

A Review Paper on Understanding Capsule Networks

¹Prasham Lalit Shah, ²Tanay Krishnakumar Gupta, ³Jash Bhupendra Dhakad, ⁴Mitchell Raymond D'silva
¹⁻³Students, ⁴Assistant Professor

¹Information Technology Engineering,
¹Dwarkadas J.Sanghvi College of Engineering, Mumbai, India

Abstract— A CapsNet consists of capsules instead of neurons. A capsule is a group of neurons whose output predicts different features of the same object. Every layer of a capsule neural network consists of many capsules[1]. Capsule Neural Network or CapsNet is a type of artificial neural network(ANN) used in machine learning system that can be used to better represent hierarchal relationships. The network has been developed to overcome the limitations associated with convolutional neural networks(CNN) such as back-propagation, translation invariance and pooling layers by adding capsules to a CNN and reusing the output from several capsules to form more stable representations. The concept of CapsNet can be extended to various applications like Dynamic Routing, EM Routing and other image classification problems. This paper also states the characteristics, advantages and disadvantages of using capsule networks. It also tells us about the future scope of using CapsNet technology.

Index Terms—Capsule Networks, Convolutional Neural Networks, Artificial Neural Network.

I. INTRODUCTION

Geoffrey Hinton, known as one of the fathers of deep learning, published a computer vision concept called capsule networks. Having revolutionized the use of Convolutional Neural Networks for classifications, it was only after publishing capsule network that the use of CNNs have been made feasible and logical[2]. The idea was primarily introduced to overcome the problems and difficulties which occur in Convolutional Neural Network (CNN). CNN being a traditional neural network is made up of neurons. Capsule networks enhance the performance of a CNN by imposing the structure on CNN and grouping neurons into capsules. A capsule is a collection of neurons whose output gives us different properties of the similar features. The output is a 4x4 matrix that is a 16 dimensional vector, where entries in the matrix(or the vector) represent information such as the x and y coordinates of the feature, scale and thickness, localized skew, width and translation and other characteristics. In digit recognition, digits are represented by a layer of capsules. One capsule in the output layer corresponds to a digit. Each and every entry in the 4x4 matrix is different and it represents properties of the digit. Consider the example below,






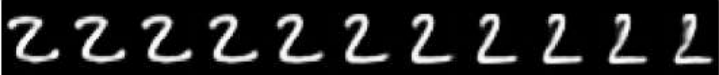
Scale and thickness	
Localized part	
Stroke thickness	
Localized skew	
Width and translation	
Localized part	

Fig.1: Spatial information of the digits [1]

Figure 1 above illustrates the spatial information about the digits [1] which are stored in every entry such as the localized part, scale and thickness, width and translation, localized skew and part etc. This information determines what happens to the digits when there is a change in any one dimension of the capsule output. The decoder network helps to find out what the individual dimensions represent. The decoder network enables the capsules to learn the features that are needed for the purpose of reconstructing the original image. It then calculates an activity vector (vector that stores the probability of entity existence, position and its orientation) for the correct digit capsule and feeds a perturbed version of this activity vector to

the decoder network and observes how the reconstruction is affected by this perturbed activity vector. One entry in the 4x4 matrix (16 dimensions) is always the width of the digit, few entries represent global variations like skew and part and other entries represent changes in localized part of the digit. Figure 1 demonstrates the reconstruction of the digit when one of the 16 dimensions is altered by 0.05 in range of [-0.25, 0.25]. Capsule networks so far have not done anything ground-breaking; after all they are just basic convolutional layers. The uses and benefits of capsules come into foray when the information is transferred from one layer to another which is also called as routing mechanism. CNN on the other hand routes information using max pooling that is not efficient

II. PROBLEMS WITH CNN

This section discusses the problems of CNN network such as max-pooling and translation invariance.

2.1 Max-Pooling

CNN for long has been the traditional way for image detection and classification. CNN functions by detecting the occurrence of attribute and features in an image and based on these features it predicts whether an object exists or not. But the problem with CNN is that it takes into consideration only if the features are present or not [2].

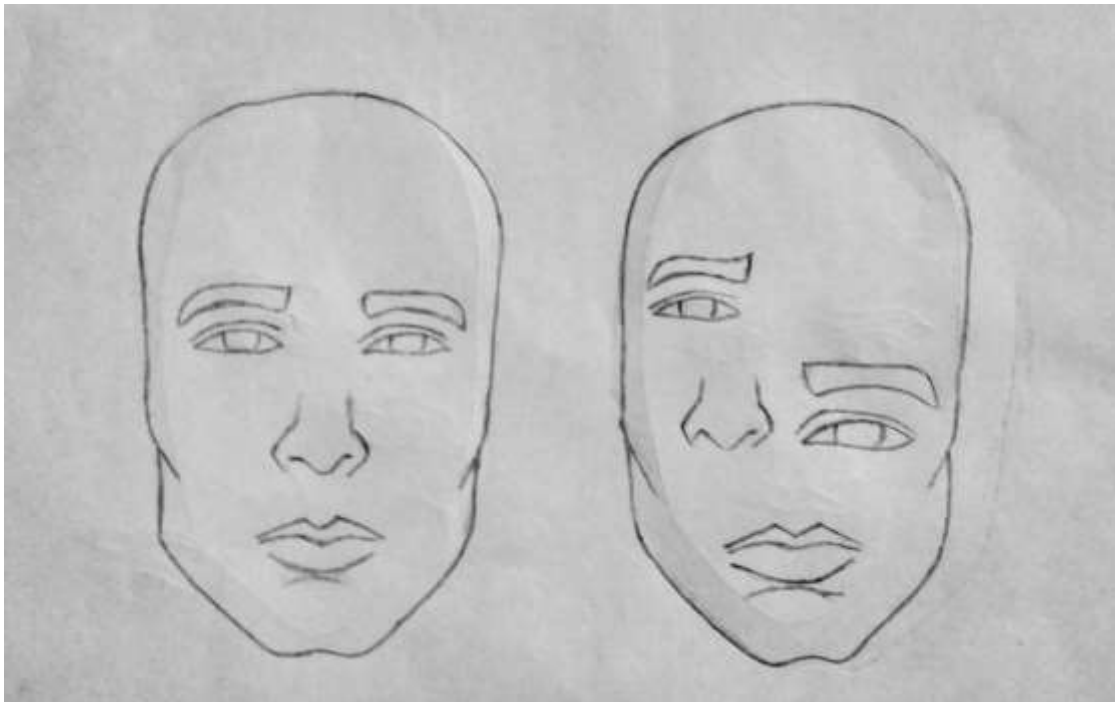


Fig.2: Image classification

Figure 2 shows two images of a face with the same features which are a pair of eyes, a nose and lips. The first image which correctly depicts a human face is correctly identified by CNN. The image to the right is also correctly identified even though it has misplaced features. This is the issue with CNN as it examines only the grid of pixels and takes the maximum activation in that region. That is it only checks whether the attributes and features are present or not irrespective of their position. This problem is called as max pooling. Capsule Networks solve this problem by capturing part-whole relationships which takes into account not only the existence of the features but also their orientation. So in the above example when we apply CapsNet, the image to the left will again be correctly identified as a face but the image to the right will not be identified as a face as the features such as eyebrows, eyes and nose are not properly oriented.

2.2 Translation invariance

Max-pooling problem leads to translational invariance and this invariance also triggers false positive for images which have the components of the correct image but not in correct order and position. That is even if an object is rotated clockwise or anticlockwise by certain amount of degrees, it will still be identified as the same object in CNN which is wrong as by changing the orientation of an object it no longer remains the same object.



Fig.3:Disfiguration Transformation

In figure 3, the invariance problem matches both the images even though the right one is having incorrect orientation. There can be a certain case where the image shown on right could be of a smaller size as illustrated in figure 4. CNN reduces the size of the original image and tries to match it with the smaller image which is not correct as by changing the orientation and size it no longer remains the same object. CapsNet solves this problem by using equivariance which understands the rotation and the proportion change and adapts itself accordingly to preserve the spatial information[3].Equivariance takes into consideration positional, proportional and translational invariances.



Fig.4: Disfiguration Transformation example

III.WORKING OF CAPSULES

Figure 5 explains the working of capsules in the following 4 steps:

1. The capsule in the given example receives 3 input vectors namely u_1 , u_2 and u_3 . These vectors are found in the capsules present in the layer just below it. These vectors have two properties: length and direction. Length finds out the probability of detecting the object and the direction determinesthe path that the capsule will follow.

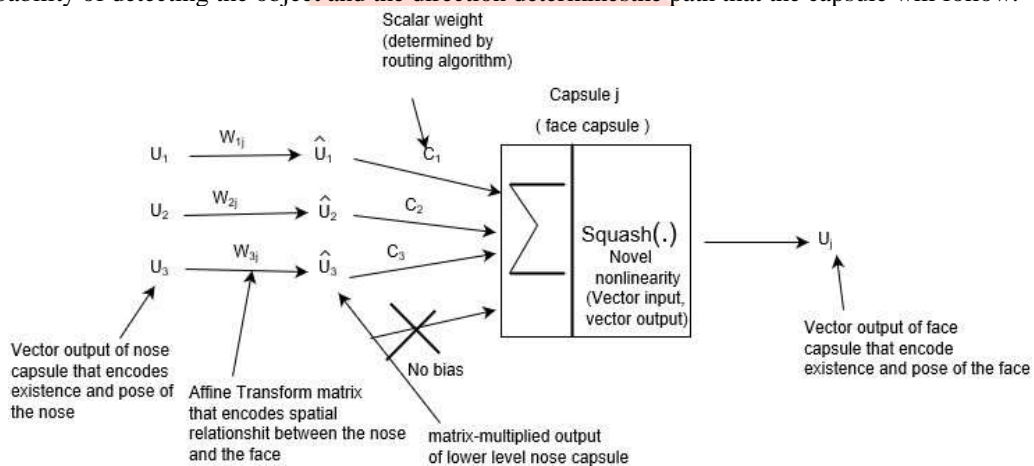


Fig.5:Working of Capsules[11]

Consider that the capsules present in the lower level represents some features of the face, say eyes, nose and mouth. The input vectors are multiplied by matrices which represent relationship between the features of the lower level capsule and the outer level capsule (which contains face). The matrices in this case are W_{1j}, W_{2j} and W_{3j} . W_{1j} shows the relationship between eyes and the face and similarly the other two matrices represent the relationship between nose and face, and mouth and face. After multiplication, the matrix multiplied outputs, \hat{u}_1, \hat{u}_2 and \hat{u}_3 are obtained. \hat{u}_1 shows where the face should be according to the detected position of the eyes, \hat{u}_2 shows where the the face should be according to the detected position of the nose and \hat{u}_3 shows where the face should be according to the detected position of the mouth [11].

2. The second step is explained with the help of the following example as depicted in figure 6.

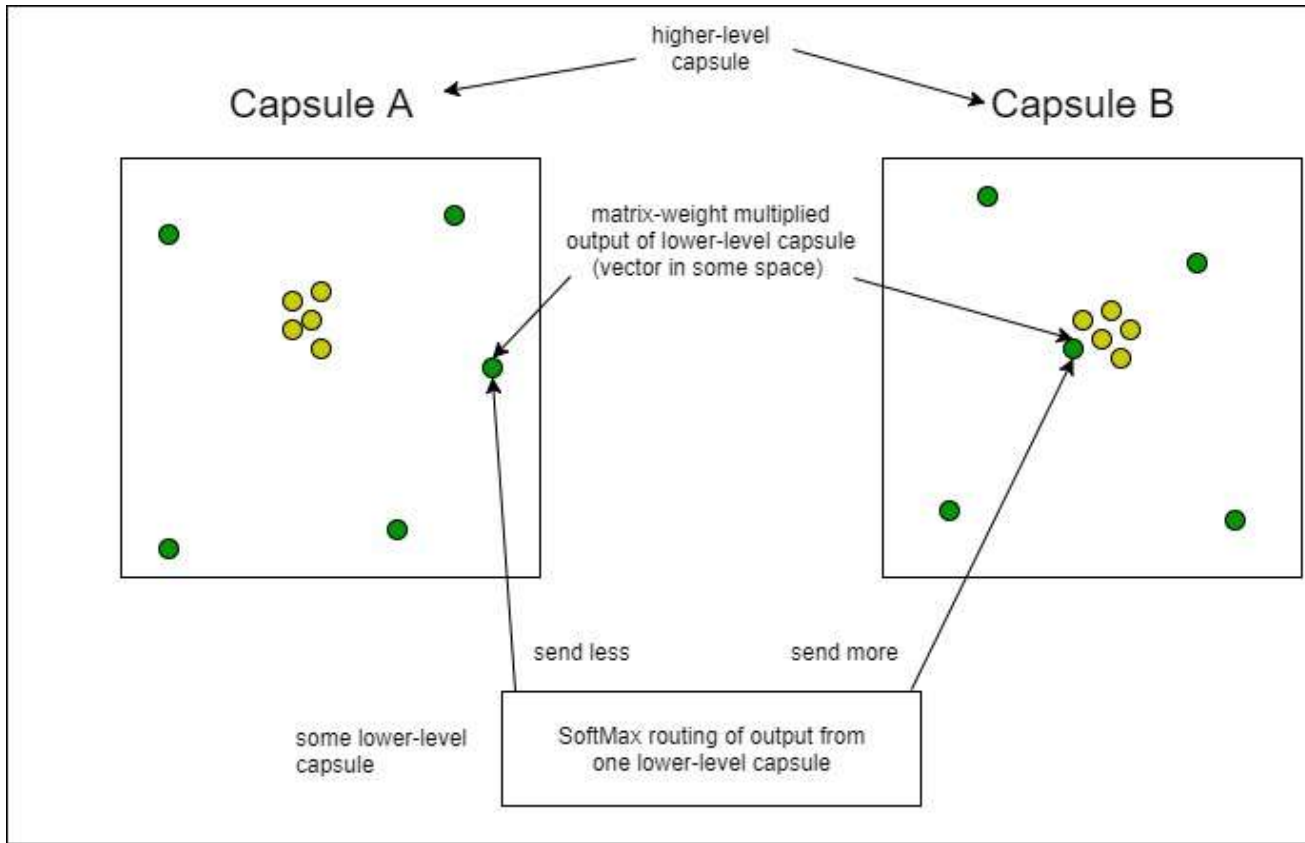


Fig.6:Dynamic routing based on agreement[11]

A lower level capsule can go to a number of higher level capsules. This is decided by adjusting the weight of the lowercapsule and then multiplying it with the capsules output(W_{1j}). The product of this multiplication is then sent to both the higher level capsules A and B to check in which of the higher capsules the output is nearer to the cluster of input vectors.This is in general is called the dynamic routing by agreement process.

3. In this step all the inputs vectors are combined by adding them.
4. The last step involves squashing. Squashing is a process which reduces the length of the vector to a maximum value of 1.The squashing function is given by the equation listed below:

$$v_j = \frac{\|s_j\|^2}{1+\|s_j\|^2} \frac{s_j}{\|s_j\|} \dots\dots\dots(\text{Equation 1}) [11]$$

where, s_j is the input vector and v_j is the output of the higher level capsules[11].

IV.APPLICATIONS OF CAPSNET

4.1Dynamic Routing:

This section describes the algorithm for dynamic routing[4]

1. procedure routing($\hat{u}_{j|i}, r, l$)
 2. for all the capsules i in layer l and capsule j in layer (l+1);
 $b_{ij} \leftarrow 0$
 3. for the r iterations do
 4. for all the capsules i in layer l:
 $c_i \leftarrow softmax(b_i)$ [softmax computes equation 3]
 5. for all the capsules j in layer (l+1):
 $s_j \leftarrow \sum_i c_{ij} \hat{u}_{j|i}$
 6. for all the capsules j in layer (l+1):
 $v_j \leftarrow squash(s_j)$ [squash computes equation 1]
 7. For all the capsules i in layer l and capsule j in layer (l+1):
 $b_{ij} \leftarrow b_{ij} + \hat{u}_{j|i} \cdot v_j$
- for return v_j [4]

$$v_j = \frac{\|s_j\|^2}{1+\|s_j\|^2} \frac{s_j}{\|s_j\|} \dots\dots\dots(\text{Equation 2})[4]$$

$$s_j = \sum_i c_{ij}, \quad \hat{u}_{j|i} = w_{ij} u_i \dots\dots\dots (\text{Equation 3})[4]$$

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})} \dots\dots\dots (\text{Equation 4})[4]$$

Where c_{ij} is a non-negative number and all its coefficients sum up to 1;

$\hat{u}_{j|i}$ is a prediction made by capsule i ;

w_{ij} is a weight matrix

u_i is output of capsule i

b_{ij} is a temporary coefficient that is constantly changed and is initialized to 0

Dynamic Routing is more efficient than max-pooling that is used in CNN, as capsules handle various visual stimulus and encode features like albedo, deformation, size, orientation and many more. The underlying principle of dynamic routing algorithm involves the transfer of inputs from lower level capsules to higher level capsules that match with its input.

In the algorithm given above, the routing algorithm takes as input all the capsules in lower level l , their output \hat{u} and the number of routing iterations r . It produces an output of higher level capsule v_j . The algorithm calculates the network's forward pass. The second step introduces a new variable b_{ij} that is initialized to zero and is a temporary value that is updated iteratively and is stored in c_{ij} at the end of the procedure. Steps 4 to 7 are then repeated r times where r is the number of routing iterations. In step 4, c_{ij} is calculated which stores routing weights for a lower level capsule i . The same procedure is done for all lower level capsules. The softmax function which calculates the probability distribution of a given event over 'n' different events, guarantees that every weight c_{ij} is a positive number and they add up to 1. It also ensures that the probabilistic nature of c_{ij} is enforced.

At the end of first iteration, the values of all the coefficients of c_{ij} are same, since in step two all b_{ij} values will be zero. If there are 3 lower level capsules and 2 higher level capsules, then c_{ij} will have all values set to 0.5. Variable c_{ij} having equal values at initialization of algorithm shows the situation of maximum confusion and uncertainty. Lower level capsules are unaware about which higher level capsules will best suit their output. However, uniform distributions will change with every iteration of the process. Step 5 determines a linear combination of input vectors, weighted by routing coefficients c_{ij} [12]. This step basically involves scaling of input vectors and adding them up which produces an output vector s_j . This step is done for every capsule in higher level. In step 6, vectors from step 5 are passed through squash non-linearity. Squashing makes sure that the alignment and direction of the vector is preserved and a threshold value of 1 is enforced to ensure that its length does not exceed 1. This produces output vector v_j for higher level capsules.

In short, steps 4 to 6 are used to obtain the output of higher level capsules and in step 7, the weight are updated. With respect to each higher level capsule j , step 7 examines each input and alters the corresponding weight b_{ij} as per equation 4 which states that the new weight value equals to the old value plus the dot product of current output of capsule j and the input to this capsule from a lower level capsule i . The dot product finds out the similarity between the input to the capsule and output from the capsule. This process is repeated until we obtain a collection of routing coefficients that effectively match the results obtained from capsules in lower layer with results of higher level capsules. It is found that 3 routing iterations produce an effective result.

5.2 EM Routing

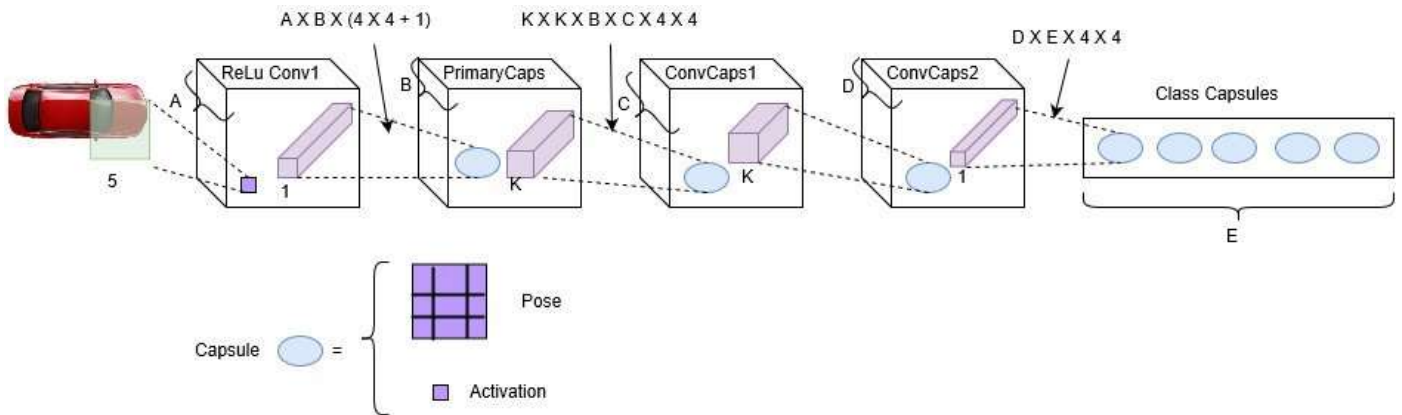


Fig.7: Matrix Capsules with EM Routing[2]

In EM Routing, a capsule in a layer votes for the pose matrix of the capsules which are in the above layer by multiplying its pose matrix with trainable viewpoint invariant transformation matrices that will help to represent part-whole relationships. A pose matrix shows the pose information such as rotation, skew and translation between the object and viewer. All these votes are weighted by an assignment coefficient using the Expectation - Maximization algorithm in which the coefficients are repeatedly updated for all the images such that the output of each capsule is routed to a capsule which is present in the above layer that receives many similar votes. The reason for using capsules in EM Routing is to reduce the number of errors by 45% as compared to the state of the art techniques. Capsules are also more resistant to white box adversary attacks.

Figure 7 illustrates a network with one ReLU convolutional layer followed by a primary convolutional capsule layer and a couple of more convolutional capsule layers. ReLU is the most common activation function used in deep learning models. The algorithm occurs in 2 steps, Expectation and Maximization. Generally both of these steps are done three times and then terminated[2].

Figure 7 starts with a 5x5 convolutional layer with $A=32$ i.e. 32 channels and a stride of 2 with a ReLU non-linearity. All the remaining layers are also capsule layers. The 4x4 pose of each of the capsules with $B=32$ are learned linear transformations of the output of the lower layer ReLUs which are entered at that location. The activations are produced by applying sigmoid function to the weighted sums of the lower layer ReLUs.

After the primary capsules, there are two 3x3 convolutional capsule layers with $K=3$ and both of them have 32 capsule types ($C=D=32$) with C and D having a stride of 2 and 1 respectively. There is another layer which is the last one and is connected to the last capsule layer which has one capsule per output class[5].

To avoid losing information about the location of the convolutional capsules, the transformation matrices are shared between different positions of the same capsule type and then the scaled coordinates of the center of the receptive field of each capsule is added to the first two elements of the right-hand column of its vote matrix. This process is called coordinate addition.

Thus expectation step assigns probabilities that all the capsules in layer $(l+1)$ are explained by capsules in layer l . Maximization step checks if the capsules in layer $(l+1)$ are active or not and decides what input is given to the capsule by taking a weighted average of the routed inputs[2].

VI. CHARACTERISTICS OF CAPSULE NETWORKS

This section explains the various characteristics of Capsule networks.

1. Better connection between layers

In CapsNet the layers are nested together which leads to a better connection scheme and also leads to better and faster solutions. To connect all the features from one layer to another layer fully connected matrices are used which also increases the computational efficiency. This feature is very useful in image classification.

2. Less back-propagation

In CapsNet, the outputs are sent from the lower level capsules to the higher level ones which have a high scalar product. This characteristic of CapsNet helps to reduce back-propagation of errors which are found in deep neural networks.

3. Higher accuracy

The work implemented in [7], introduced a new activation function that used lesser epochs and achieved a 2.57% accuracy improvement over the traditional MNIST model. Capsule networks are used with Non-Parametric Transformation Networks (NPTNs), replacing the convolution layers in primary Capsules with NPTN layers, using same number of parameters. NPTN layers are direct and natural generalization of Convolutional networks and can be optimized directly using gradient descent. This improved the accuracy by 1.03% in MNIST dataset[7].

VII. ADVANTAGES AND DISADVANTAGES OF CAPSULE NETWORKS:

This section describes the advantages and disadvantages of CapsNet.

I] Advantages of CapsNet

The three major advantages of CapsNet are as described below.

1. Viewpoint Invariance - A capsule network recognizes the images and the objects efficiently irrespective of the viewpoint from which they are viewed by the use of their pose matrices.

2. Number of Parameters - CapsNet makes use of capsules which group neurons thereby reducing the number of connections required as compared to that of CNN. A model using fewer parameters can be generalized in a better way and achieves better performance. Thus, a CapsNet model is preferred over CNN. For example a CNN model implementing dynamic routing and having 35.4 million parameters performs worse than a CapsNet model having only 8.2 million parameters [4].

3. Defence against white-box adversarial attacks - The Fast Gradient Sign Method (FGSM) is one of the most common white-box adversarial attacks on CNN. It evaluates the gradient of the pixel and changes it to maximize the loss. This loss reduces the accuracy of CNN to a large extent (below 20%). CapsNet on the other hand is more resistant as it still maintains accuracy above 70% [2].

4. Position and pose information - CapsNet preserves the position and pose information of an object (equivariance).

5. Amount of data - Less data is required to train CapsNet as compared to other neural networks.

6. Object detection and image segmentation - CapsNet is favorable for object detection and image segmentation.

7. Affine transformation - CapsNet are more robust to affine transformation.

8. Activation vectors - In CapsNet, activation vectors such as rotation, thickness, skew etc. can be easily interpreted.

9. Loss of information - CapsNet does not lose information between layers unlike CNN.

III. Disadvantages of CapsNet

The disadvantages of CapsNet are as follows:

1. There is still an uncertainty over the testing of CapsNet on larger images.
2. CapsNet is not suitable for CIFAR-10 models as it has a more varied background which leads to some noise and hence a poor performance.
3. In dynamic routing by agreement algorithm, there is an inner loop which results in slow training.
4. CapsNet cannot identify two very close identical objects. This issue is called as crowding.

VIII. CONCLUSION

CapsNet is a fairly new concept and so has not made many inroads in the field of neural network. As a result it still has a lot of inconsistencies-most notably the amount of time required for routing which leads to issues while parallelizing over GPUs. Also there are some other issues which need to be taken care of such as inadequate testing on larger and complex datasets. Also in EM routing, a lot of time is required to dynamically route from layer to layer. Another issue is that CapsNet has only limited applications and thus more and more experiments need to be conducted to prove that the technique is suitable for all kinds of experiments. But the concept has shown a lot of promise and so over time it is thought that it will become more efficient. At the same time, CapsNet has a good hierarchical structure which has led to increased research on this concept around the world. CapsNet is effective for 3D reconstruction from images by extracting pose information from images and also can be quite useful for grasping in the field of robotics. [13]. Also the reconstructions from capsule networks are stunning [14]. Thus I would like to conclude that though CapsNet has been working on limited applications, the results obtained in them have shown that its working can be extended to a wide array of other applications too.

REFERENCES

- [1] Capsule neural network (2018, September, 2) Wikipedia [Online] Available : https://en.wikipedia.org/wiki/Capsule_neural_network#Human_vision
- [2] Demystifying "Matrix Capsules with EM Routing." Part 1: Overview (2017, November, 23) Blog By Sahaj Garg [Online] Available: <https://towardsdatascience.com/demystifying-matrix-capsules-with-em-routing-part-1-overview-2126133a8457>
- [3] What is a CapsNet or Capsule Network? (2017, November, 1) Blog By Debarko De [Online] Available: <https://hackernoon.com/what-is-a-capsnet-or-capsule-network-2bfbe48769cc>
- [4] Sara Sabour, Nicholas Frosst, Geoffrey E. Hinton, "Dynamic Routing Between Capsules", 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA. arXiv:1710.09829v2 [cs.CV] 7 Nov 2017.

- [5] Geoffrey Hinton , Sara Sabour , Nicholas Frosst , “ MATRIX CAPSULES WITH EM ROUTING” , Published as a conference paper at ICLR 2018.
- [6] Deep Neural Network Capsules (2017 , November , 7) Blog By Eugenio Culurciello [Online] Available: <https://medium.com/mlreview/deep-neural-network-capsules-137be2877d44>
- [7] RinatMukhometzianov , Juan Carrillo , “ CapsNet comparative performance evaluation for image classification “ , Cornell University Library , [arXiv:1805.11195](https://arxiv.org/abs/1805.11195) [cs.CV] , <https://arxiv.org/abs/1805.11195>
- [8] A Nice Easy Tutorial To Follow On Capsule Networks Based On Sabour ,Frosst , And Hinton’s Paper (2017 , December , 17) Blog By KirtiBakshi [Online] Available: <https://www.techleer.com/articles/447-a-nice-easy-tutorial-to-follow-on-capsule-networks-based-on-sabour-frosst-and-hintons-paper/>
- [9] Deep learning : the capsule network revolution (2018 , July , 19) Blog By Laurent Gloaguen [Online] Available: <https://www.spiria.com/en/blog/artificial-intelligence/deep-learning-capsule-network-revolution>
- [10] Introducing capsule networks (2018 , February , 6) Blog By AurélienGéron [Online] Available: <https://www.oreilly.com/ideas/introducing-capsule-networks>
- [11]Understanding Hinton’s Capsule Networks.PartII:How Capsules Work (2017,November,15) Blog By Max Pechyonkin [Online] Available: <https://medium.com/ai%C2%B3-theory-practice-business/understanding-hintons-capsule-networks-part-ii-how-capsules-work-153b6ade9f66>
- [12]Understanding Hinton’s Capsule Networks.PartIII:Dynamic Routing Between Capsules (2017,November,29) Blog By Max Pechyonkin [Online] Available: <https://medium.com/ai%C2%B3-theory-practice-business/understanding-hintons-capsule-networks-part-iii-dynamic-routing-between-capsules-349f6d30418>
- [13]Capsule Networks: A Critique (2018,March,12)Blog By ShubhamJha [Online] Available: <https://medium.com/@jha1shubham/capsnet-a-critique-a4ca2945d06c>
- [14]Understanding Capsule Networks-AI’s Alluring New Architecture (2018,February,12)Blog By Nick Bourdakos [Online] Available: <https://medium.freecodecamp.org/understanding-capsule-networks-ais-alluring-new-architecture-bdb228173ddc>

