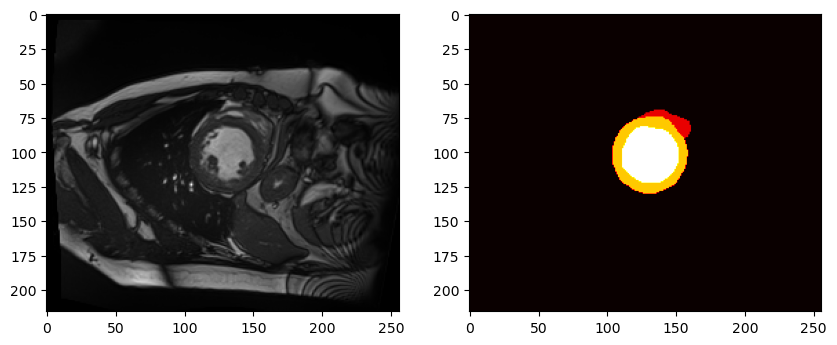
**Progress Report Week 3**

**Group Name: Anhinga**

**Part 1: Cardiac MRI image segmentation. Data exploration, preprocessing, and data augmentation**

**1. Data exploration**

The subset of ACDC dataset that we use in these exercises contains 40 3D images (28 for training, 4 for validation, and 8 for test). Figure 1 shows one frame of a 3D image of this dataset and its corresponding segmentation that consists of 3 classes (1, 2, 3).



**Figure 1:** Example from first image of train loader showing slice 5 of image and corresponding label image.

Although small, this dataset is heterogeneous. The image voxel resolutions ranges from 1.37 to 1.88 mm. (y-axis, first dimension), from 1.37 to 1.88 mm. (x-axis, second dimension), and from 10 to 10 mm. (z-axis, third dimension). The image sizes of the ranges from 184 to 256 voxels (y-axis, first dimension), from 8 to 11 voxels (x-axis, second dimension), and from 216 to 256 voxels (z-axis, third dimension).

**2. Preprocessing**

During training, we standardized the voxel resolution to (1.47, 1.47, 10) mm. with the *Spacingd* transform. This voxel resolution is now set to the median voxel resolution, which is chosen since the distribution of voxel resolution is left-screwed – hence the median takes this into account . While the image is resized using bilinear interpolation, the label is resized using nearest neighbor interpolation. If, on the contrary, we use bilinear interpolation to resize the label, the labels would not be interpolated to discrete values.

During training, we standardized the image size to 128 x 128 x 10 voxels with *ResizeWithPadOrCrop*. This image size was chosen because this size downscales the image to a reasonable size without too much information loss.

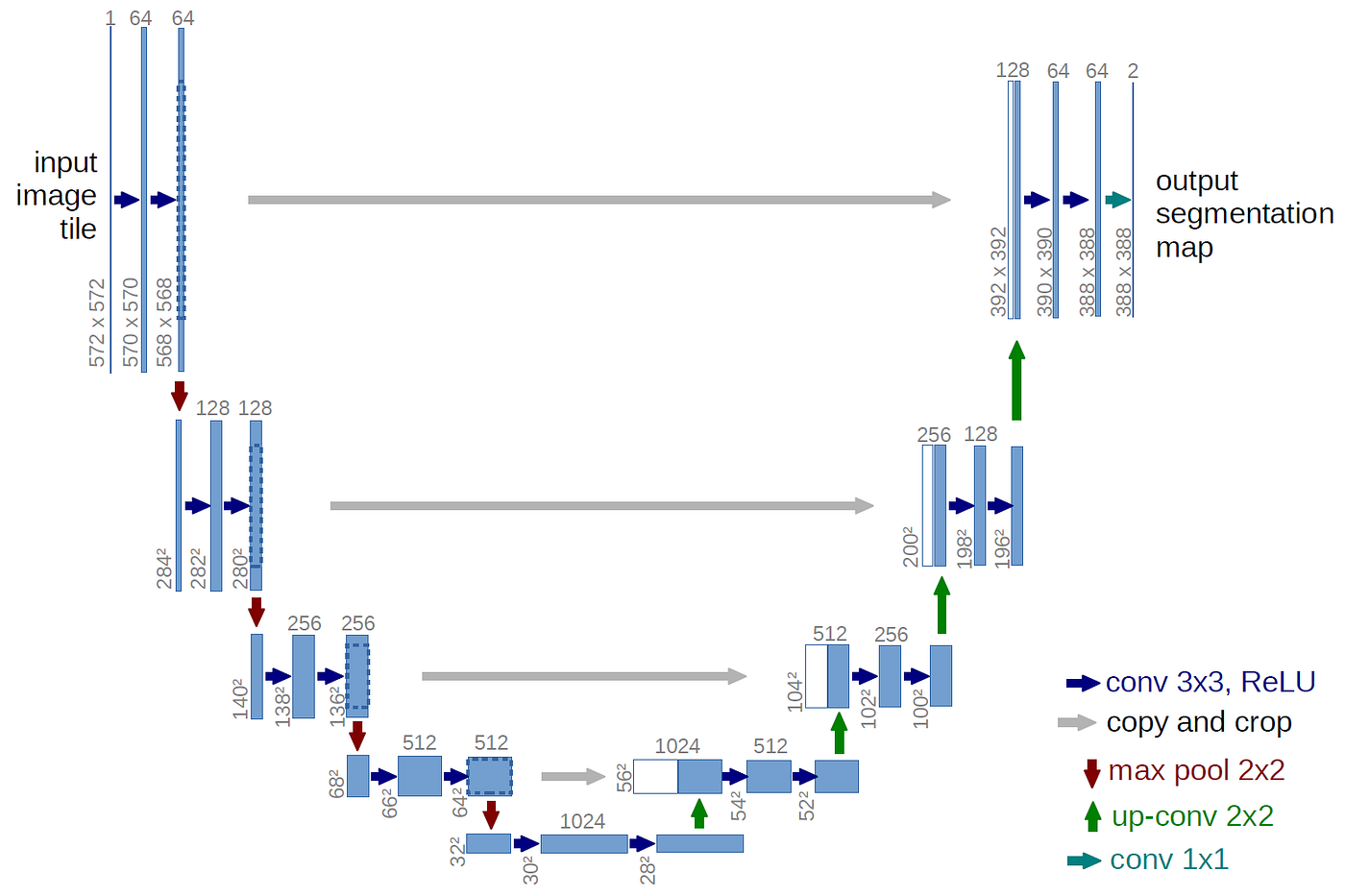
**3. Data augmentation**

Since our training set is very small, we applied data augmentation. Specifically, we applied RemovedSmallObjectsd transform with the parameters min\_size=64 on the label, RandGaussianNoised on image with parameters p=0.5 and std=0.1.

Finally, since we trained a 2D network, we sampled random 2D patches from the 3D images with *RandSpatialCropd*.

**4. Combining all Data augmentation and preprocessing transforms**

After combining all the transforms, the images that will be fed into the neural network shown in Figure 2.

**Figure 2:** Example of an Unet – which is the same structure utilized in this exercise.

**Part 2: Training, Validation, and Inference**

**5. Training a neural network**

We optimized a DynUNet with Dice loss function for 50 epochs and batch size of 4. The optimization was done with Adam optimizer with a learning rate of 1e-3. During training, we keep track of two measurements: Dice loss that is computed at the end of every epoch, and dice coefficients that is computed at validation time. The last Dice coefficients we computed at validation time was: ([0.7842576 0.76700699 0.86222601]).

**6. Training and validation curves**

Figure 3 shows two plots: On the left, we observe that the train loss decreases over time, and, on the right, we observed that the Dice coefficients of each of the 3 classes increases over time.

A graph of different colored lines

Description automatically generated **Figure 3:**Left: Training loss of DynUNet. It can be observed that the training loss is decreasing, hence the model is fitting to the training data. Right: The Dice coefficients can all be seen to increase over time. This indicates that the overlap between the true labels and the predicted labels are increasing.

**7. Inference at test time**

Finally, we evaluated our trained model on the test set, where we obtained a final average dice coefficient of 0.81.

**8. Visualize the results**

Figure 4 shows a few slices of the test set together with their corresponding labels and the predictions.

A collage of images of a person's body

Description automatically generated **Figure 4:** The figure above shows the results of the segmentation using slice 1,2,3,4 of the first test image. Row1: Input image. Row2: Predicted labels. Row3:True labels. Difference between target and prediction.