# **VideoCapsuleNet: A Simplified Network for Action Detection (Summary)**

## Abstract

The main subject of the paper is **Action Detection**. The paper mainly used **Capsules** in their network to address this problem. Their network results **pixel-wise action segmentation** and **action classification**. The network is a generalized from 2D to 3D and that issued a problem of increasing the network parameters then the network has been a computationally expensive. They addressed the previous issue with **Capsule-Pooling.**

## Introduction

**Capsules** is one of the main parts used in VideoCapsuleNet, and we can define Capsules as a group of neurons which can model different or parts of entities. The network relies on **routing-by-agreement** as a routing algorithm between capsules. One of the advantages of the capsules is that **viewpoint invariant.** The network also used 3D convolutions. In general, the network aims to learn the **semantic information** necessary for action detection.

## Generalizing Capsules to higher dimensional inputs

Each capsule consists of **2 main units**, the **pose matrix (M)** and the **activation probability (a).** The pose matrix contains **the instantiation parameters or the properties of the entity**. On the other hand, the activation probability is a **scalar between (0-1)** that present **the existence of the entity**. **Votes** between each 2 capsules are casting **using the transformation matrix** as follow:

This vote **employing the transformation matrix Mj for capsule** j in the next layer.

Generally, in Capsule networks, they **reduce** the number of computations by **only compute votes for capsules within a local respective field**. But in this paper, they introduced **2 new methods**, first one: they **share transformation matrice**s between capsules of the same type. Since they model the same entity so their votes should not vary based on their positions. Second one: they only apply **transformation matrix on the mean of the capsules in the respective field of each capsule type**.

## Network Architecture

The network started with **input** of size **8 frames of 112 as width size and 112 as height size**. Then **6 convolutions of 3x3x3** with **ReLU as an activation function**. Then **a 512 feature maps as a result of the convolution with 8x28x28 dimensions**. There is **2 convolution capsule layers**, first one **of 32 capsule types** and each capsule consist **of 4x4 pose matrices**. The activation obtained by applying **3x9x9 convolution operation**. And the activation functions **are ReLU and sigmoid respectively**. **Second** convolution capsule layer consist **of 32 capsule types with 3x5x5 respective** fields and **1x2x2 stride.** The second convolution capsule **layer followed with fully connected** capsule layer connected **to c capsules** as **c is the number of action classe**s. All capsules of the **same type are sharing the same transformation** matrices.

To **preserve** the information about the **convolution capsule’s location,** **coordinate addition is applied**. At each pixel capsule coordinate is added **(time, row, column)** to **the final three entries of the vote matrix.**

**Frame-level localization is obtained by masking procedure.**

During the **training**, **all pose matrices** are **masked** by setting their values to **0** **except the one corresponding to the ground-truth.**

At **test** time**, all classes except the one with the largest activation probability are masked**. **Class capsules’ poses** are **fed** to **fully connected layer produces 4x8x8** feature maps. With up-scaling using **transpose convolution** result **8x112x112 localization maps**. **Skip connections** are **used** to **fine** **positional** **information**.

The **loss** **function** consist of 2 parts **classification** **loss** and **localization** **loss**.