Semi-Supervised Segmentation of Cardiac MRI Images Using Self-Training and Tversky Focal Loss

Joaquim Jusseau, Paul Merceur, Flavian Theurel
École de Technologie Supérieure
MTI865 - Deep Learning for Computer Vision
Instructor: Christian Desrosiers

Fall 2024

Abstract

Medical image segmentation is crucial for diagnosing and treating cardiovascular diseases. However, obtaining large labeled datasets is challenging due to the need for expert annotations. This project explores a semi-supervised approach using self-training to improve segmentation of cardiac MRI images from the ACDC dataset. We enhance a UNet-based architecture with attention mechanisms, residual connections, and dilated convolutions. The Tversky Focal Loss function is employed to address class imbalance and emphasize hard-to-classify regions. Our results demonstrate that leveraging unlabeled data through self-training, combined with an advanced loss function and architecture, significantly improves segmentation performance.

1 Introduction

Accurate segmentation of cardiac structures in MRI images is essential for assessing heart function and planning clinical interventions. However, traditional supervised learning approaches require large labeled datasets, which are challenging to obtain in medical imaging due to the time-consuming and expertise-intensive annotation process.

To address these challenges, semi-supervised learning offers a promising solution by leveraging both labeled and unlabeled data to enhance model performance. In this project, we propose a semi-supervised approach to improve the segmentation of cardiac MRI images. Our method incorporates unlabeled data through self-training and enhances a UNet-based architecture with attention mechanisms and other advanced features.

To tackle the issue of class imbalance, a common challenge in medical imaging datasets, we adopt the Tversky Focal Loss function, which is particularly effective in such scenarios. This approach aims to deliver accurate and robust segmentation of cardiac MRI images.

2 Related Work

The UNet architecture [1] is widely adopted in medical image segmentation for its ability to capture both local and global context through its encoder-decoder structure and skip connections. However, standard UNet may struggle with class imbalance and small target structures.

Loss functions like Cross-Entropy and Dice Loss are commonly used but may not adequately handle class imbalance. The Tversky Loss [2] generalizes the Dice coefficient, allowing control over false positives and negatives, making it suitable for imbalanced datasets. The Focal Tversky Loss [3] introduces a focusing parameter to emphasize hard-to-classify regions.

Self-training is a simple yet effective semi-supervised learning method where a model trained on labeled data generates pseudo-labels for unlabeled data, which are then used to retrain the model [4]. This approach can improve performance when labeled data is scarce.

3 Methodology

3.1 Data

3.1.1 Dataset Description

We use the Automated Cardiac Diagnosis Challenge (ACDC) dataset, which includes cardiac MRI images from multiple patients. Each patient has images for two phases: end-diastole and end-systole, with 10-13 slices per phase. Segmentation masks label four classes:

- 0: Background
- 1: Right Ventricle
- 2: Myocardium
- 3: Left Ventricle

3.1.2 Dataset Split

To prevent data leakage, we ensure patients are not shared between sets.

- **Training Set:**
  - Labeled Images: 204
  - Unlabeled Images: 1004
- **Validation Set:** 74 images

3.2 Data Preprocessing

3.2.1 Data Augmentation

To enhance data diversity and reduce the risk of overfitting, we employ a set of random data augmentation techniques during training. These augmentations include horizontal and vertical flips, which randomly invert the image and mask along their respective axes, introducing variability in spatial orientation. Additionally, we apply random rotations by discrete angles, such as 0°, 90°, 180°, and 270°. These rotations simulate different perspectives, helping the model learn rotational invariance.

3.2.2 Normalization

We normalize image intensities using histogram equalization to standardize brightness and contrast across images.

3.3 Model Architecture

3.3.1 Overview

Our model builds upon the standard UNet architecture by integrating several advanced components to enhance its performance. These additions allow the model to focus on more relevant features, improve training efficiency, and reduce overfitting:

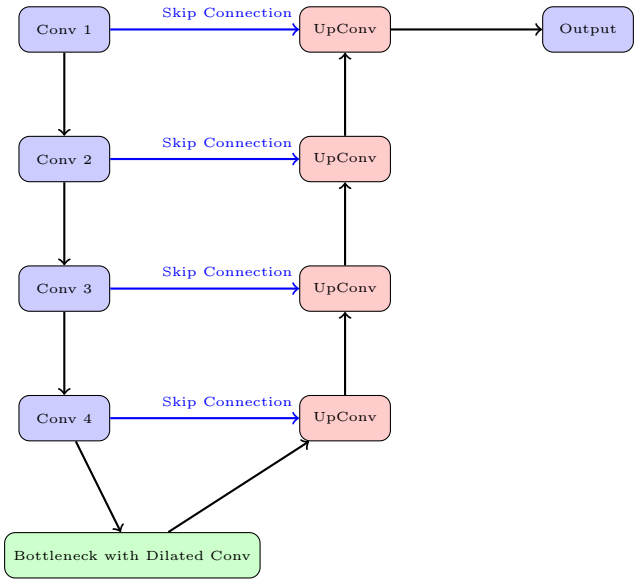
- **Attention Blocks:** Channel and spatial attention mechanisms are used to focus on the most relevant features, improving segmentation accuracy.
- **Residual Connections:** These connections help improve gradient flow and allow the network to be deeper without the risk of vanishing gradients.
- **Dilated Convolutions:** These convolutions capture a broader context at the bottleneck layer, expanding the receptive field without losing resolution [6].

- **Dropout:** Applied in deeper layers to mitigate overfitting and improve generalization.
- **Kaiming Initialization:** This weight initialization technique facilitates efficient convergence during training by preventing gradient issues [5].

3.3.2 Architecture Details

The architecture follows a traditional UNet structure, with several modifications for enhanced performance. The following describes the components in detail:

1. **Encoder:** Consists of four convolutional blocks with increasing filter sizes, residual connection, followed by max-pooling layers to downsample the image and extract hierarchical features.
2. **Bottleneck:** Dilated convolutions are applied to capture a wider context, enhancing the model's ability to process complex features in the deeper layers.
3. **Decoder:** Utilizes transposed convolutions for upsampling, spatial attention, and incorporates skip connections, which are refined through attention mechanisms, to accurately reconstruct the segmentation map.
4. **Output Layer:** The final layer generates segmentation maps that match the input image dimensions, providing pixel-wise classification.



3.3.3 Forward Pass Workflow

The forward pass follows a step-by-step procedure to process the input and produce the final output. This workflow ensures that the model effectively captures the relevant features and makes accurate predictions:

1. Input images are passed through the encoder blocks to extract relevant features at multiple scales.
2. The bottleneck layer processes the features using dilated convolutions to expand the receptive field and capture broader context.
3. The decoder upsamples the features and merges them with attention-refined skip connections to preserve fine-grained details.
4. The final layer produces the segmentation map, which matches the input dimensions, ready for evaluation.

3.4 Loss Function

3.4.1 Tversky Loss

The Tversky Loss addresses class imbalance by controlling the trade-off between false positives and false negatives.

$$\text{Tversky Index} = \frac{|\text{TP}|}{|\text{TP}| + \alpha|\text{FP}| + \beta|\text{FN}|}$$

$$\text{Tversky Loss} = 1 - \text{Tversky Index}$$

3.4.2 Focal Tversky Loss

To focus on hard examples, we introduce a focal parameter γ:

$$\text{Focal Tversky Loss} = (1 - \text{Tversky Index})^{\gamma}$$

3.4.3 Parameter Selection

Based on experimentation, the following parameters were selected:

- $\alpha = 0.7$ (emphasizes precision)
- $\beta = 0.3$
- $\gamma = 2.0$ (focuses on hard-to-classify regions)

3.5 Hyperparameters

Hyperparameters play a critical role in training deep learning models effectively. This section details the general training settings and specific configurations for self-training.

3.5.1 General Parameters

The general parameters govern the overall training process and were chosen to balance computational efficiency and performance:

- **Epochs:** Initially 300, increased to 1000.
- **Batch Size:** 16 or 32, depending on hardware.
- **Learning Rate:** 0.001 with Adam optimizer.

3.5.2 Self-Training Parameters

Self-training introduces additional hyperparameters to manage the use of pseudo-labels and the integration of unlabeled data. The following settings were applied:

- **Self-Training Start Epoch:** 50
- **Confidence Threshold:** 0.97
- **Similarity Threshold:** 0.85
- **Gamma Loss Weight:** $\gamma_{\text{loss}} = 0.3$

3.6 Self-Training Method

3.6.1 Rationale

Self-training utilizes unlabeled data by generating pseudo-labels from the model's own predictions. These pseudo-labels are then incorporated into the training process, allowing the model to iteratively refine its performance. This approach enables the effective use of abundant unlabeled data, significantly enhancing the model's accuracy and generalization capabilities without requiring additional manual annotations.

3.6.2 Implementation

1. **Initial Training:** Model is trained on labeled data.
2. **Pseudo-Label Generation:**
   (a) Predict on augmented unlabeled images.
   (b) Check if confidence exceeds threshold.
   (c) Make a second prediction without augmentation.
   (d) De-transform the first prediction.
   (e) Compute IoU between predictions.
   (f) If IoU exceeds similarity threshold, accept pseudo-label.
3. **Retraining:** Model is retrained using both labeled and pseudo-labeled data.

3.6.3 Loss Adjustment

Total loss during self-training:

$$\text{Total Loss} = (1 - \gamma_{\text{loss}}) \times \text{Loss}_{\text{labeled}} + \gamma_{\text{loss}} \times \text{Loss}_{\text{pseudo-labeled}}$$

4 Experiments and Results

4.1 Experimental Setup

We conducted experiments to assess the impact of architectural changes, loss functions, and hyperparameters. Metrics were recorded over the validation set for each configuration.

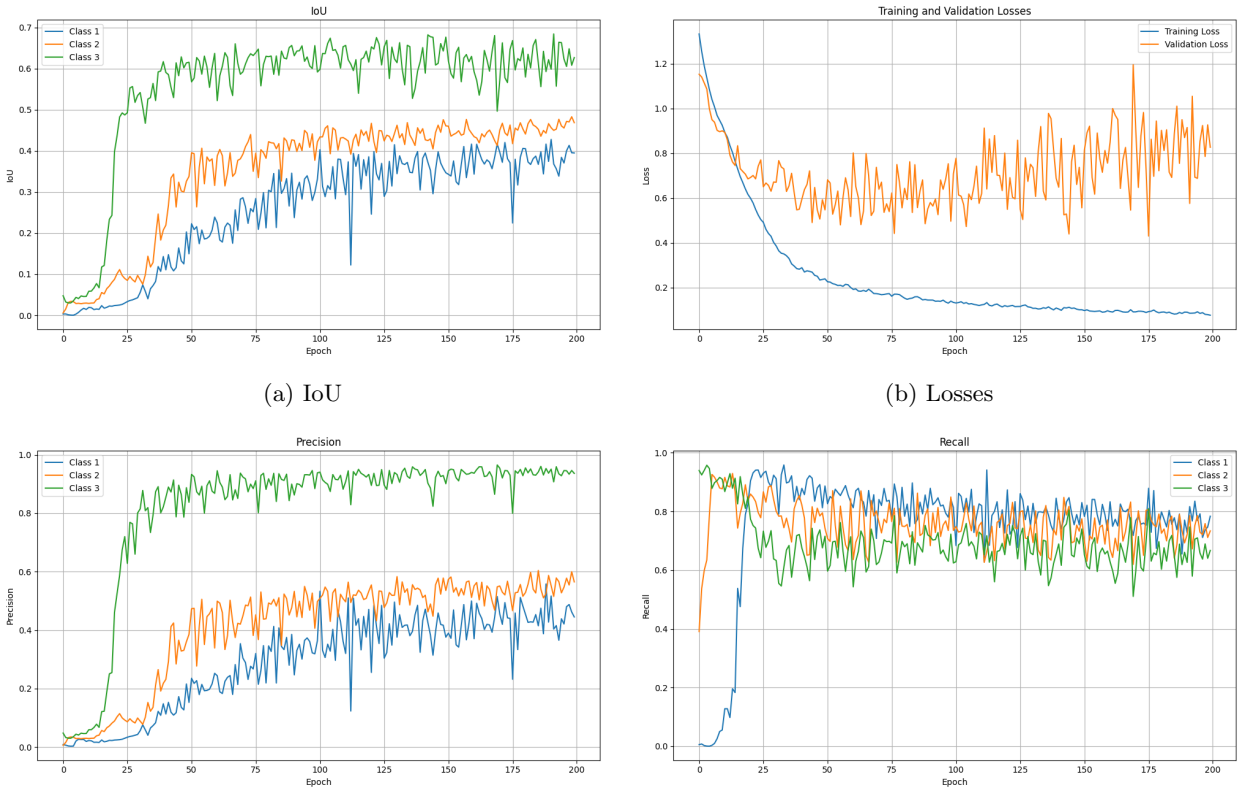
4.2 Metrics

- **Precision and Recall** for each class.
- **Intersection over Union (IoU)**
- **Dice Score**
- **Hausdorff Distance**

4.3 Results

4.3.1 Fully-Supervised Model

As a starting point for the project, we implemented a fully-supervised model using a basic UNet architecture combined with Cross-Entropy loss. This served as a baseline to evaluate subsequent improvements introduced through semi-supervised learning and advanced loss functions.



(a) IoU

(b) Losses

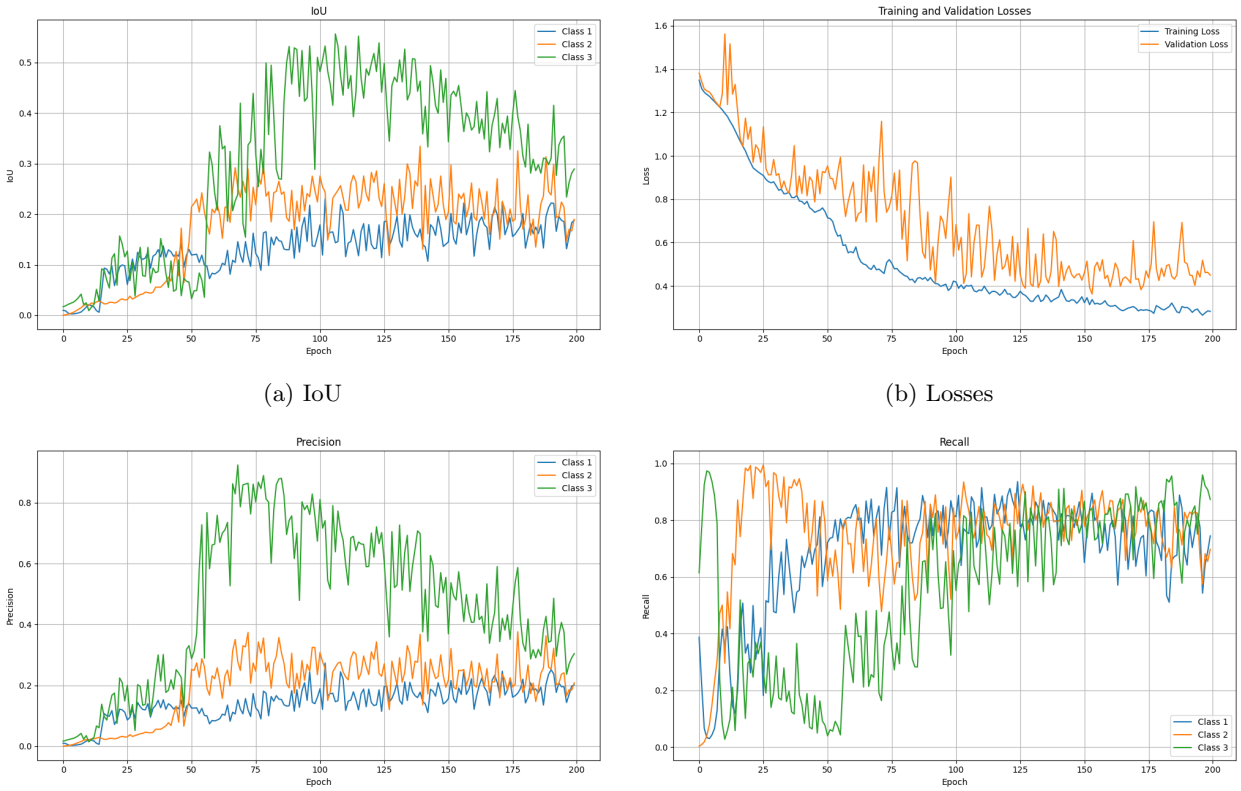
(c) Precision

(d) Recall

Figure 1: Fully-supervised model with Cross-Entropy

4.3.2 Impact of Self-Training

Early self-training attempts did not improve performance, mainly because we failed to save the pseudo-labels properly in a new dataset (a problem that we only noticed later), but also because of relatively poor pseudo-label quality.



(a) IoU

(b) Losses

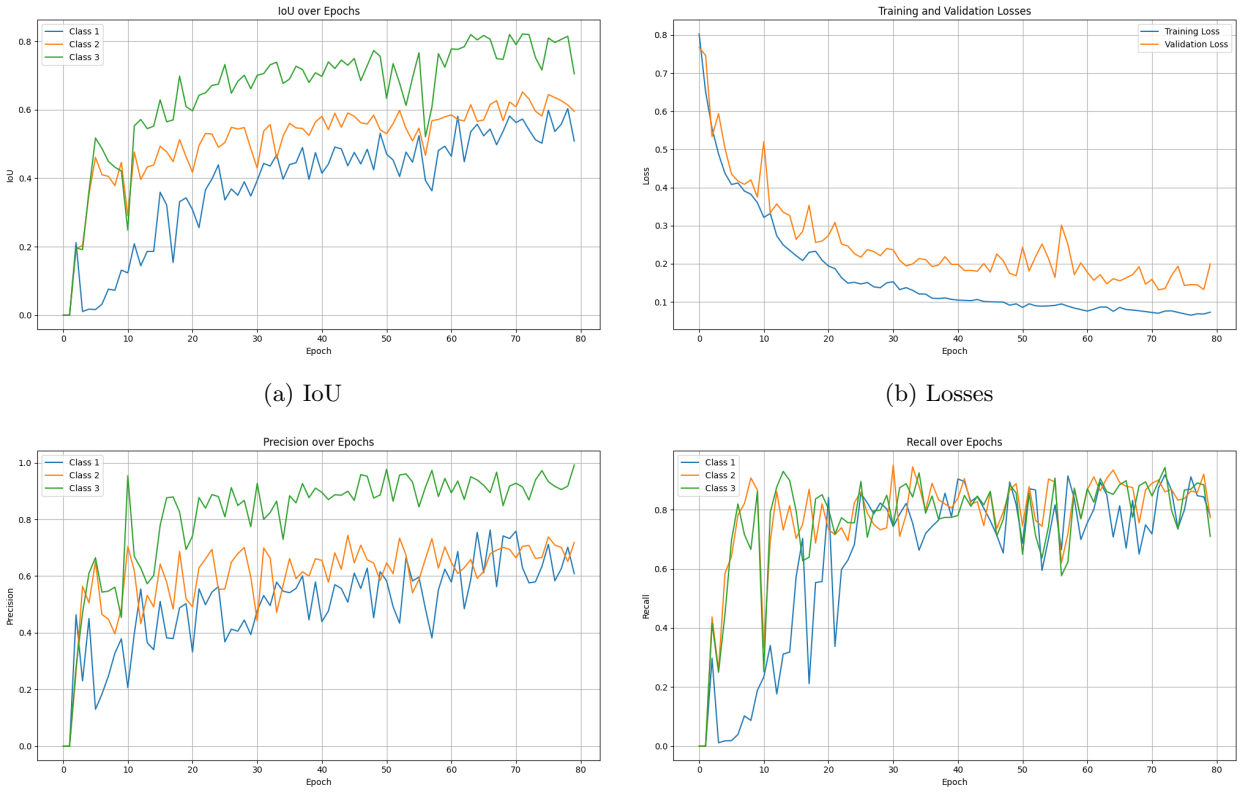
(c) Precision

(d) Recall

Figure 2: Semi-supervised model with Cross-Entropy

4.3.3 Tversky Focal Loss

Initial models using Cross-Entropy Loss underperformed. Switching to Tversky Focal Loss and enhancing the architecture improved results.



(a) IoU

(b) Losses

(c) Precision

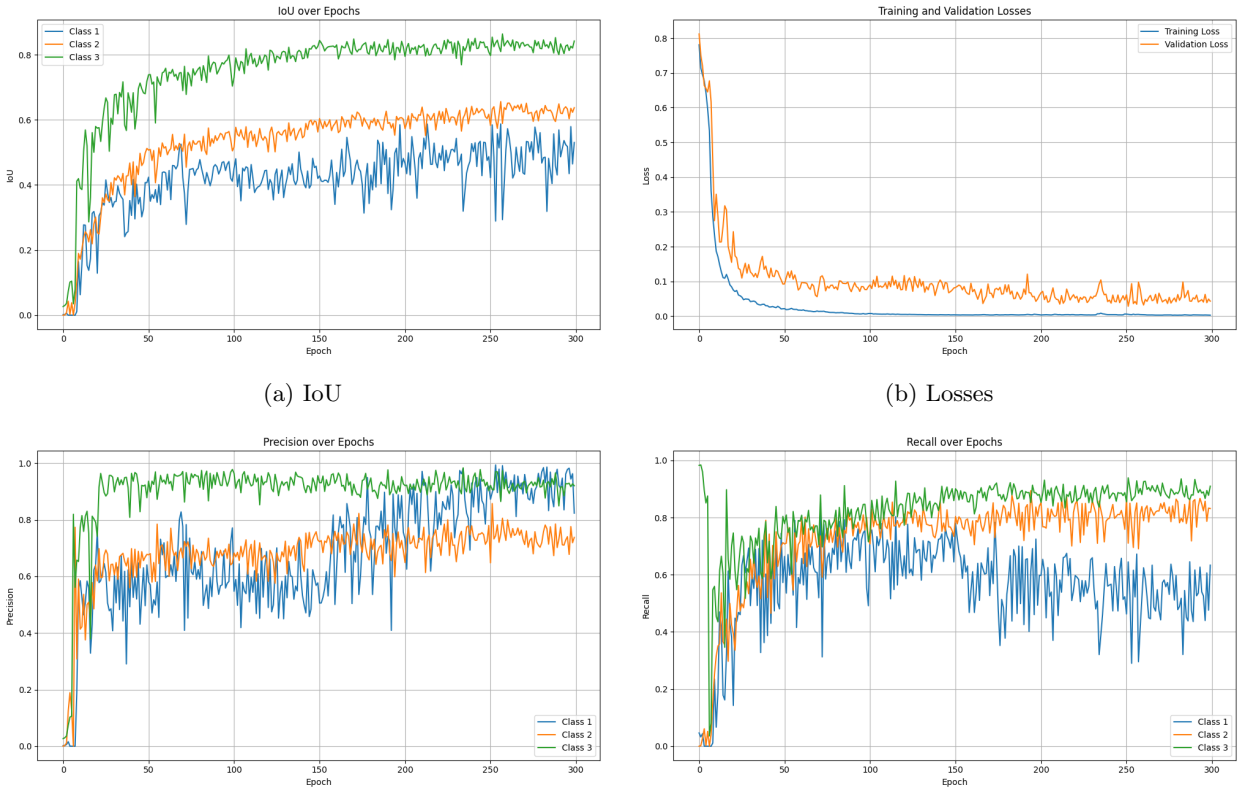
(d) Recall

Figure 3: Fully-supervised model with Tversky Focal Loss

Running the model training for fewer epochs allowed us to isolate and study the fully supervised component. This approach enabled us to evaluate performance relative to each loss.

4.3.4 Self-Training is working properly

At that point, we managed to identify and fix our problem with the saving of the pseudo-labels mentioned above. Self-training did not particularly improve performance, but was at least operational.



(a) IoU

(b) Losses

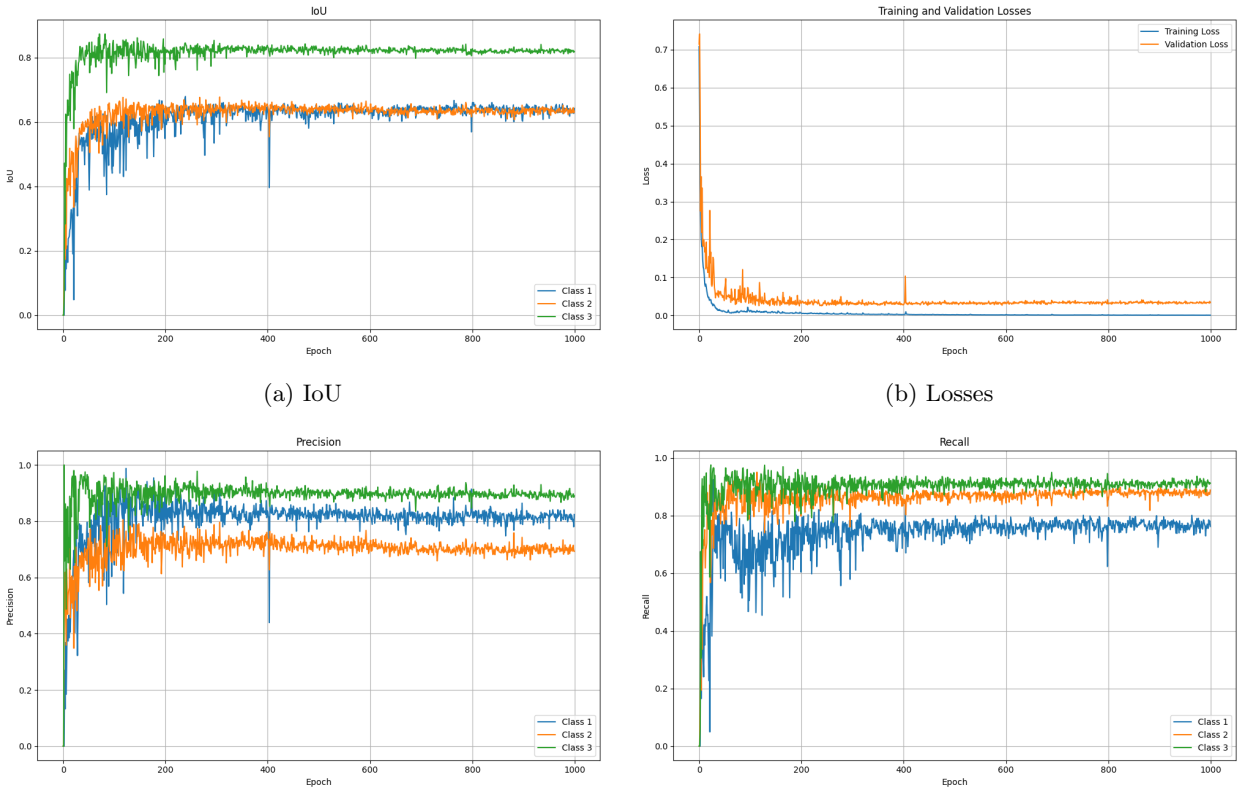
(c) Precision

(d) Recall

Figure 4: Fully functional semi-supervised model

4.3.5 Pseudo-label similarity check

We decided to add another condition before adding pseudo-labels to the training set: a second prediction with a similarity check (see 3.6.2 for details). This ended up being the largest single improvement to our model.



(a) IoU

(b) Losses

(c) Precision

(d) Recall

Figure 5: Semi-supervised model with pseudo-label similarity check

4.3.6 Hyperparameter Tuning

Due to extreme variability in out results (see 5.3), it was extremely difficult to optimize our hyperparameters. We tested each set of hyperparameters multiple time to combat the high variability, and picked the set of hyperparameters that gave our model the highest *Dice Score* on the validation set.

5 Challenges and Solutions

5.1 Pseudo-Label Saving Issues

We discovered that pseudo-labels were not saved correctly, leading to degraded performance during self-training. By revising the saving process, we ensured that pseudo-labels matched input images in format and alignment.

5.2 Self-Training Implementation

Initial self-training did not produce improvements due to low quality pseudo-labels. Enhancing the pseudo-labeling process with confidence and similarity checks improved the quality of pseudo-labels and overall model performance.

5.3 Results Inconsistency

Variability in model performance made it very difficult to assess improvements. We could try to fix this by:

- Setting random seeds to reduce variability.

- Ensuring consistent data preprocessing and augmentation.
- Training the model multiple times for each set of hyperparameters

However, adding those changes could create new problems. Having fixed seeds for random calculations could prevent the model's loss function from reaching a better minimum (e.g. if we prevent the weights initialization from being random). Training the model multiple times for each set of hyperparameters would considerably increase our training time.

6 Discussion

The combination of an enhanced UNet architecture, the Tversky Focal Loss, and self-training effectively improves segmentation performance on the ACDC dataset. The loss function addresses class imbalance by allowing control over false positives and negatives, which is crucial in medical imaging. Self-training leverages unlabeled data, and the introduction of confidence and similarity thresholds ensures only high-quality pseudo-labels are used.

Challenges such as the issues with pseudo-labels were resolved through careful debugging and code refinement. Ensuring data transformations and tensor dimensions were correctly handled was essential for stable training.

7 Conclusion

We developed a semi-supervised segmentation model for cardiac MRI images that effectively utilizes unlabeled data through self-training. The use of the Tversky Focal Loss function addresses class imbalance, and architectural enhancements improve feature extraction and segmentation accuracy.

Our results demonstrate that combining advanced loss functions with semi-supervised learning can significantly improve performance in medical image segmentation tasks with limited labeled data.

7.1 Future Work

Future research could explore other semi-supervised methods, such as consistency regularization or adversarial training, and experiment with transformer-based architectures or domain-specific constraints to further enhance performance.

References

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, 2015.

[2] S. S. Salehi, D. Erdogmus, and A. Gholipour, "Tversky Loss Function for Image Segmentation Using 3D Fully Convolutional Deep Networks," *International Workshop on Machine Learning in Medical Imaging*, pp. 379–387, 2017.

[3] N. Abraham and N. M. Khan, "A Novel Focal Tversky Loss Function With Improved Attention U-Net for Lesion Segmentation," *2019 IEEE 16th International Symposium on Biomedical Imaging*, pp. 683–687, 2019.

[4] X. Zhu, "Semi-Supervised Learning Literature Survey," University of Wisconsin-Madison, 2006.

[5] Kaiming He and Xiangyu Zhang and Shaoqing Ren and Jian Sun,Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification,2015.

[6] Fisher Yu and Vladlen Koltun, Multi-Scale Context Aggregation by Dilated Convolutions,2016.