Model architecture:

The model is structured as a convolutional autoencoder, tailored explicitly for volumetric data segmentation. It utilizes a series of 3D convolutional, group normalization, and ReLU activation layers to encode the input into a compressed latent representation and decode it into a segmented output. The architecture is divided into two main parts: the Encoder and the Decoder.

Encoder

The encoder is responsible for progressively compressing the input data, extracting and encoding salient features necessary for segmentation:

1. Initial Convolutional Layer:
    - ❖ Configuration: A 3D convolutional layer with 64 filters, each with a kernel size of 3x3x3 and a stride of 2.
    - ❖ Purpose: This layer takes the 4-channel input (e.g., different MRI modalities or time points) and applies a stride of 2 to reduce the dimensionality by half. It effectively captures basic features from the raw input while reducing the computational load for subsequent layers.
2. Dropout: Dropout is set with a rate of 0.2, meaning that each feature map has a 20% chance of being dropped during training.

3. Group Normalisation:
    - Implementation: Follows the initial convolution to normalize the data within small groups of channels, stabilizing the learning process by reducing internal covariate shifts.
4. ReLU Activation:
    - Function: Applied after each convolutional and normalization step to introduce non-linearity, allowing the model to learn more complex patterns.
5. Residual Blocks:
    - Structure: Each block consists of two convolutional layers with intermediate group normalization and ReLU activation. These are designed to allow the training of deeper networks by addressing issues related to vanishing gradients through skip connections.
    - Downsampling: When the stride is set to 2 or when increasing the number of filters, a downsampling layer (a convolution with stride 2) adjusts the dimensions to match the output of the convolutions, preserving the ability for element-wise addition with the input of the block.

Decoder

The decoder reconstructs the segmented output from the encoded data using a series of upsampling blocks and a final convolutional layer:

1. UpSample Blocks:

- Upsampling Mechanism: Utilises transposed 3D convolutions to double the dimensions of the feature maps, gradually restoring the data to its original dimensions.
- Refinement: Each upsampling step is followed by a standard convolutional layer to refine the upsampled features, ensuring that the output maintains spatial accuracy.

2. Final Convolutional Layer:
    - Setup: A single 3D convolutional layer with three filters matching the number of expected segmentation classes.
    - Output: Produces three channels where each channel corresponds to a class probability map, indicating the likelihood of each voxel belonging to a particular class.