the model had sufficient capacity to learn rotated and scaled images.

The accuracies for the Caps-Net on different datasets is shown along with those for a bench-mark CNN is shown in the table3.1.

Network	MNIST	Rotated and scaled MNIST
CNN	0.9940	0.8944
Caps-Net	0.9949	0.9099
Caps-Net(est-loss)	-	0.9236

Table 3.1: Test results for MNIST and rotated and scaled MNIST on CNN and Caps-Net,values represent accuracies.

Attempt was made to train several feed forward neural networks on the outputs of capsule vectors to predict the rotation angle and scale factor. The tolerance for scale was set at 0.075 and for angle it was set to 27°. The models did not yield good accuracies implying that the capsule by themselves have not learned elegant or continuous representations for rotation and scale.

The parameter prediction accuracies for different feed forward neural networks are shown in the table3.2.

Parameter	Linear	512x1024	1024x1024
Scale	0.3718	0.5430	0.5464
Theta	0.3796	0.6491	0.6433
Scale&Theta	0.3669 | 0.3774	0.5159 | 0.6224	0.5217 | 0.6149

Table 3.2: Test results for prediction of scale and angle from outputs of Capsule net using several feed forward neural networks.The Column labels represent size of hidden layers.

In order to encourage the capsules to learn better representations for angle and scale end-to-end models were trained first for estimation of individual parameters and then later to both angle and scale together. As expected adding estimation loss improves both the classification accuracy as well as the parameter estimation accuracy. The accuracies for various estimator models along with classification accuracies are shown in the table3.3.

Parameter	Linear	512x1024	1024x1024
Scale	0.7755	0.8156	0.8073
Theta	0.5316	0.7059	0.6972
Scale& Theta	0.7673 | 0.4874	0.8010 | 0.6825	0.7965 | 0.6693

Table 3.3: Test results for prediction of scale and angle for different estimation net architectures.The Column labels represent size of hidden layers in the estimator network.

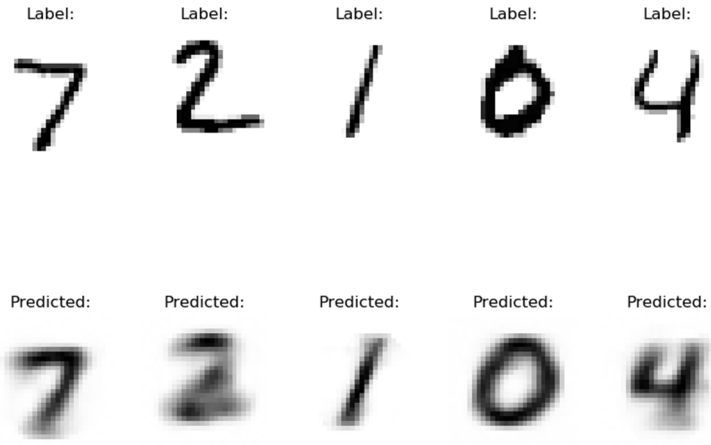

Figure 3.4: The original and reconstructed images of the standard MNIST Dataset

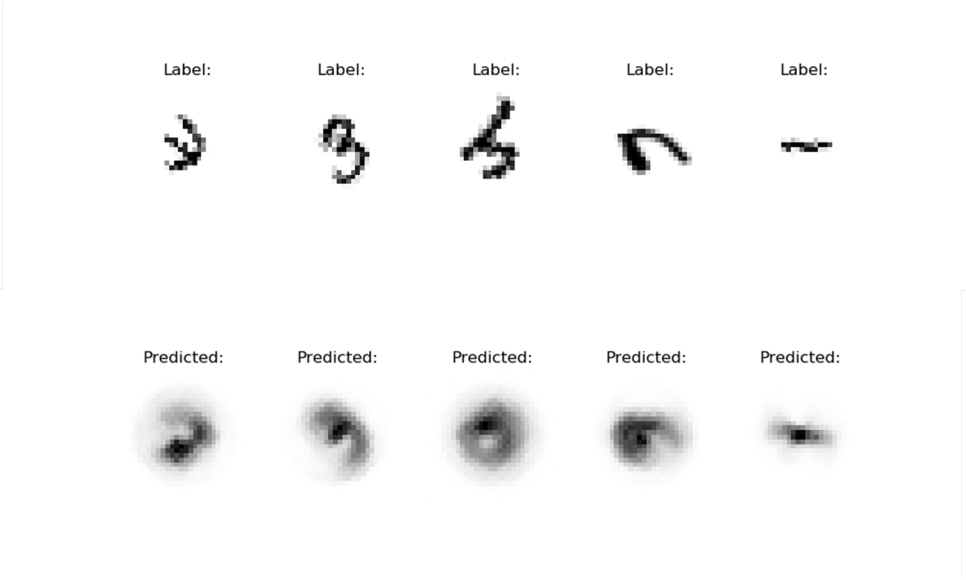

Figure 3.5: The original and reconstructed images of the rotated and scaled MNIST Dataset

3.5 Conclusion

Even though training end-to-end models improves the parameter estimation accuracy it is still not as high as expected. This means even though the capsules are able to learn about rotation and scale(as evident from reconstructed images figure3.5) it is not in a linear or a continuous manner.

Another interesting observation is that the capsules are able to provide higher accuracies for scaling than for rotation indicating that they are inherently better at learning representation for scale compared to rotations.

Both scale and angle estimation accuracies saturate around 82% and 72% respectively implying that this must be the limit to the representational capacity of the capsules.

The accuracies of 82%, 72% can also be misleading in a way,because this also includes cases where the network's prediction for digit itself was wrong.The classification accuracy of the model itself was around 92%, when we factor in this into consideration ,we can say that the actual estimation accuracies are higher than those in results.

CHAPTER 4

Future Work

1. The wt-share harmonic net can be fined tuned in attempt to further improve its performance.
2. A CNN model whose initial filters perform scale invariant convolution similar to mellin transformed can be learned and such features could be fed as inputs to the harmonic nets.
3. Other methods to encourage the capsule to learn invariant representations can be tried.
4. The capsule net can be trained to predict homo-graphic matrix for MNIST transformed by all general affine transformations.

CHAPTER 5

References

1. Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27-30, 1989]*,pages 396-404,1989.

2. G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders In *Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I*,pages 44-51,2011.

3. W. T. Freeman and E. H. Adelson.The design and use of steerable filters. In *IEEE Transactions on Pattern analysis and machine intelligence*,13(9):891-906, 1991.

4. Sabour, Sara; Frosst, Nicholas; E Hinton, Geoffrey.Dynamic Routing Between Capsules.In *[NIPS]*,2017.

5. Esteves, Carlos; Allen-Blanchette, Christine; Zhou, Xiaowei; Daniilidis, Kostas.Polar Transformer Networks,In *ArXiv e-prints*,1709.01889,september 2017.

6. Jaderberg, Max; Simonyan, Karen; Zisserman, Andrew; Kavukcuoglu, Koray. Spatial Transformer NetworksIn *eprint arXiv:1506.02025*,06/2015.

7. Joseph Rosen and Joseph Shamir.Scale invariant pattern recognition with logarithmic radial harmonic filters.In *Appl. Opt.*,28,pages 240-244,Jan 1989.

8. Joseph Rosen and Joseph Shamir.Circular harmonic phase filters for efficient rotation-invariant pattern recognition.In *Appl. Opt.*,27,pages 2895-2899,July 1988.

9. Daniel E. Worrall and Stephan J. Garbin and Daniya Turmukhambetov and Gabriel J. Brostow.Harmonic Networks: Deep Translation and Rotation equi-variance.In *CoRR*,abs/1612.04642,2016.