Deep Learning Approach to Gated Coronary Artery Calcium Scan Segmentation

Ryan Chew

April 24, 2024

Abstract

Coronary artery calcium (CAC) tests are a CT scan of the heart that takes detailed images of arteries to locate deposits of calcium. Excessive calcium in the arteries can lead to reduced blood flow to the heart and cause heart diseases. These are important indicators for heart attacks and diseases and getting CAC scans are useful in identifying high-risk individuals. Agatson scores are the calculations of total calcium in the heart, state of the art programs are currently about 95 percent accurate. The work done in this paper highlights the use of Tversky loss in small segmentations and using an ensemble model to achieve higher accuracy.

1 Introduction

One person dies every 33 seconds in the US from cardiovascular disease and that number totals almost 700,000 people every year [2]. About 20 percent of all deaths in the US can be attributed to heart disease. One of the ways to identify a patient at risk early is by determining their coronary artery calcium (CAC) score. CAC scans take a CT of the patient's heart that allows doctors to identify deposits of calcium within the heart. The higher the CAC score, the higher the risk the patient is of developing or having heart disease. Jamma Internmed [1] proves the value in getting CAC noting that 15 percent of patients classified as high risk by CAC scores had a cardiovascular event within the next 5 to 10 years.

CAC scores must be analyzed and provide good accuracy to result in correct and accurate scoring. Implementing computer vision and AI can improve efficiency as well as accuracy when it comes to analyzing CAC scans.

Currently, CAC analysis requires lots of labor from highly skilled professionals, making it inaccessible and often expensive, up to 400 USD for a scan. This project hopes to introduce a more accurate and efficient way to analyze CAC scans to aid medical professionals and make CAC scans more accessible.

2 Background

The current approach to segmenting calcium deposits is through UNETs, Kazemzadeh22 [3] made use of attention UNETs and found that a variation of the dice loss worked best. To maintain class balance, they kept a 50/50 of positive and negative images. This idea for class balance was also adopted in our project. Zelenik [4] earlier used a combination of three models to localize the heart, segment the heart, and then segment the calcium deposits. Drawbacks of this approach include the slower computation time.

3 Dataset

The dataset originates from Stanford's Artificial Intelligence Medical Imaging Center (AIMI). The dataset contains heart CT scans of 450 labeled patients, containing roughly 40,000 images across around 900 total patients. Each image is 512x512 pixels in resolution and is provided in DICOM file format. There is one channel in all of the images: grayscale. All images are uniform in orientation (100010). The images were normalized to range from 0 to 1.

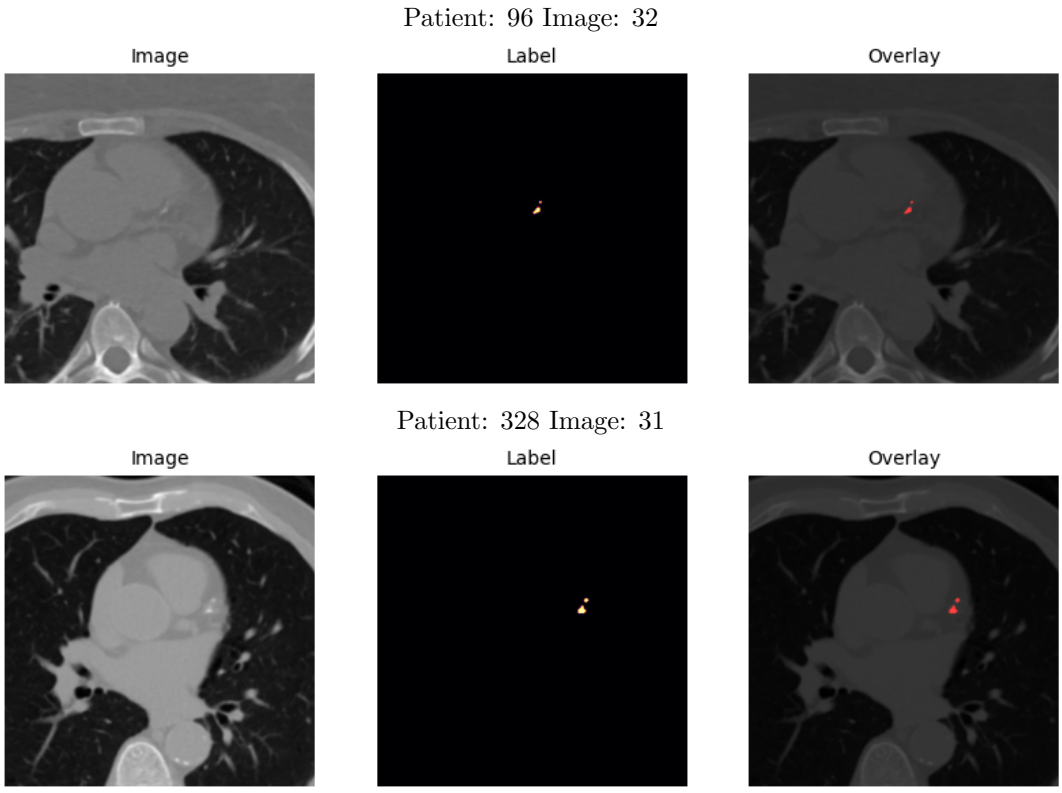

Figure 1: Dataset Images

The labels were provided in XML format. Data was provided by patients, with each XML file corresponding to a single patient. Images that contained regions of interest were included in the XML file with an Image Index. The image index (i) in the XML file corresponds to the **(n-i)th** image where n denotes the total number of images in the patient's image folder. The segmentation mask was written as the points that compose the border of the ROI. Using Numpy meshgrid, segmentations were extracted and labels were created as 256x256 arrays with binary values (1=segmentation, 0=not segmentation).

Due to an abundance of blank images in the dataset, only an estimated one-eighth of all blank images were included. The final dataset yielded 3062 total images with segmentations and — blanks. Tests were conducted using a train-test-split as the data-splitting method.

4 Methodology

4.1 Models

Model	Loss Function	Number of Parameters	Epochs	Batch Size	Spatial Attention
Baseline	BCE	7,759,521	20	16	No
Model 2	Tversky	7,759,521	100	16	No
Model 3	Tversky	131,420,267	100	16	No
Model 4	BCE	124,424,363	200	16	No
Model 5	Tversky	131,420,368	100	16	Yes

The dataset poses a semantic segmentation task. We proposed a UNET as it is the standard for medical imaging segmentation. A UNET consists of a contracting and an expanding path. The contracting path downsamples the image into latent space before the expanding path upsamples it using transposed convolutions to return the image to its original dimensions. Using a crop and copy, information from the original downsampled image can be retained and concatenated into the final segmentation. UNET is the model of choice as it produces a pixel-level segmentation map and the downsample/upsample

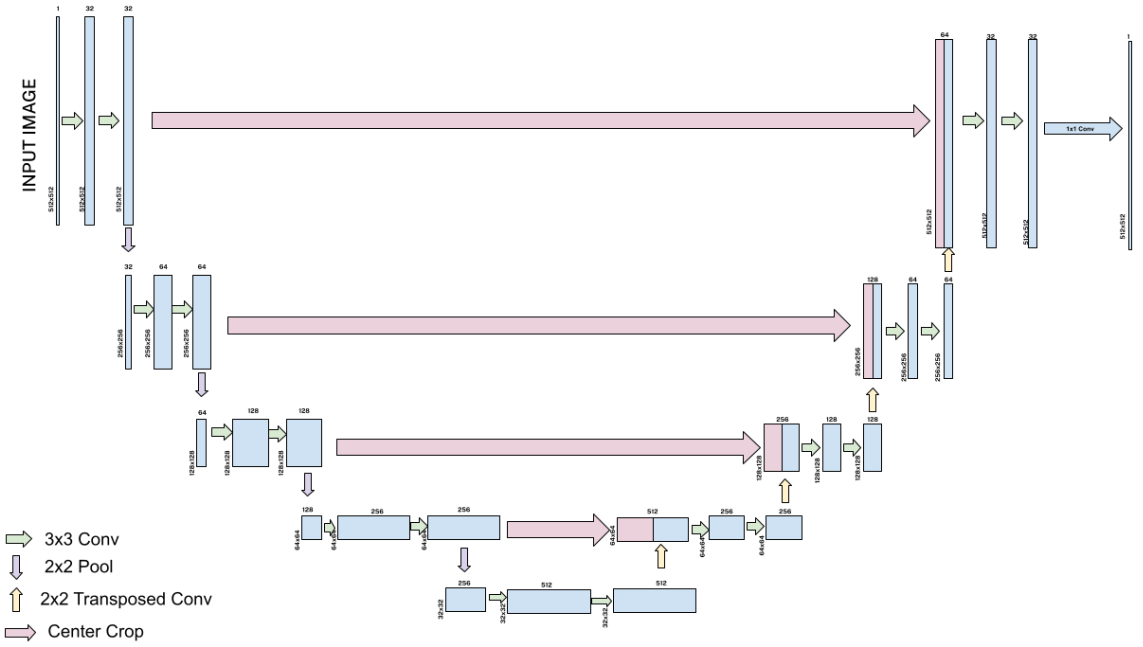

Figure 2: Baseline UNET Architecture

allows the model to learn both detailed and undetailed information. In the end, we multiplied the predicted values with the original image to eliminate guessing blank pixels. All models were trained with batch sizes of 16 and initial learning rates of 0.1 using a learning rate reduction on plateau scheduler with a factor of 0.5.

4.1.1 Baseline

The baseline model is a standard UNET consisting of 2 3x3 convolution layers and a 2x2 Max Pool to downsample. The copy and crop use PyTorch's center crop and are concatenated with the output of the transposed convolution layer. The transposed convolution is a 2x2 that takes the image from 512 channels in latent space back to 1 in the end. The output makes use of a 1x1 convolution layer to make pixel-level predictions. The baseline model uses the Dice-Sorenson Coefficient as a metric. The initial learning rate was set to 0.1 with a learning rate reduction factor of 0.5 on plateau. Batch size was set to 16 and run for 20 epochs.

4.1.2 Model Two

The model is a standard UNET consisting of 2 3x3 convolution layers and a 2x2 Max Pool to downsample. The copy and crop use PyTorch's center crop and are concatenated with the output of the transposed convolution layer. The transposed convolution is a 2x2 that takes the image from 1024 channels in latent space back to 1 in the end. The output makes use of a 1x1 convolution layer to make pixel-level predictions. The model uses the Tversky as a metric. The initial learning rate was set to 0.1 with a learning rate reduction factor of 0.5 on plateau. The alpha and beta values were searched across a random search space of 0.5-1 for the alpha value and 0-0.5 for beta. The final model used 0.9 for alpha and 0.1 for beta. This model was trained with batch sizes of 16 and ran for 100 epochs.

4.1.3 Model Three

The model is a UNET consisting of 2 3x3 convolution layers, a 1x1 convolution layer, and a 2x2 Max Pool to downsample. The copy and crop use PyTorch's center crop and are concatenated with the output of the transposed convolution layer. The transposed convolution is a 2x2 that takes the image from 2048 channels in latent space back to 1 in the end. The output makes use of a 1x1 convolution

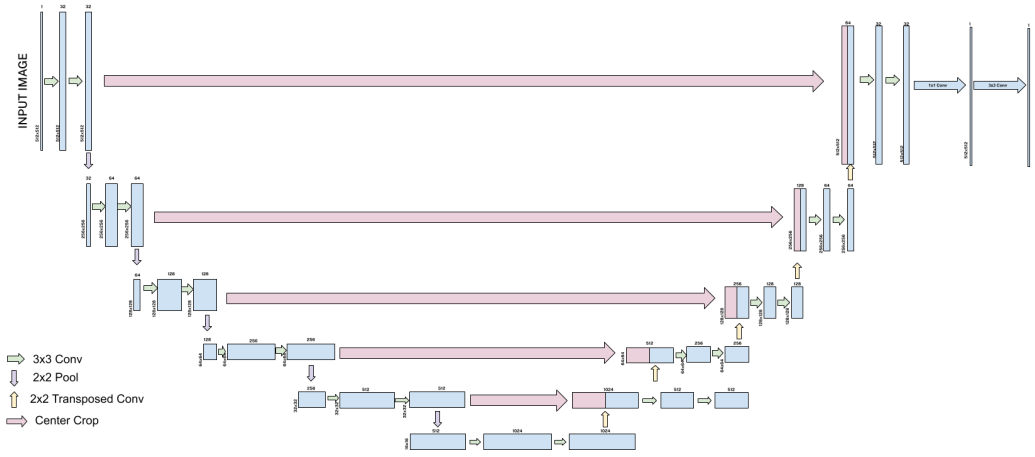

Figure 3: Model 4 UNET Architecture

layer followed by a 3x3 convolutional layer to make pixel-level predictions. The model uses the Tversky as a metric. The initial learning rate was set to 0.1 with a learning rate reduction factor of 0.5 on plateau. The alpha and beta values were searched across a random search space of 0.5-1 for the alpha value and 0-0.5 for beta. This model was run for 200 epochs with the final alpha and beta values set at 0.82 and 0.2 respectively.

4.1.4 Model Four

The model is a standard UNET consisting of 2 3x3 convolution layers and a 2x2 Max Pool to downsample. The copy and crop use PyTorch's center crop and are concatenated with the output of the transposed convolution layer. The transposed convolution is a 2x2 that takes the image from 2048 channels in latent space back to 1 in the end. The output makes use of a 1x1 convolution layer to make pixel-level predictions. The model uses the BCE With Logits Loss as a metric. The initial learning rate was set to 0.1 with a learning rate reduction factor of 0.5 on plateau. This was run for 100 epochs in total.

4.1.5 Model Five

The model is a standard UNET including a spatial attention module (SAM) at the bottom of the contracting path. It consists of 2 3x3 convolution layers and a 2x2 Max Pool to downsample by a factor of two. The copy and crop use PyTorch's center crop and are concatenated with the output of the transposed convolution layer. The transposed convolution is a 2x2 that takes the image from 2048 channels in latent space back to 1 in the end. The output makes use of a 1x1 convolution layer to make pixel-level predictions. The model was trained with both BCE and Tversky loss, with the Tversky model using alpha set to 0.82 and beta set to 0.2. This was trained for 100 epochs using batch sizes of 16. The final model uses Tversky loss.

4.1.6 Ensemble Model: Baseline + Model 3

This model combined model 1 and model 3 using the strengths of the baseline model in accuracy and the larger predictions in model three to create a more accurate overall model. It multiplies the output of the Tversky model by the squared values of the baseline prediction and multiplied by the original image.

4.1.7 Ensemble Model: Baseline + Model 5

This model combined model 1 and model 5 using the strengths of the baseline model in accuracy and the larger predictions in model three to create a more accurate overall model. It multiplies the output of the spatial attention U-NET model by the squared values of the baseline prediction and multiplied by the original image.

4.2 Metrics

Several metrics were tested for the accuracy of the segmentation models. All metrics were tested on training data, validation data as well as testing data.

4.2.1 Dice Sorenson Coefficient

Dice Loss provides a metric to compare the total area of the union relative to the total predicted and the total true segmentation area. Dice Loss provides a gauge for true positives and false negatives. However, since the segmentations in this problem were extremely small, the intersection value would also be small making it hard to tune the model.

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

4.2.2 Binary Cross Entropy

BCE loss (log loss) is evaluated on the probability of each pixel. It accounts for both predicted ones and zeros and even in small segmentations provides a good idea of the accuracy. On top of that, binary cross entropy tunes the model to favor predicting only ones and zeros making inference far easier.

$$BCE = \frac{1}{N}\sum_{n=1}^{N}[y_n log\hat{y_n} + (1 - y_n)log(1 - \hat{y_n})]$$

4.2.3 Binary Cross Entropy Dice Coefficient

BCE dice provide the stability of a BCE model and the benefits of a dice coefficient. This computes the pixel-level accuracy by averaging the log loss of the predicted pixel. This is achieved by returning the sum of the BCE and the dice loss.

4.2.4 Intersection Over Union

IOU computes the total area of overlap between the predicted segmentation and the ground truth, then divides by the sum of the areas added up. Intersection over union is useful in determining the accuracy of the model but can over-penalize when segmenting small objects.

$$IOU = \frac{|X \cap Y|}{|X \cup Y|}$$

4.2.5 Tversky Loss

Using values for alpha and beta, Tversky loss is able to weigh false positives or false negatives more heavily. This is good for class-imbalanced datasets. It is calculated as true positives divided by the sum of true positives and false positives each multiplied by a defined factor. The benefit of Tversky loss is the ability to change the weight to account for larger or smaller segmentations. In this case, the alpha value is much larger to punish the prediction of false positives to prevent predicting the entire image.

$$Tversky = \frac{TP}{TP + \alpha(FP) + \beta(FN)}$$

4.3 Hardware

This experiment was run locally on a personal computer using 64GB DDR4 RAM, a 12th generation Intel CPU, and an NVIDIA RTX3060 with CUDA enabled. The PyTorch version is 2.2.2 and uses CUDA version 12.1.

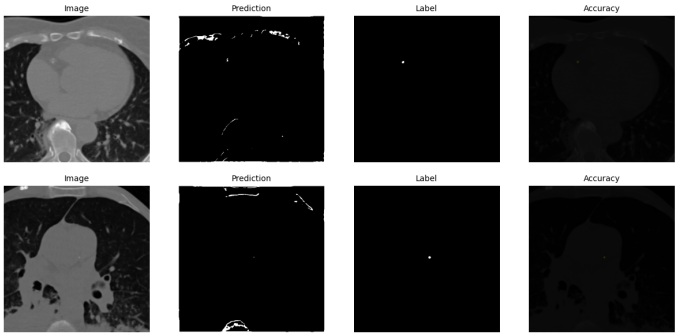

Figure 4: Baseline Model Result Inference Images

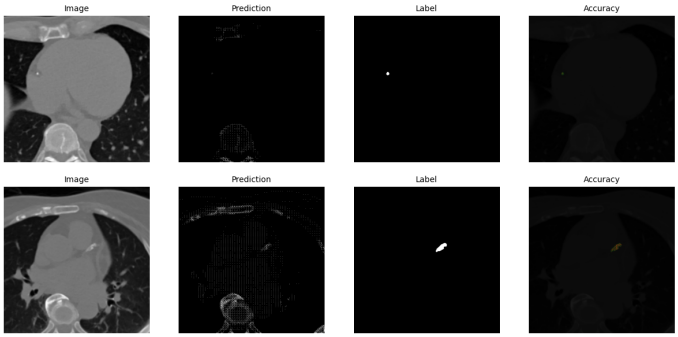

Figure 5: Model 2 Result Inference Images

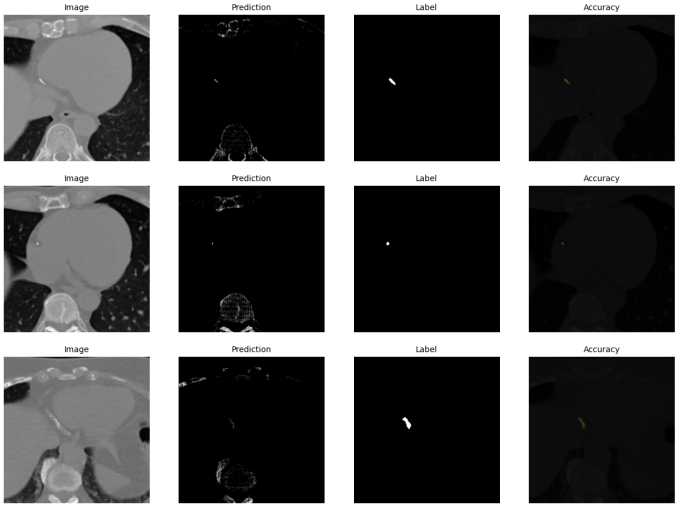

Figure 6: Model 3 Result Inference Images

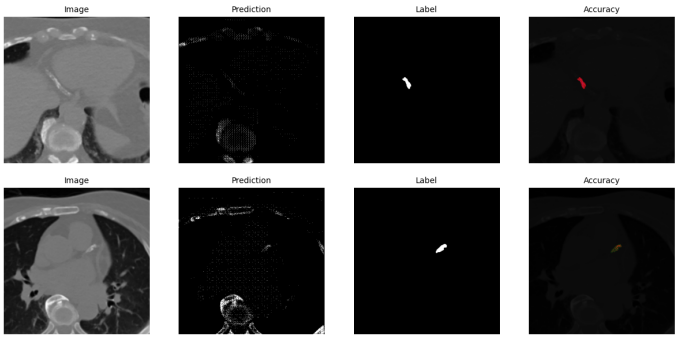

Figure 7: Model 4 Result Inference Images

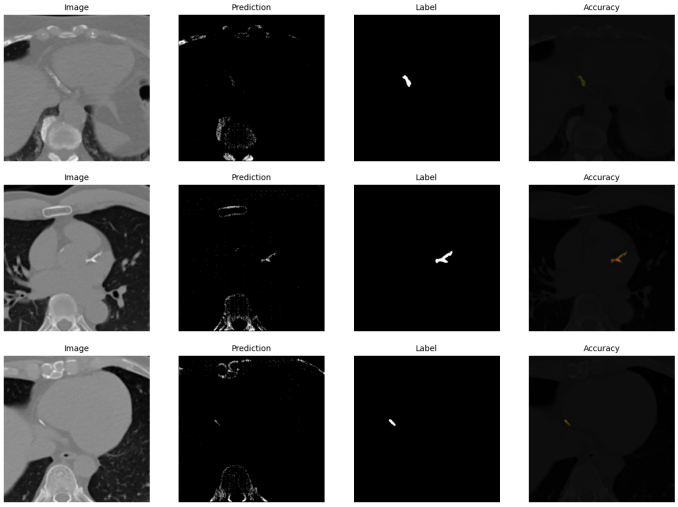

Figure 8: Model 4 Result Inference Images

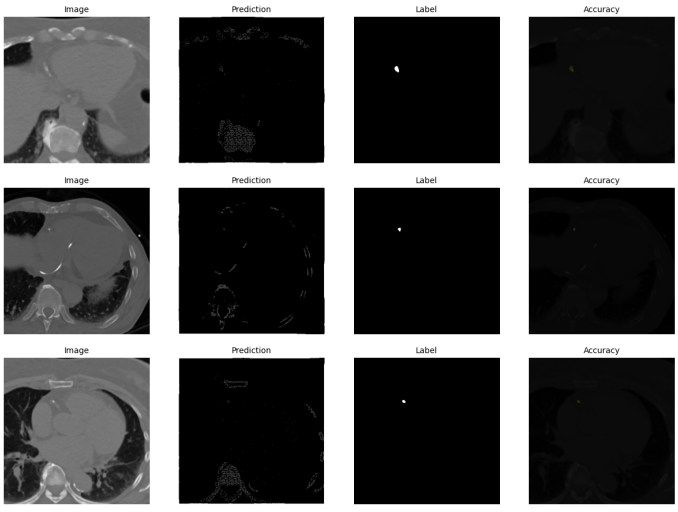

Figure 9: Baseline + Model 3 Result Inference Images

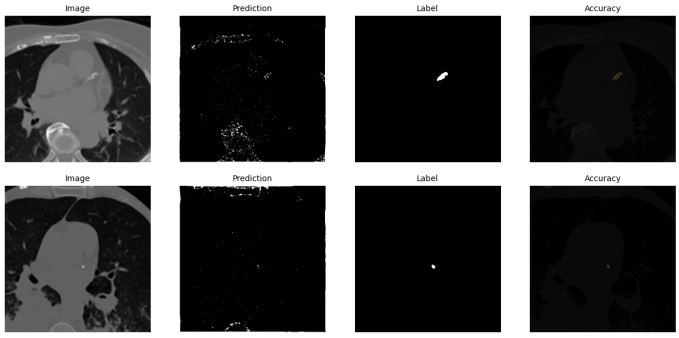

Figure 10: Baseline + Model 5 Result Inference Images

5 Results

Model	BCE Loss	Dice Loss	IOU Loss	Inference Time (ms)
Baseline	0.6934	0.9914	0.9999	0.312
Model 2	0.7895	0.9997	0.9999	0.312
Model 3	0.7543	0.9997	0.9998	0.781
Model 4	0.7689	0.9997	0.9999	0.781
Model 5	0.7470	0.9997	0.9998	0.938
Baseline + Model 3	0.6931	0.3288	0.3288	1.090
Baseline + Model 5	0.6931	0.3205	0.3205	2.190

6 Conclusions

Results show that amongst the four models, the baseline model using BCE Loss outperformed all other models tested. Although achieving the highest accuracy, inference shows that the baseline model tends to predict extremely small segments across the entire image. On the contrary, model 3 would guess larger areas with good accuracy but include a lot more noise. Models 2 and 4 had significantly more noise and predictions were often not as certain as models one and three. The ensemble model using model 5 slightly outperforms the ensemble using model 3.

6.1 Further Research

This research yielded results showing the promise of Tversky loss in training models for CAC scan analyses. However, as seen in the results section many models predicted small dots across larger areas, experiments may be done with shape and size to filter out false positives. Other implementations such as including layers and using 3D UNETs may also yield more results. This research was limited to 2D UNETs as a result of time and resource constraints. Although from a metrics standpoint, the baseline model outperformed all other models, the small segments might be a cause for concern when calculating scores, and rather, model 3 does significantly better on an eye test. The ensemble model as a result improves the results significantly, reducing both Dice and IOU loss greatly.

7 Acknowledgments

References

[1] Katy J. L. Bell, Sam White, Omar Hassan, Lin Zhu, Anna Mae Scott, Justin Clark, and Paul Glasziou. Evaluation of the Incremental Value of a Coronary Artery Calcium Score Beyond Traditional Cardiovascular Risk Assessment: A Systematic Review and Meta-analysis. *JAMA Internal Medicine*, 182(6):634–642, 06 2022.

[2] Centers for Disease Control and Prevention, May 2023.

[3] Sahar Kazemzadeh, Manish Singh, Besher Ashouri, and Samvel Gyurdzhyan. Prediction of coronary artery disease via calcium scoring of chest cts, 2021.

[4] Roman Zeleznik, Borek Foldyna, Parastou Eslami, Jakob Weiss, Ivanov Alexander, Jana Taron, Chintan Parmar, Raza M. Alvi, Dahlia Banerji, Mio Uno, Yasuka Kikuchi, Julia Karady, Lili Zhang, Jan-Erik Scholtz, Thomas Mayrhofer, Asya Lyass, Taylor F. Mahoney, Joseph M. Massaro, Ramachandran S. Vasan, Pamela S. Douglas, Udo Hoffmann, Michael T. Lu, and Hugo J. W. L. Aerts. Deep convolutional neural networks to predict cardiovascular risk from computed tomography. 01 2021.