ST-GCN Review

Zhexuan GU

2023.08.16

ST-GCN is a special variation of Graph Convolution Neural Network which makes use of innate human body structure designed for human action recognition and pose estimation.

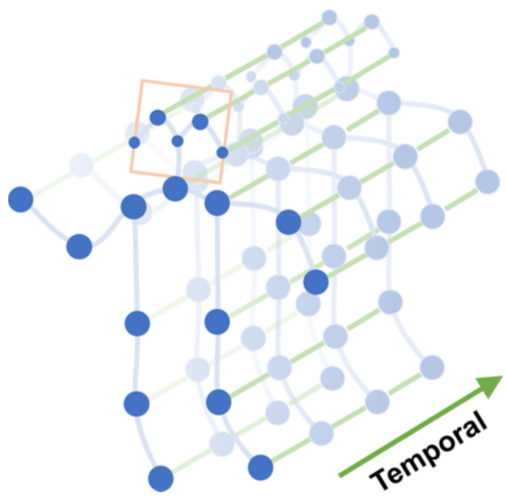

Figure 1: Human Body Structure

As we can see in Figure 1, the blue dots stand for the human body joints while the connected edges represent the human body segments. The inter-frame edges connect the same joints between consecutive frames.

An undirected graph $G = (V, E)$ is contructed to fit the model. In this graph, the node set $V = \{v_{ti}|t = 1, \ldots, T, i = 1, \ldots, N\}$ includes all the joints in a skeleton sequence. T is total frames and N is total joints. Each node has a feature vector $f \in \mathbb{R}^c$.

It should not be suprising that our input is expected to have the shape $B \times C \times T \times N$. The parameter B here is just the batch size.

Figure 2 shows the core operations of ST-GCN model. Now I'll briefly illustrate the pipelines.

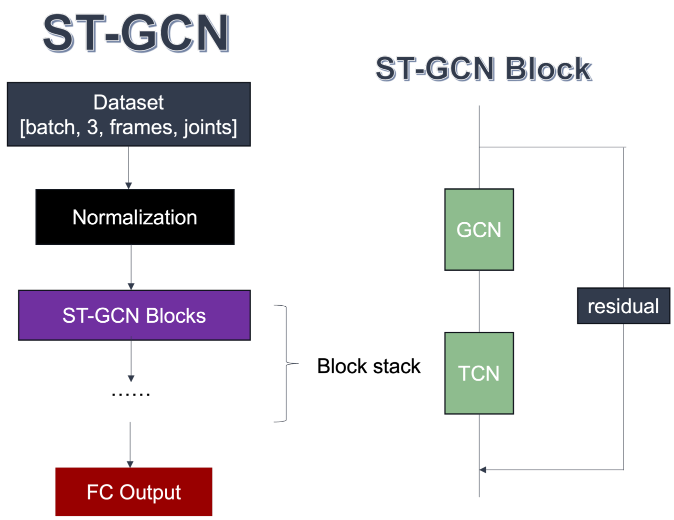

Figure 2: ST-GCN Pipeline

1. Given a dataset, no matter whether you do the feature engineering or data cleaning, make sure that the input data fed to the model has the shape talked above.

2. Use Batch Normalization layer to normalize the data.

3. Stack several ST-GCN blocks together to extract meaning features. A ST-GCN block is mainly consisted of a simple GCN Block, a Temporal Convolution Network Block and a residual connection.

   (a) For a GCN block, it firstly uses 1×1 conv block to extract high dimensional features of each joint at each frame. Then graph adjacency matrix is applied to aggregate features for each node.

   (b) A TCN block is used to swap meaningful information along temporal axis. Since a feature map is in shape $T \times N$, apparently a convolutional layer whose kernel has size $(t\_kernel, 1)$ can do this job.

4. Use the features to classify the action or pose.

Notably, graph partition strategy extends the naive GCN to help the model have a better performance.

Figure 3 demonstrates the strategy visually. For the most original GCN which corresponds to the Figure 3 (b), the center node get features from its neighbor nodes with same weight. Figure 3 (c) partition the neighbor set according to the nodes' distance $d(, v_{ti})$ to the root node v_{ti}. Figure 3 (d) strategy obtains the best result. This strategy generates 3 different subsets of nodes as well as 3 adjacency matrices:

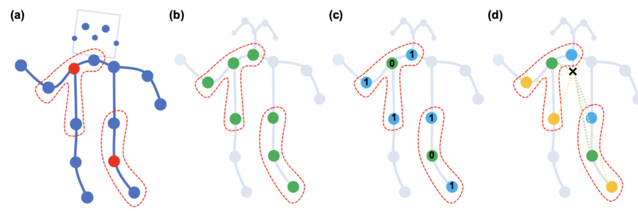

Figure 3: graph partition strategy

1. the root node itself, a diagonal adjacency matrix;
2. centripetal group: the neighboring nodes that are closer to the gravity center of the skeleton than the root node;
3. otherwise the centrifugal group.

In implementation, we set the expected output channels of the GCN block as $num\_of\_adj\_matrices \times C_{out}$ when we are extracting high dimensional features. Undoutedly, we can reshape the output into shape $B \times num\_of\_adj\_matrices \times C_{out} \times T \times N$. Then we can multiply each adacency matrix with its corresponding feature map and finally add all the features together.