Evaluation of ML Networks for Systems Management AR Applications

About Us

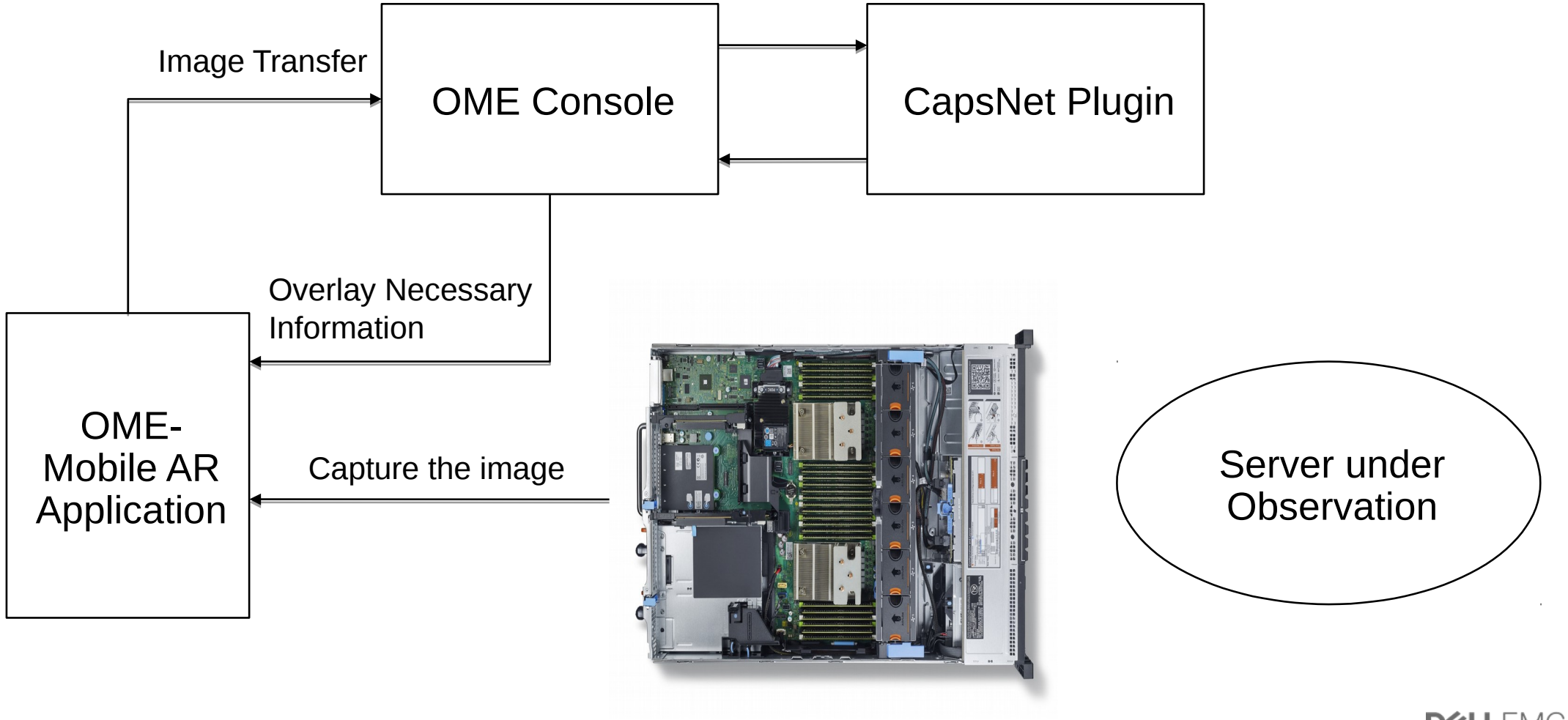
Name : Abhishek Gupta
Institution : IIIT Delhi
Designation : Software Engineer 2
Organization : Dell EMC R&D Center

Name : Megha Tyagi
Institution : IIIT Delhi
Designation : Software Engineer 2
Organization : Dell EMC R&D Center

Agenda

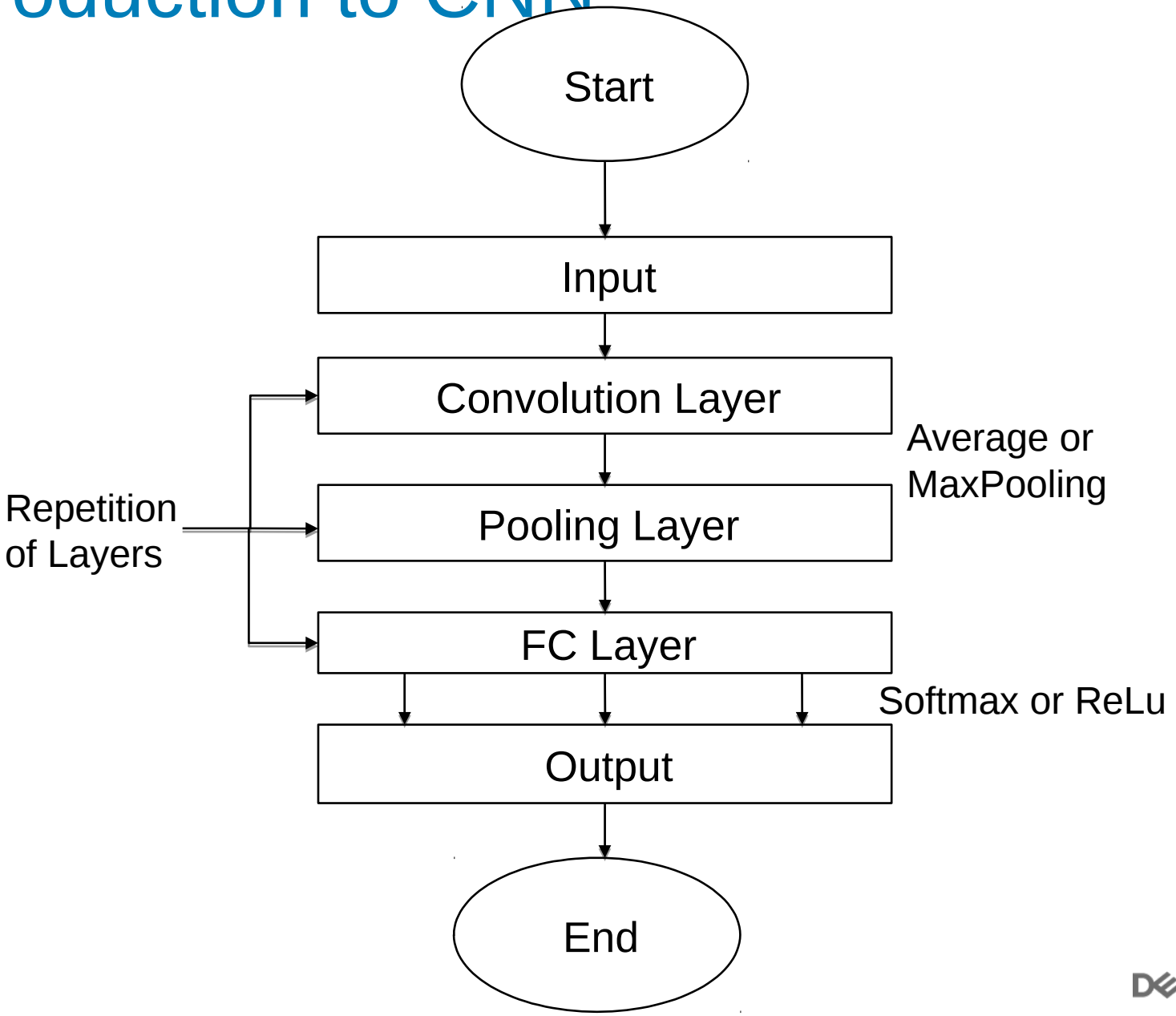
1. Orientation
2. W’s of Capsule Network?
3. CNN vs CapsNet
4. Experimental study and comparative analysis
5. Conclusion

Systems Management Use Case : Use in AR App for identification & Component replacement in Ent. Servers



Introduction to CNN

In deep learning, a **convolutional neural network** (**CNN**, or **ConvNet**) is a class of deep neural networks, most commonly applied for analyzing visual imagery.



Convolutional Neural Networks

- Convolution

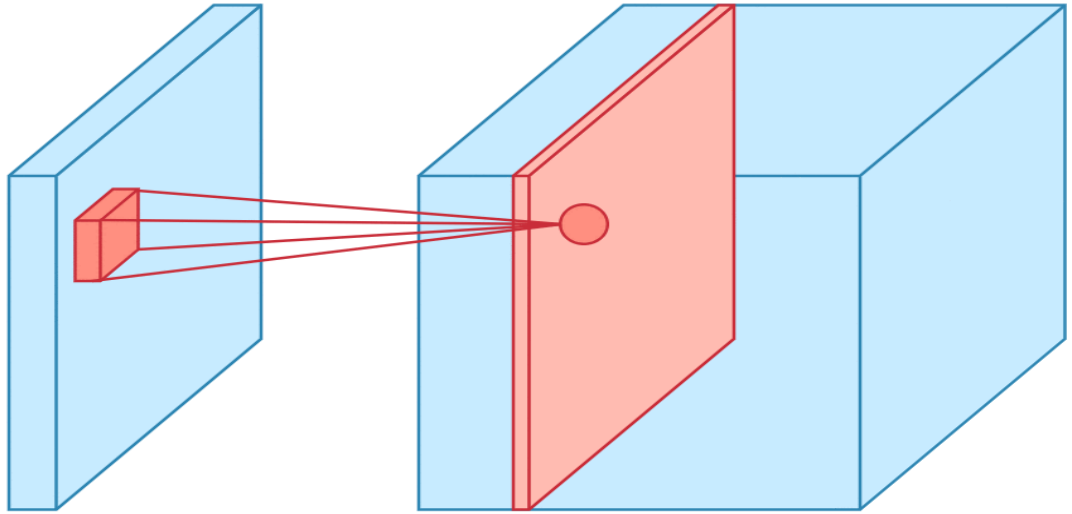
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1	0	1
0	1	0
1	0	1

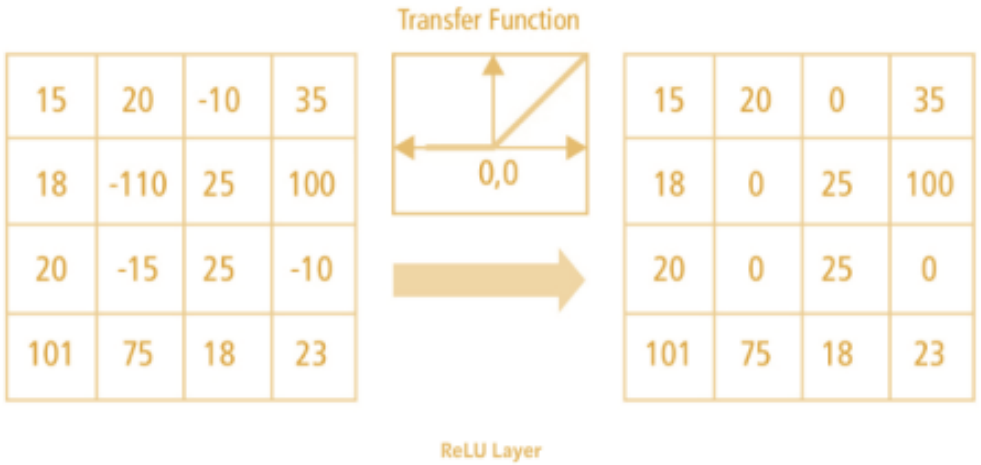
1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		



Convolutional Neural Networks

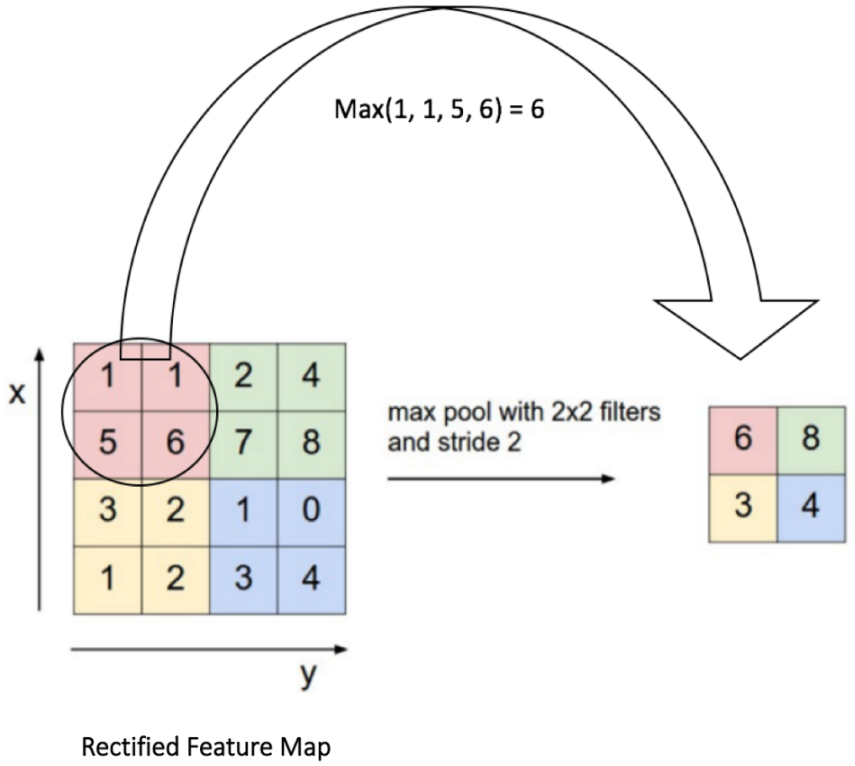
- Non Linearity(ReLu)



15	20	-10	35
18	-110	25	100
20	-15	25	-10
101	75	18	23

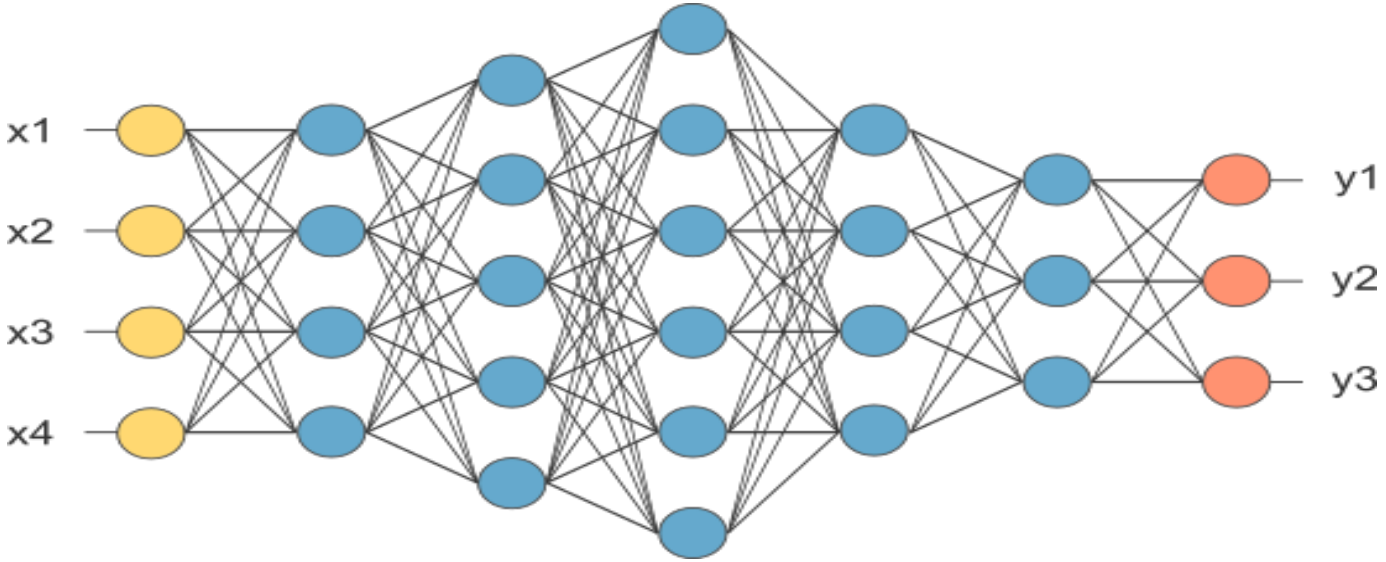
15	20	0	35
18	0	25	100
20	0	25	0
101	75	18	23

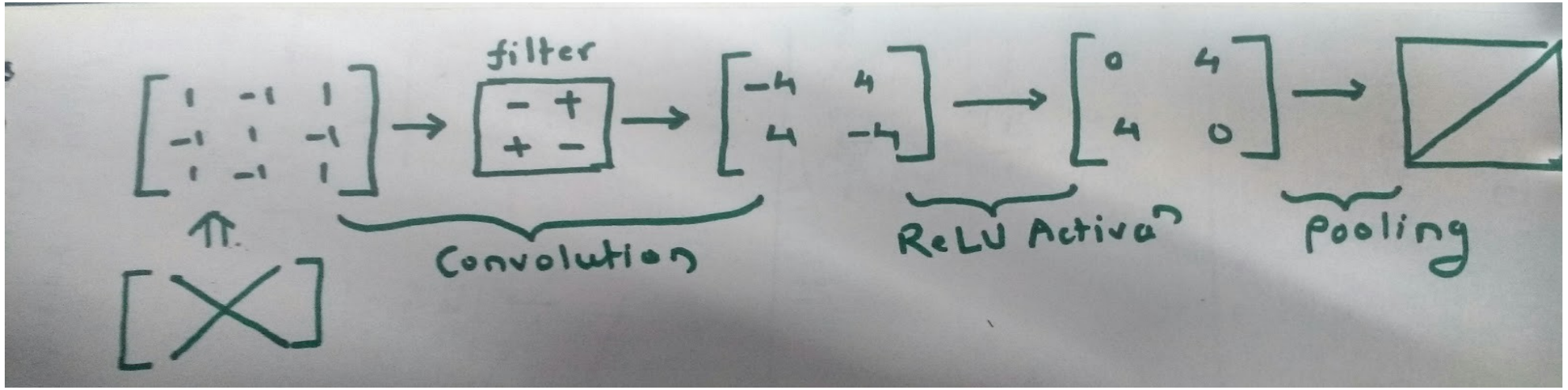
- Pooling



Convolutional Neural Networks

- FC layer





What is Capsule Network ?

***CapsNet** is a ML system that is a type of ANN that can be used to better model hierarchical relationships.*

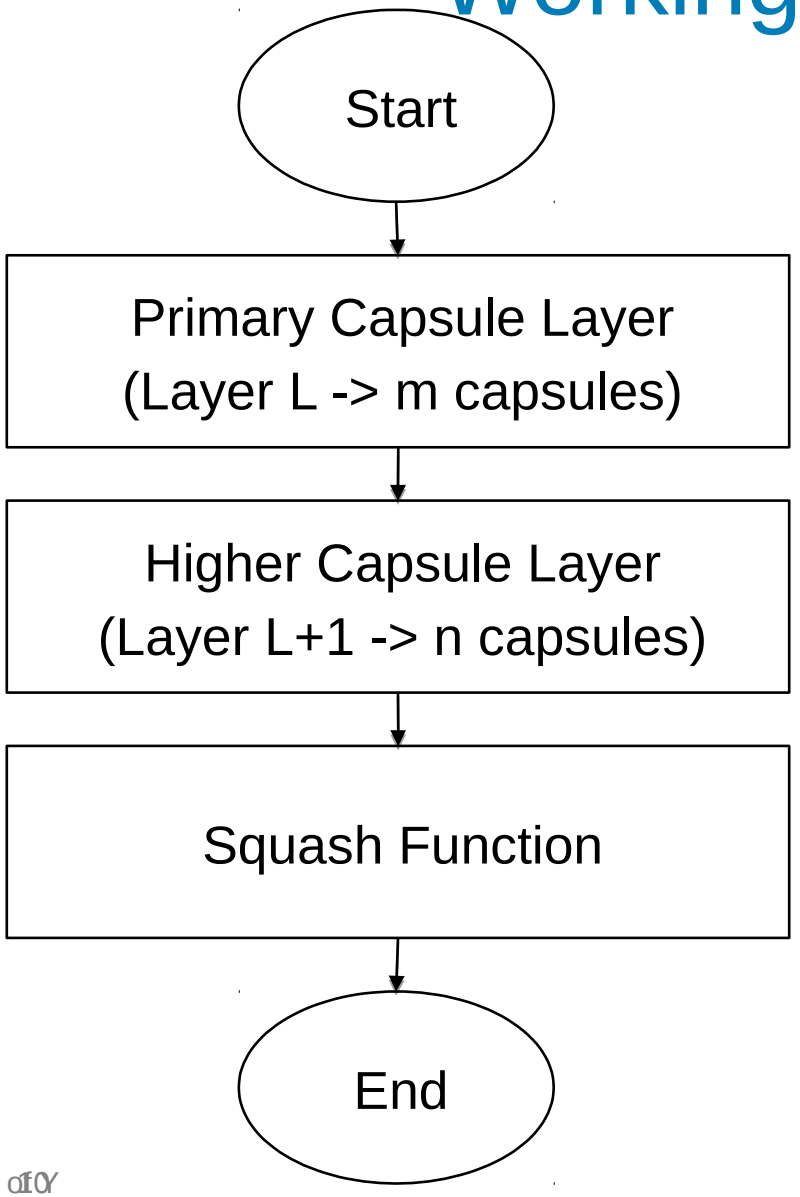
- x_1 NICs
- x_2 DIMMs
- x_3 Controllers
- x_4 FRUs
- x_5 Processors

$\mathbf{s}_j$ = o/p of 5 dimensional server capsule, j

P(server) =

$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2} \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}$$

Working of Capsule Network



$\mathbf{u}_i$ Prediction vector by capsule, i

$\hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij}\mathbf{u}_i$ $\quad \mathbf{s}_j = \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$ capsule, j

$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2}\frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}$$

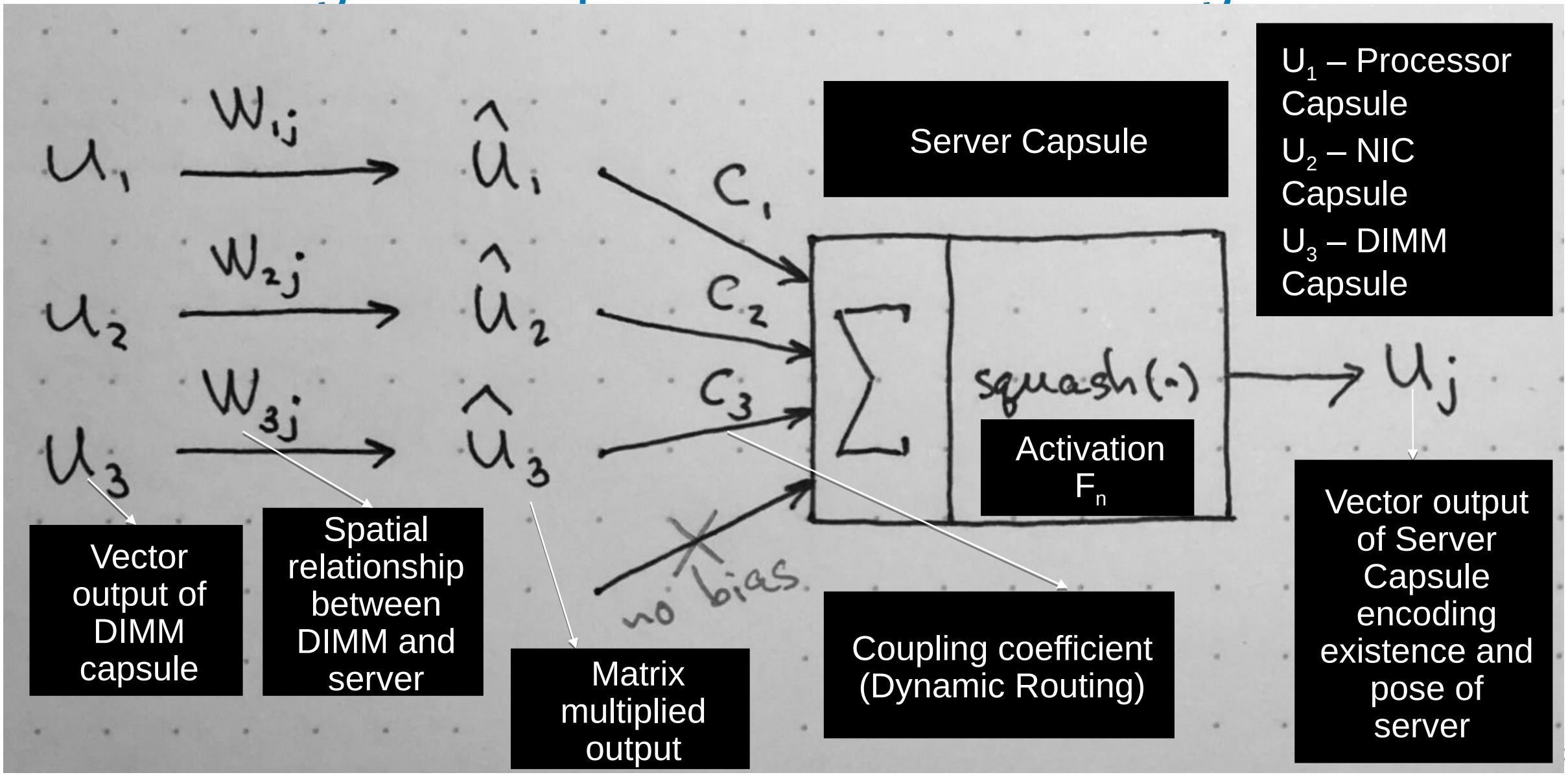
c_{ij}

Iterative Dynamic Routing

Continued…

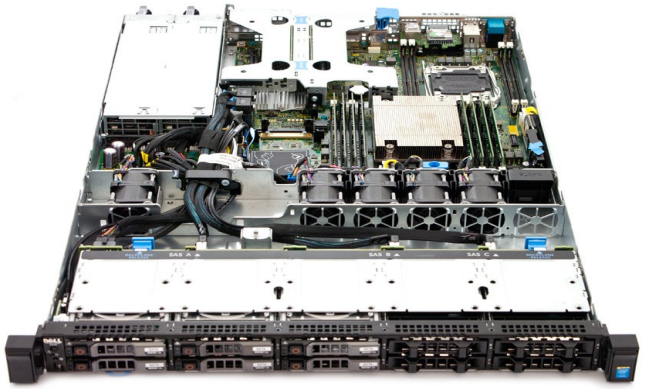
- Capsule networks perform following steps on the input:
  - **matrix multiplication of input vectors**
  - **scalar weighting of input vectors**
  - **sum of weighted input vectors**
  - **vector-to-vector nonlinearity**

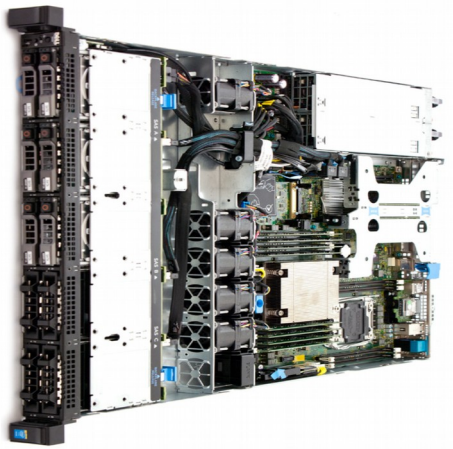
Working of a Capsule for Server Recognition



CapsNet vs CNN

- CNN does not encode the position and orientation of the object into their predictions. This is the problem of Invariance – CNNs can handle translational invariance but not rotational invariance.
  - Example: CNN won't be able to predict the right side as same species if training was carried using left oriented images as depicted in Fig.

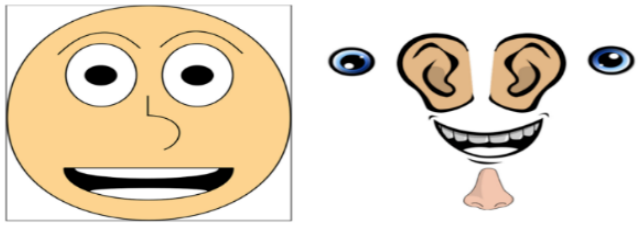




  - A Capsule Network not only detects the features, but also detects its orientation and the relationship between them. If a feature moves around image then though its vector representation will change but not its probability of existence

Continued…

- A CNN makes predictions by looking at an image and then checking to see if certain components are present in that image or not. If they are, then it classifies that image accordingly.
  - Example: In following cases Fig, both left and right will be predicted as face by CNN.



  - CapsNet outputs capsnet vector in which length of vector is probability of existence of feature in image and direction of vector would represents its positional info.

Continued…

- In a CNN, all low-level details are sent to all the higher-level neurons. These neurons then perform further convolutions to check whether certain features are present, instead it is suggested to route the image to specific neurons that have the capability to deal with those features.
  - Presence of a lower level feature along with its activation parameters is determined by each capsule. After lower level capsules detect feature, information is sent to the higher level capsules that have good fit with it.

- Maxpooling layers transfers the activation information from one layer to the next layer in CNN. It tells the layers about the presence of a part, but not the spatial relation between the parts.
  - CapsNet saves spatial relationship between features.

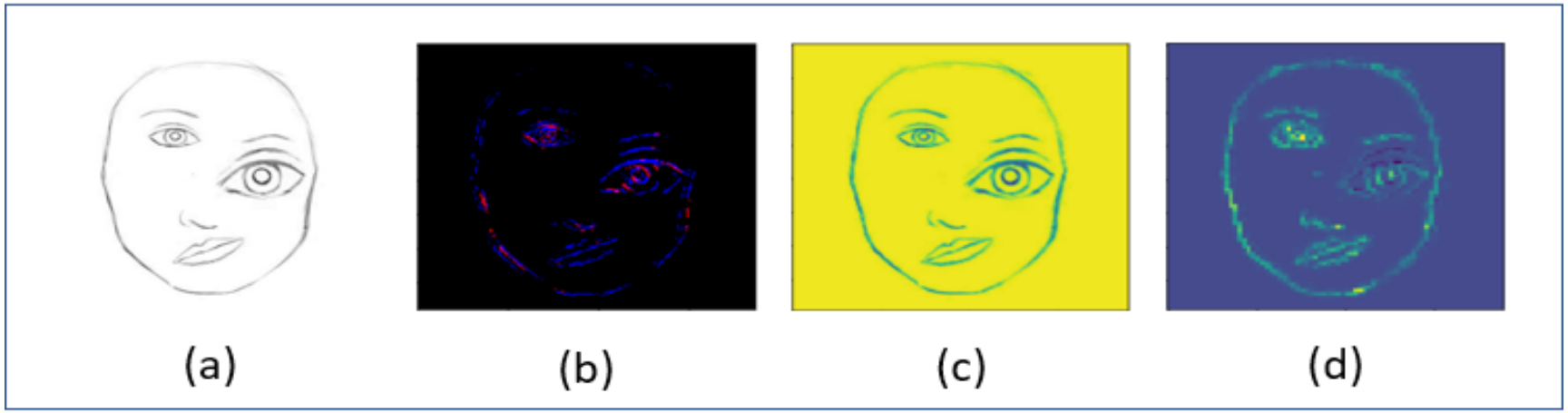
Case Study :

Comparative Analysis of CNN and CapsNet

DELL EMC

CNN Feature Extraction

- CNN feature extraction on an input image of size (398,383,3)



- CNN Rule:

```
if (2 eyes && nose && mouth) {
    It's a face!
}
```

- CapsNet Rule :

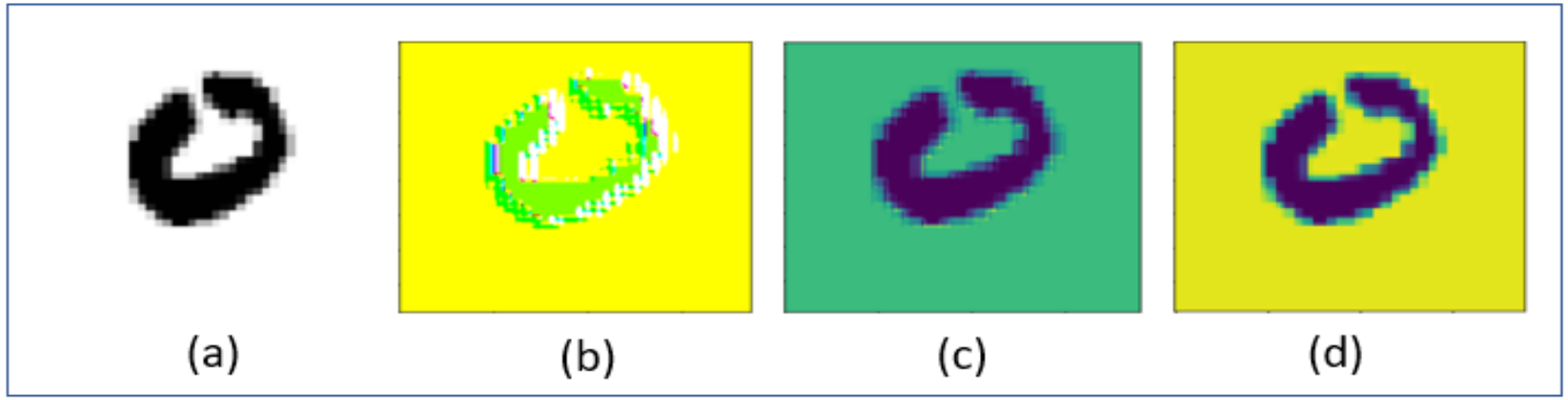
```
if (2 adjacent eyes && nose under eyes && mouth under nose) {
    It's a face!
}
```

Analysis of CNN

- Powerful in identifying the features of the image and predict the image based on the identified features.
- It does not consider the orientation of the features while prediction.
- The CapsNet model considers the pose information and generates the probability of the existence of the entity.

CNN Feature Extraction on MNIST Dataset

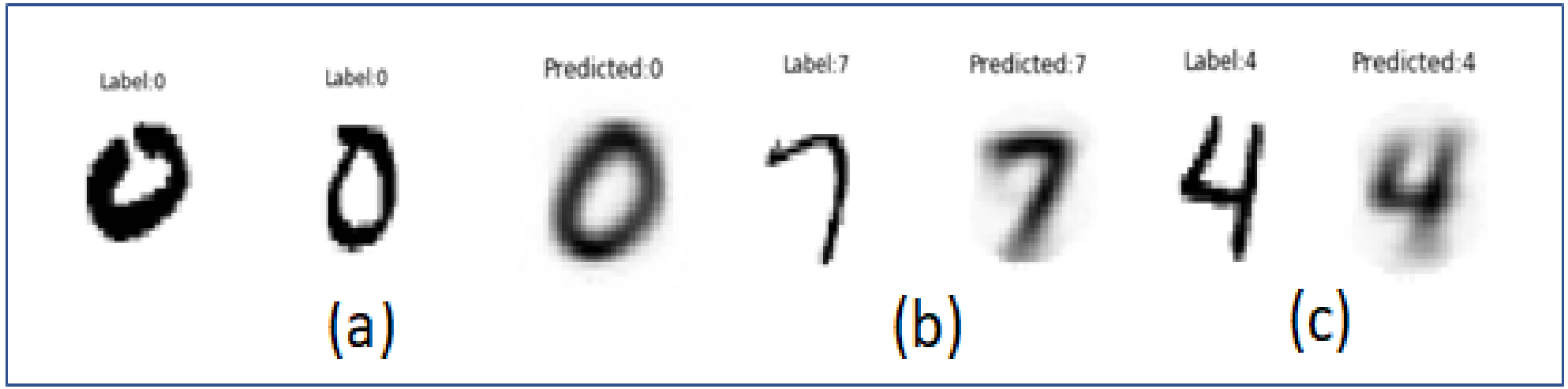
- CNN feature extraction on a digit image taken from MNIST dataset



  - Convolution layer : 3 filters of kernel size 10*10
  - ReLu activation layer
  - MaxPooling2D layer
  - Next Steps : Pass this digit image through CapsNet.

CapsNet Feature Extraction on MNIST Dataset

- Capsule network on MNIST dataset and passing a digit image through the layers of convolution.



Architecture of CapsNet

- Initial Convolution Layer
  - Convolution 2D layer : 256 filters, kernel size = 9*9, stride = 1, activation = ReLu

- Primary Capsule Layer
  - Convolution 2D layer : 256 filters, kernel size = 9*9, stride = 2
  - Reshape to generate capsules of 8 dim vectors.
  - Pass output vector through squash function.

Architecture of CapsNet

- Digit Capsule Layer
  - Initialize weight matrix for each capsule in lower layer.
  - Dynamic Routing – Learn the weights.

- Decoder Network
  - Boost the learned pose parameters.
  - Reconstruct the original image.
  - Input : Output from Digit Capsule Layer.

Outcomes/Conclusion

- The predictions by CapsNet are more accurate as in the predicted image for label “0”, the gap present in the original image gets refilled as compared to CNN.

- For further analysis, we added Dropout and Dense layers and tested our CNN network for 10 epochs with MNIST dataset. We also tested CapsNet with 10 epochs.

- It can be seen from the table as per data obtained from [2], CapsNet performed much better than CNN in predicting correct output.

Outcomes/Conclusion

Dataset	CNN	CapsNet
MNIST	99.2%	**99.30%**
Fashion-MNIST	83%	**89.8%**
CIFAR10	49.97%	**68.53%**
SVHN	87.43%	**91.06%**

Conclusion – Systems Management Use Case

- Use of CapsNet in these scenarios will reduce the error in augmenting the information on to the captured image from AR App.

References

- Rinat Mukhometzianov & Juan Carrilo (2018) “CapsNet comparative performance evaluation for image classification”.
- Prem Nair,Rohan Doshi & Stefan Keselj (2017) “ Pushing the Limits of Capsule Networks”.
- Understanding Hinton’s Capsule Networks by Max Pechyonkin.
- Performance Analysis of Deep Learning Algorithms by Team Research Nest
- Applied Deep Learning - Part 4: Convolutional Neural Networks by “Towards Data Science” community.

Questions?

Thank You

DELL EMC

Procedure

for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow 0$.
for r iterations **do**
    for all capsule i in layer l: $\mathbf{c}_i \leftarrow \texttt{softmax}(\mathbf{b}_i)$
    for all capsule j in layer $(l+1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$
    for all capsule j in layer $(l+1)$: $\mathbf{v}_j \leftarrow \texttt{squash}(\mathbf{s}_j)$
    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 $\mathbf{v}_j$

Dynamic Routing by Agreement

- Lower layer capsules -> layer, L -> let us denote a particular capsule as capsule, i
- High level capsules -> layer, L+1 -> let us denote a particular capsule as capsule, j
- Layer L capsules bet for high level entities and their bets works together to get the output of high level capsules.
- **Coupling Effect :** High level capsule sends feedback signals to low level capsules.
- Prediction of low level capsules for high level capsule should sum to one.

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

Systems Management Use Case : Use in AR App for identification & Component replacement in Ent. Servers

- Use of Augmented Reality(AR) is a ubiquitous choice to improve UX and less lead time for understanding any product
- There are AR based apps which provide precise realtime info on the server that needs attention and its health in a data center
- Challenge with these solutions are the image capture of the server(when opened) needs a particular orientation for the app logic/Algo to work as the internal layout of the server is heavily polarized.
- Use of CNN to process these captured images may lead to error in providing augmented information over the image captured(as described in this PPT)
- Evaluation of Deep Learning algorithms CNN and CapsNet for AR applications.