* Overview of problem:
  + Current cancer treatments involve the use of x-rays to target and destroy cancer cells within the patient's body.
  + These x-rays need to be carefully target so as not to damage any of the surrounding tissue
  + Currently this is accomplished using MRI imaging and having a doctor manually trace out the surrounding organs, a tedious and time consuming process
  + We aim to automate this process through the use of convolutional neural networks. The image can be automatically segmented in a matter of minutes removing the risk of human error and improving the quality of life of patients by reducing treatment times
* Methods:
  + Brief preprocessing
    - During MRI imaging many slices are taken. Not all of these slices contain the stomach, small intestine, or large intestine. Therefore, the data needs to be pruned before it can be trained on. This was accomplished through the use of a standard [?] neural network to classify images into 2 categories: those containing the desired organs, and those without.
    - [Include data for classifier]
  + Segmentation
    - Once we have pruned the data it is passed into our fully convolutional neural network.
    - This stage passes several hundred convolutional filters over the 3 consecutive images until a final feature extraction layer is reached.
    - The use of three consecutive images was used as we don't expect the segmentation to change dramatically between slices and the performance of our NN should improve if it is able to learn this
    - The final feature extraction layer is then passed to transposed convolution in the upscaling stage. Also during this step we performed a skip connection by passing in the corresponding downscaled layers image
    - In the end the output is a single binary image containing our image mask.
  + Loss function rationale
    - We opted to use a combination focal and tversky loss function.
      * Focal loss minimizes the pixel-wise error between the masks
        + Works well with images that are hard to classify since it down weights easy examples
      * Tversky Loss is a generalized dice loss and it measures the overlap between the true mask and the predicted mask
        + Handles class imbalance (which we have as not all images contain all classes)
        + Doesn’t work well with images that are hard to classify
* Results:
  + In stage 1 our classifier was able to correctly select all images with useful data and only include a small number of false positives, greatly improving our training times.
  + Our full segmentation architecture had an overall loss of [insert relevant loss value], a marked improvement over the other architectures we tried
  + Other metrics we have gauged our model by are [Talk about additional metrics] which represent the [Talk about what they represent]
* Potential improvement:
  + Increasing Model complexity - Using more complex/sophisticated pretrained network such as replacing ResNet50 with ResNet110 (50 layers → 110 layers) and/or replacing vit\_base\_patch16\_224 with vit\_large\_patch16\_384 (increase transformer layers from 12 → 24 & handle larger dimension image data 224 → 338)
  + Improving the pipeline - Adding a smaller segmentation model to obtain a single mask for all 3 interest organs, and feeding this mask as additional (4th layer of the 2.5 Dimension axis) information of the input image to the regular segmentation network.
  + Loss function - Further tuning the hyperparameter of loss function with more experiment runs that could better pick out the organs that have smaller masks/regions. Or even adding a new loss function - boundary loss (aka Hausdorff distance loss) to better predict the boundary of the organs.
  + Changing the dataset - working with a 3D organ dataset instead of 2.5D slices to preserve more spatial information if we have enough computing power/resources