Capsule Networks (CapsNets) – A Comprehensive Theoretical Guide

Nasar Reddy Naru

23100228

Github: https://github.com/Nasarreddy903/Implementing-Capsule-Networks-with-PyTorch.git

Capsule Networks (CapsNets) are an advanced type of neural network architecture introduced by Geoffrey Hinton to address the shortcomings of Convolutional Neural Networks (CNNs), particularly their inability to efficiently encode spatial hierarchies and handle object transformations.

This guide covers:

- The **limitations of CNNs**
- **Capsule Networks architecture**
- The **encoder-decoder structure** in CapsNets
- **Mathematical foundations** of dynamic routing
- Applications and **advantages over CNNs**

1. Limitations of CNNs and the Need for Capsule Networks

Problems with CNNs

CNNs have been widely used for image classification and object detection, but they have several **limitations**:

1. **Loss of Spatial Information:**

- CNNs use **max pooling** to reduce feature map size, which discards **precise spatial relationships**.
- Example: If a CNN sees an eye and a nose, it may fail to recognize if the nose is above the eye.

2. **Lack of Viewpoint Invariance:**

- CNNs struggle with **rotations, scaling, and perspective changes**.
- They require **massive datasets** with various angles to generalize well.

3. **Insensitive to Object Hierarchy:**

- CNNs recognize **local features** independently and **fail to learn** how they relate to the overall object structure.

Why Capsule Networks?

Capsule Networks solve these problems by:

- Using **vectors** instead of scalars to represent features.
- Maintaining **part-whole relationships** of objects.
- Using **dynamic routing** instead of max pooling to better capture spatial hierarchies.

2. Capsule Networks – Core Concepts

🔹 **Capsules – The Fundamental Building Blocks**

A **Capsule** is a small group of neurons that encodes not just the presence of a feature but also its **orientation, scale, and position**.

- CNN neurons output a **single scalar** (e.g., "Is this an eye? 1 or 0").
- A **Capsule outputs a vector** (e.g., "Yes, it's an eye, and its orientation is [x, y, θ]").

🔹 **Properties of Capsules:**
✅ They output **probabilities + transformations** of features.
✅ They store **spatial hierarchies** between objects.
✅ They allow the model to generalize across **rotations and deformations**.

🔹 **Dynamic Routing – The Core of Capsule Networks**

CNNs use **fixed pooling**, but Capsule Networks use **dynamic routing**, which allows lower-level features to be sent to the correct high-level capsule.

Example:

- **An eye capsule should connect to a "face" capsule** but not to a "car" capsule.
- Routing dynamically **decides which connections are relevant** based on agreement.

🔹 **How It Works?**
Each lower-layer capsule sends its output to all higher-layer capsules, and the higher-layer capsule decides **how much to trust** each input.

- If many capsules agree on a transformation, the connection is strengthened.
- If they disagree, their contributions are reduced.

Mathematical Representation of Routing:

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

where:

- c_{ij} = routing coefficient
- b_{ij} = agreement score
- Softmax ensures weights sum to 1

3. Capsule Network Architecture

Capsule Networks consist of **three main layers**:

1. **Convolutional Layer:**
    - Extracts basic image features like edges and textures.
    - Functions as a traditional CNN but outputs to **primary capsules** instead of neurons.
2. **Primary Capsules:**
    - Each capsule represents a **part of an object** (e.g., an eye or nose).
    - Converts CNN features into capsule vectors.
    - Uses **squashing function** to normalize the vectors:

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

where s_j is the weighted sum of inputs to a capsule.

3. **Digit Capsules:**

- Represents whole objects (e.g., a full digit "7").
- Uses **dynamic routing** to determine **which primary capsules contribute most**.

4. **Reconstruction Decoder:**

- A separate **decoder network** reconstructs the input image to ensure **capsules encode meaningful information**.
- Helps prevent overfitting.

4. Encoder-Decoder Structure in Capsule Networks

Capsule Networks naturally follow an **encoder-decoder paradigm**:

● **Encoder (Feature Extraction & Representation)**

- Extracts hierarchical representations of input data.
- Composed of **convolutional layers** and **capsule layers**.
- Uses **dynamic routing** to preserve part-whole relationships.

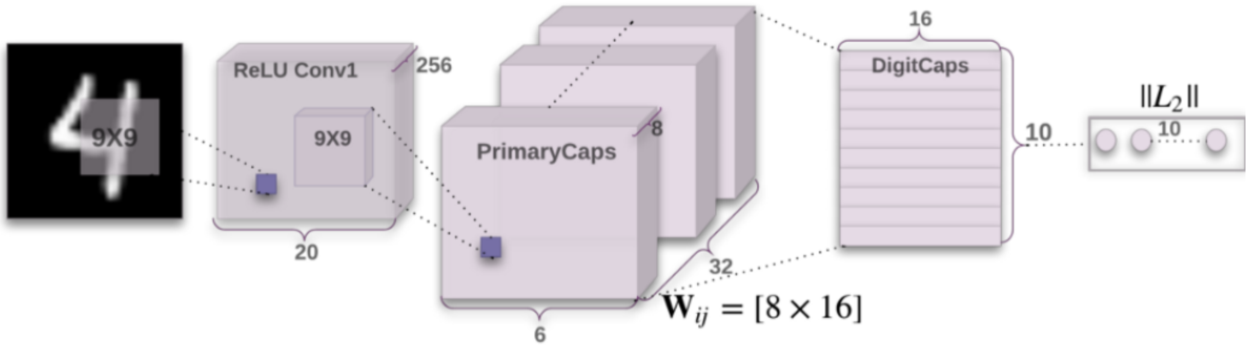

Figure 1: Encoder Network

Decoder (Image Reconstruction)

- Uses capsule outputs to reconstruct the original input image.
- Ensure that the capsules encode meaningful, spatially-aware representations.
- Typically, a **fully connected neural network** with activation functions like ReLU or Sigmoid.

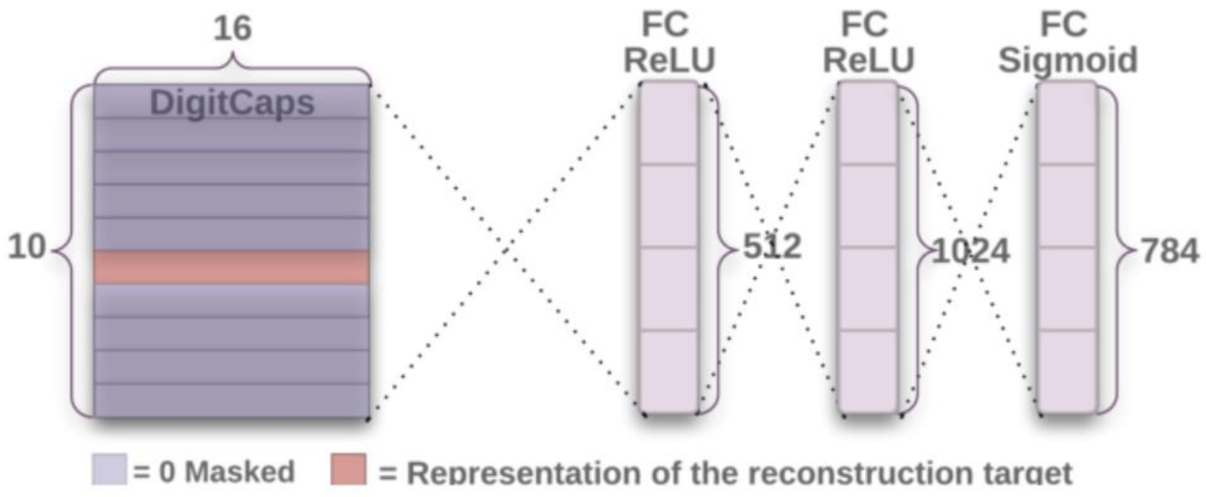

Figure 2: Decoder Network

Mathematical Representation of Reconstruction:

$$\hat{X} = f(W \cdot v + b)$$

where:

- v = capsule representation
- W = learned transformation matrix
- f = non-linear activation function
- $\hat{X}$ = reconstructed image

Why Include a Decoder?

- Forces the network to **learn interpretable representations**.
- Encourages capsules to **encode all necessary spatial information**.
- Acts as **regularization**, preventing overfitting.

5. Advantages of Capsule Networks Over CNNs

Feature	CNNs	Capsule Networks
Feature Representation	Scalars (single values)	Vectors (pose + activation)
Spatial Hierarchy Preservation	Lost in pooling	Preserved
Handling of Rotations & Viewpoint Changes	Poor	Strong
Training Data Requirement	High	Lower
Part-Whole Relationships	Ignored	Modeled via routing
Robustness to Noise	Moderate	High

Capsule Networks excel in:

- **Object recognition with fewer training samples**.
- **Better generalization across orientations and scales**.
- **Preserving hierarchical relationships between features**.

6. Challenges & Future Directions

Computational Complexity:

- More expensive than CNNs due to **dynamic routing iterations**.

Scalability Issues:

- Current CapsNets struggle with **large-scale datasets** like ImageNet.

Limited Hardware Support:

- Modern GPUs are optimized for CNN operations, not capsule computations.

Future Research Areas:

- Improving efficiency of dynamic routing.
- Application to 3D vision, medical imaging, NLP.
- Hybrid architecture (CapsNet + CNNs).

References

1. Afshar, P., Heidarian, S., Naderkhani, F., Oikonomou, A., Plataniotis, K.N. and Mohammadi, A. (2020) 'COVID-CAPS: A capsule network-based framework for identification of COVID-19 cases from X-ray images', *Pattern Recognition Letters*, 138, pp. 638-643. Available at: https://doi.org/10.1016/j.patrec.2020.09.010 (Accessed: 27 March 2025).

2. Haq, M.U., Sethi, M.A.J. and Rehman, A.U. (2023) 'Capsule Network with Its Limitation, Modification, and Applications—A Survey', *Machine Learning and Knowledge Extraction*, 5(3), pp. 891-921. Available at: https://doi.org/10.3390/make5030047 (Accessed: 27 March 2025).

3. Xi, E., Bing, S. and Jin, Y. (2017) 'Capsule Network Performance on Complex Data', *arXiv preprint*, arXiv:1712.03480. Available at: https://doi.org/10.48550/arXiv.1712.03480 (Accessed: 27 March 2025).