PointNeuron: 3D Neuron Reconstruction via Geometry and Topology Learning of Point Clouds Supplementary Material

1. Network Architecture Details

As mentioned in the main paper, our proposed pipeline for the 3D neuron reconstruction task, *PointNeuron*, is composed of two major modules: a skeleton prediction module and a connectivity prediction module. Figure 1 shows the details of our designed network architecture.

The encoder for the neuron point cloud input, based on DGCNN, has 3 EdgeConv blocks. Each EdgeConv block is used to compute edge features for each point with 20 nearest neighboring points, which has a MLP network $\{c_1, .., c_n\}$ where c_i is the number of output channels of the layer i, and a max-pooling layer to generate the locality-aware feature maps. Each MLP layer is followed by a batch normalization and a LeakyReLU activation function with a negative slope of 0.2. These feature maps with multi-scaled

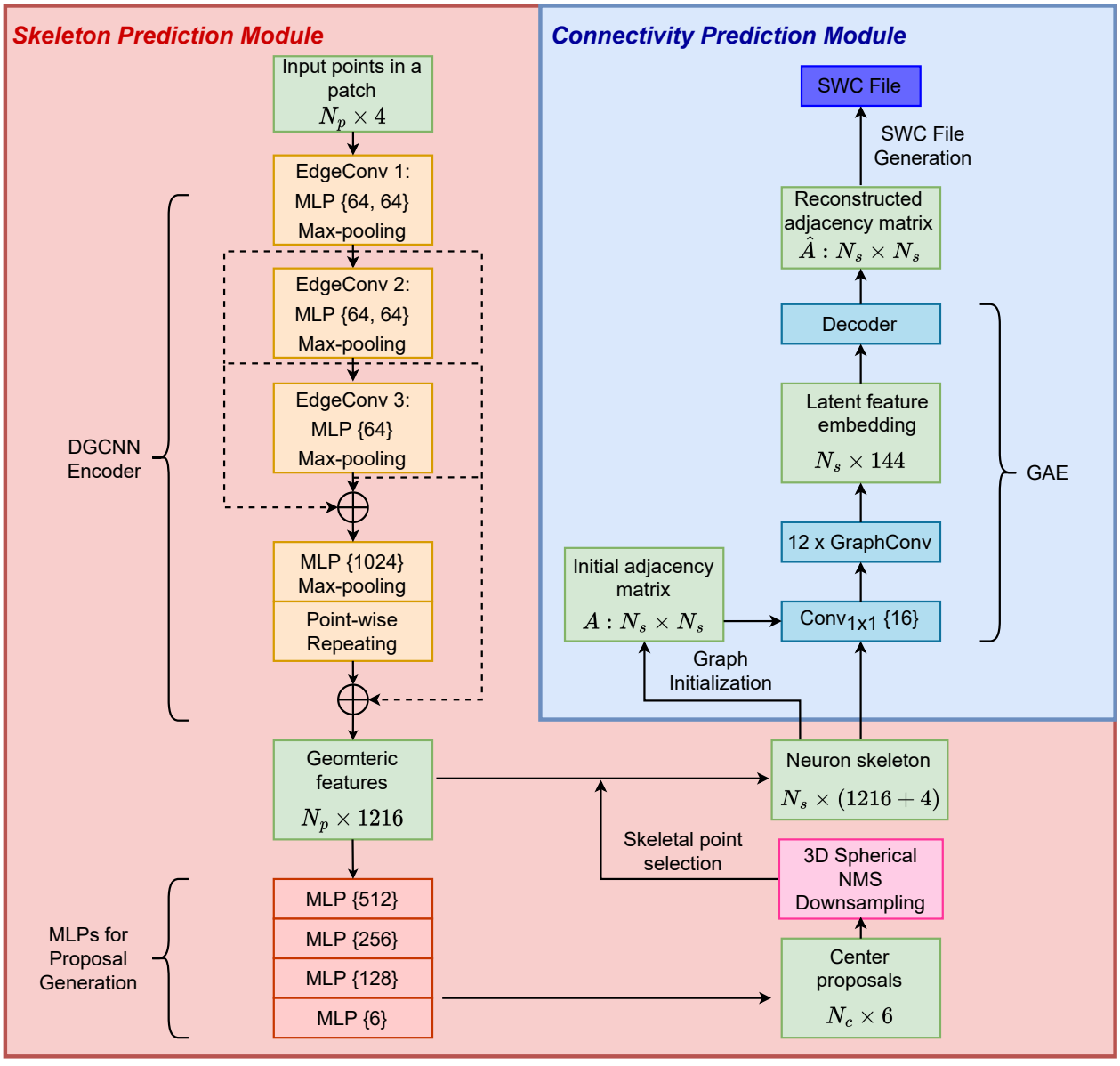

Figure 1. Overall *PointNeuron* architecture. ⊕ means the channel-wise concatenation.

receptive fields are concatenated as one through residual connections. As per the main paper, the proposal generation is based on the shared MLP layers to process geometric features, which outputs the center proposals of a 6-channel depth dimension that contains two objectness score values, a radius value, and three XYZ coordinate offset values. The last MLP layer in the proposal generation is not followed by a batch normalization and a ReLU activation function. After 3D spherical NMS downsampling, we could select the skeletal points (or sampled critical center proposals) to constitute the compact neuron skeleton.

The geometric features of skeletal points are concatenated with the point geometric predictions of Cartesian coordinates and radii, which are used as the graph node features in the connectivity prediction module to produce the relationships among skeletal points. The graph adjacency matrix is initialized by a ground truth SWC file when training, or by a traditional tracing algorithm when inferring. Each GraphConv is followed by a batch normalization and a ReLU activation function, and the output feature channels of 12 GraphConv blocks in GAE are $\{32, 32, 48, 64, 64, 80, 96, 96, 102, 128, 128, 144\}$ to compress the node features into the latent embedding. The reconstructed adjacency matrix is obtained through the inner product decoder. In the end, based on the predicted neuron geometric skeleton information and point-wise topology relationships, we design a recursive function to generate the target SWC file.