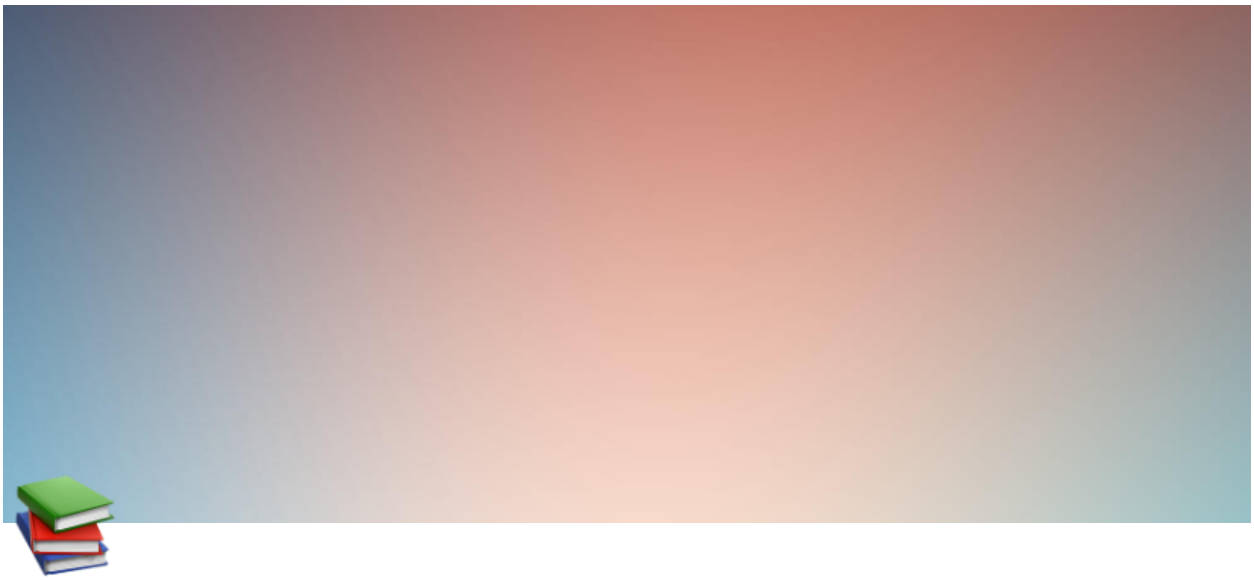


Reproducing the TransUNet Model: Task 1

1. Objective

The task involved reproducing the performance of the TransUNet model for medical image segmentation using two datasets: the ACDC dataset and the CAMUS dataset. The performance of the model was evaluated using these following metrics:

- DICE Score
- Hausdorff Distance (95th percentile)
- Intersection over Union (IoU) (something I thought could be relevant as the overlap ratio is be useful)

2. Background

Before starting this project, I spent time thoroughly studying the TransUNet model to understand its underlying principles and architecture. This includes reading the original paper, analyzing its procedure, and reviewing related concepts on medical image segmentation and transformers online. This caused me to take longer to

fully learn the concept before trying to reproduce it for these two datasets, so apologies for the delay.

TransUNet

TransUNet is a hybrid architecture that combines the strengths of transformers and U-Net for medical image segmentation. The key components include:

- **Vision Transformer (ViT)**: A pre-trained ViT is used as the encoder to capture global context and long-range dependencies.
- **U-Net Decoder**: Convolutional layers are employed in the decoder to recover spatial details and generate segmentation maps.

This approach bridges the gap between the global feature extraction of transformers and the spatial localization abilities of convolutional networks.

As suggested in the github repository I used the R50_ViT_B_16 pretrained model for both datasets. For the ACDC Dataset, I used the resources in the repository to help me run the model for the dataset. However, for CAMUS, I tried to start from scratch.

3. Datasets

ACDC Dataset:

- Cardiac MR images for multi-class segmentation (e.g., left ventricle, right ventricle, myocardium).
- Results were successfully reproduced.

CAMUS Dataset:

- Echocardiographic images for segmentation of the left ventricle (LV), myocardium (MYO), and left atrium (LA).
- Data consisted of images and ground truth labels for two views: 2CH and 4CH.

4. Implementation Steps

A. Preprocessing

- Resampled images to uniform spacing.
- Normalized intensity values.
- Split data into training, validation, and testing sets as per the `subgroup_training.txt`, `subgroup_validation.txt`, and `subgroup_testing.txt` files provided with the dataset.
- Saved preprocessed data in `.npy` format for efficient loading.
- Code in `preprocessing script.py` .

B. Model Implementation

- Utilized the pre-trained Vision Transformer (ViT) weights from Google's repository:
- Adjusted the `vit_seg_configs.py` and `vit_seg_modeling.py` from the TransUNet repository to support the datasets and load the model and the data.
- Defined specific configurations for segmentation:
  - Input size: (224, 224)
  - Classes: 3 (LV, MYO, LA)

C. Training

- Implemented `train.py` using in both:
  - Cross-Entropy Loss for multi-class segmentation.
  - Adam optimizer with learning rate scheduling.
  - Data augmentation strategies (e.g., random rotations, flips).

D. Testing

- Evaluated model predictions on the test set.
- Saved predicted segmentation maps and corresponding ground truth for visualization.
  (Separate script for the ACDC dataset as I was testing. For the CAMUS I integrated this in the test script itself and saved the png file directly in the output directory.)

- Calculated evaluation metrics.

5. Evaluation Metrics

→ DICE Score: Measures the overlap between predicted and ground truth masks.

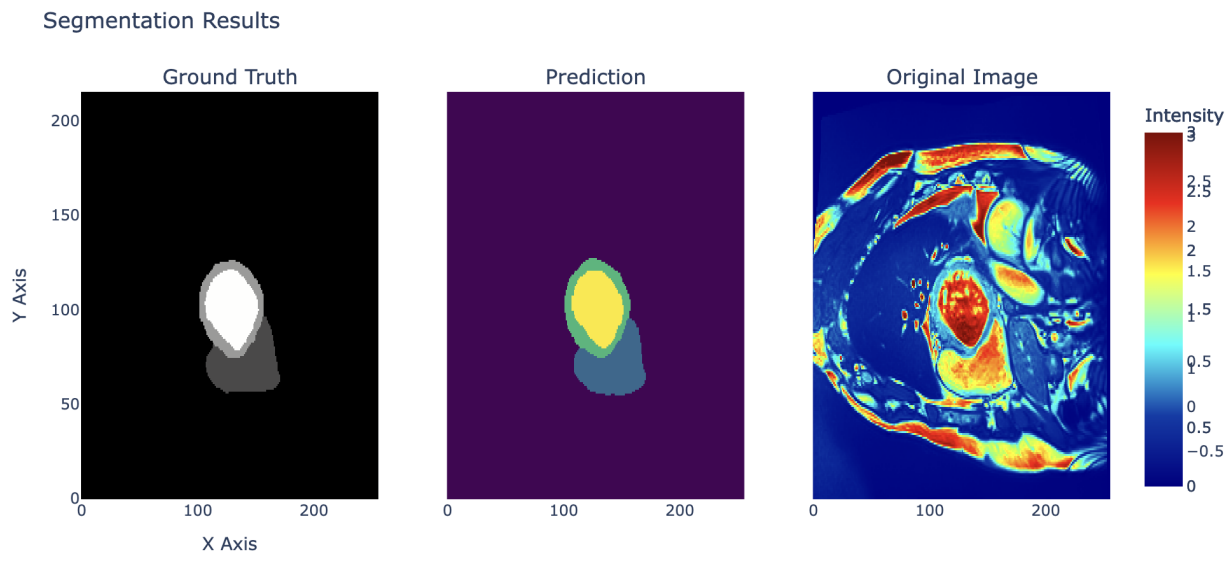
→ Hausdorff Distance: Measures the distance between the boundary points of the prediction and ground truth masks.

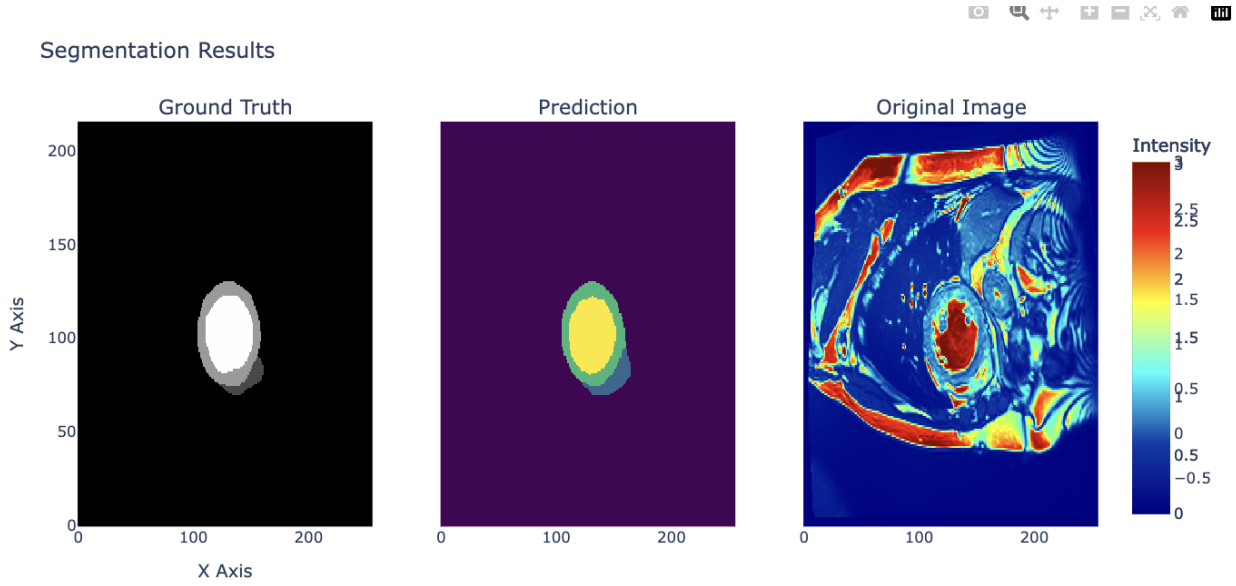
→ Intersection over Union: Measures the overlap ratio.

6. Results

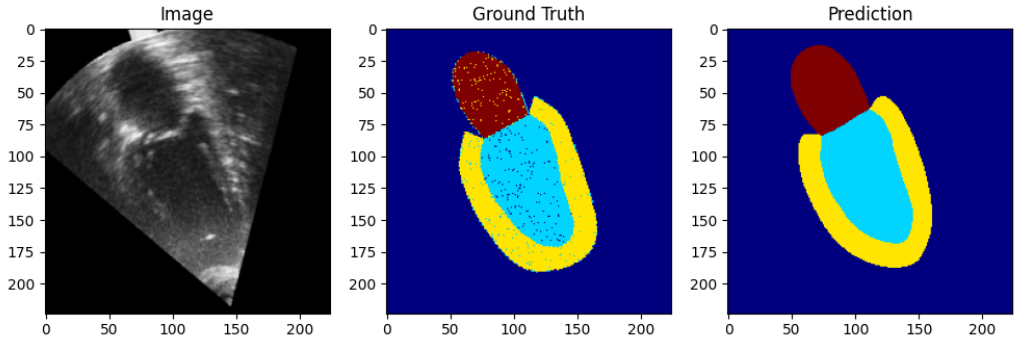
	Dice	Hausdorff	IoU
ACDC	0.897481	2.239824	0.821313
CAMUS	0.9128	17.9379	0.8451

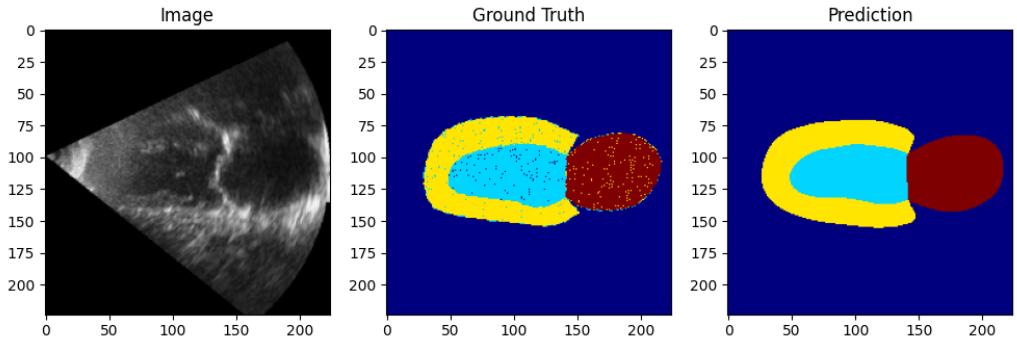
ACDC Visualisation results:

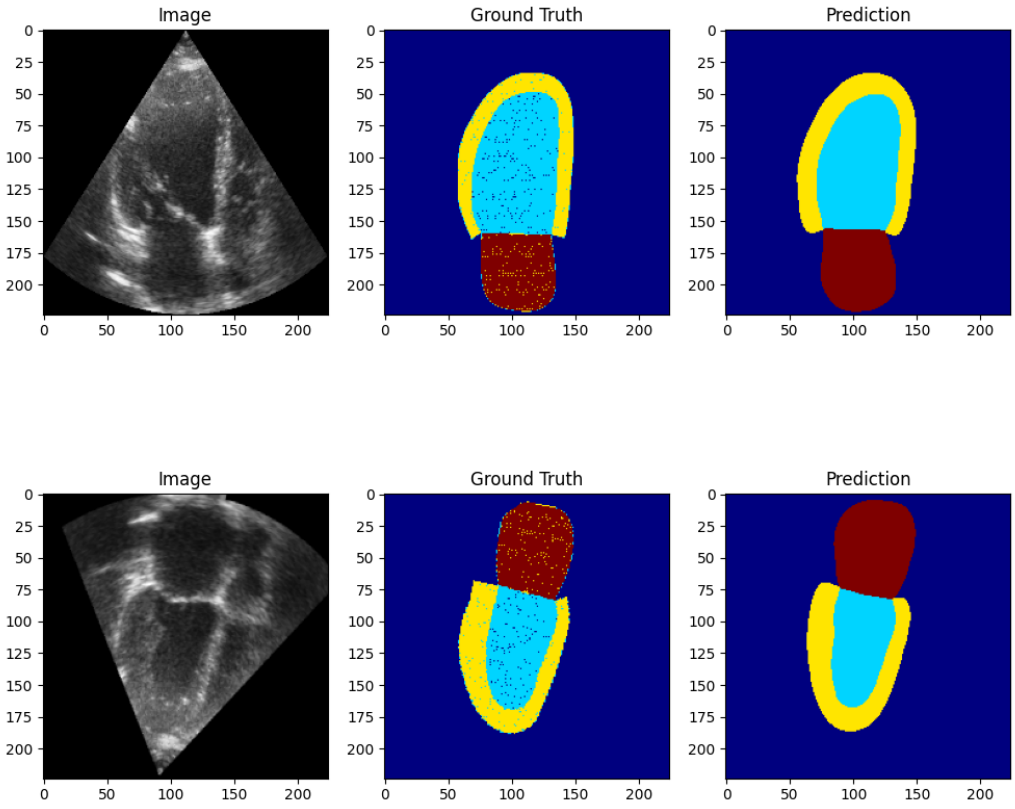




CAMUS Visualisation results:







results acdc

results camus

github links to the code:

ACDC: https://github.com/gavkujo/ACDC_TransUNet_test

CAMUS: https://github.com/gavkujo/CAMUS_TransUNet_test