CMPSC 496 Capstone Weekly Report

Project Title: Development of an artificial intelligence algorithm to detect melanoma development in transmitted images: A tool to accelerate access to care in Pennsylvania

Summary (temp): Pennsylvania State University is a land grant institution and, as such, we are poised to utilize our resources and synergize our expertise to improve the health of Pennsylvanians. This proposed project will allow for development of a novel AI algorithm that can be utilized by dermatologists and primary care providers to assist in melanoma detection from dermoscopy images.

GitHub

GitHub Repository: https://github.com/jinyoonok2/Skin-Cancer-Detection-Capstone

Dataset

1. **ISIC 2018:** https://challenge.isic-archive.com/data/#2018

   - The HAM10000 dataset is included in ISIC 2018 and is also a part of the ISIC 2019 dataset.
   - A portion of this dataset was manually labeled using Roboflow.
   - This manually labeled data was employed to train a model capable of automatic and comprehensive self-labeling.
   - The trained model was subsequently utilized to assist in instance segmentation within the combined dataset.

2. **Combined Dataset:** only training proportion of each was used (50000+)

   - ISIC 2019: https://challenge.isic-archive.com/data/#2019
   - ISIC 2020: https://challenge.isic-archive.com/data/#2020

Weekly Report

1/12/2024 (Week 1)

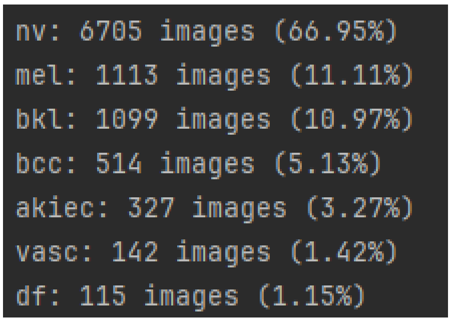
1. **The model is trained on ISIC 2018 dataset. The information of the model trained on ISIC 2018 (HAM10000) is like the following:**

- Model Type: YOLOv8s-seg.pt (segmentation model)
- Class Names:

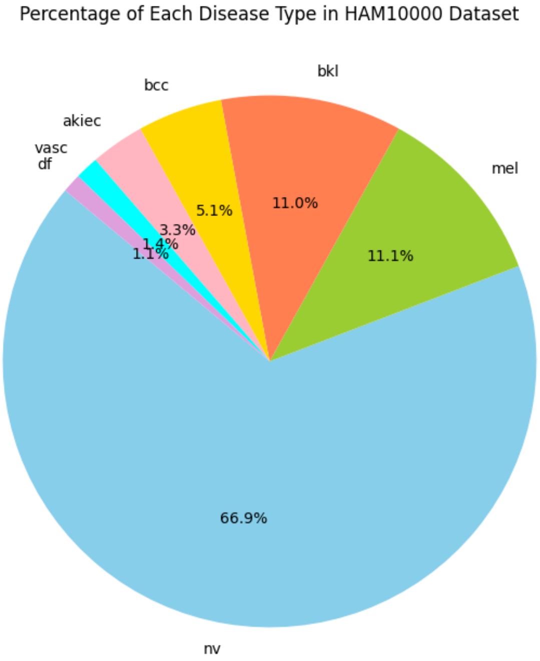
AKIEC: Actinic keratoses and intraepithelial carcinoma / Bowen's disease.
BCC: Basal cell carcinoma.
BKL: Benign keratosis-like lesions, including solar lentigines, seborrheic keratoses, and lichen-planus like keratoses.
DF: Dermatofibroma.
MEL: Melanoma.
NV: Melanocytic nevi.

VASC: Vascular lesions, including angiomas, angiokeratomas, pyogenic granulomas, and hemorrhage.

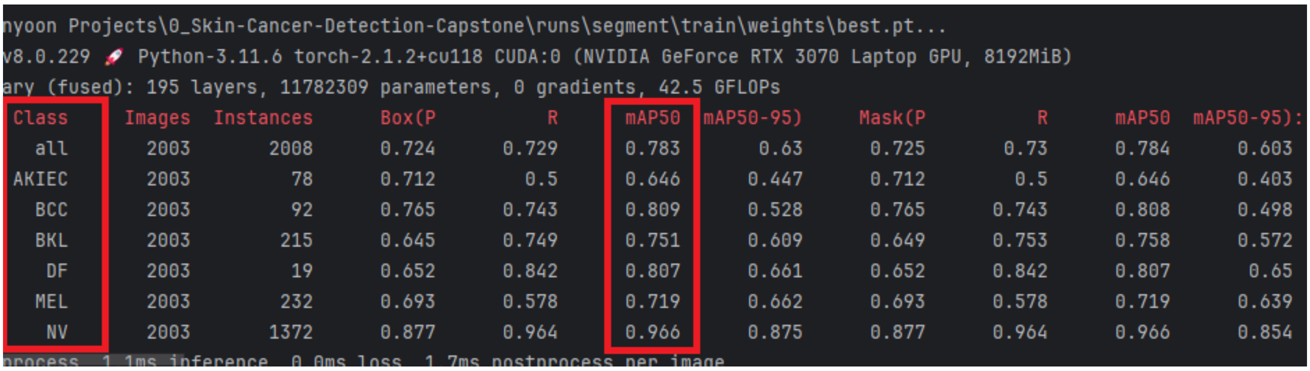
- Dataset Distribution



```
nv: 6705 images (66.95%)
mel: 1113 images (11.11%)
bkl: 1099 images (10.97%)
bcc: 514 images (5.13%)
akiec: 327 images (3.27%)
vasc: 142 images (1.42%)
df: 115 images (1.15%)
```



- Performance of the YOLOv8 pretrained model trained on this dataset (20 epochs):



```
nyoon Projects\0_Skin-Cancer-Detection-Capstone\runs\segment\train\weights\best.pt...
v8.0.229 🚀 Python-3.11.6 torch-2.1.2+cu118 CUDA:0 (NVIDIA GeForce RTX 3070 Laptop GPU, 8192MiB)
ary (fused): 195 layers, 11782309 parameters, 0 gradients, 42.5 GFLOPs
 Class    Images  Instances      Box(P          R       mAP50  mAP50-95)     Mask(P          R       mAP50  mAP50-95):
   all      2003       2008      0.724      0.729      0.783       0.63      0.725       0.73      0.784      0.603
 AKIEC      2003         78      0.712        0.5      0.646      0.447      0.712        0.5      0.646      0.403
   BCC      2003         92      0.765      0.743      0.809      0.528      0.765      0.743      0.808      0.498
   BKL      2003        215      0.645      0.749      0.751      0.609      0.649      0.753      0.758      0.572
    DF      2003         19      0.652      0.842      0.807      0.661      0.652      0.842      0.807       0.65
   MEL      2003        232      0.693      0.578      0.719      0.662      0.693      0.578      0.719      0.639
    NV      2003       1372      0.877      0.964      0.966      0.875      0.877      0.964      0.966      0.854
process  1.1ms inference  0.0ms loss  1.7ms postprocess per image
```

2. **Next, the segmentation model that is trained on ISIC 2018 dataset was used to assist most of the labeling process of the combined dataset (ISIC 2019 + ISIC 2020).**

- Combined dataset is later manually processed through Roboflow computer vision platform to correct missing or crucially misinterpreted labels.

- Dataset is oriented and resized (640x640, 256x256) then downloaded to local to perform experiments

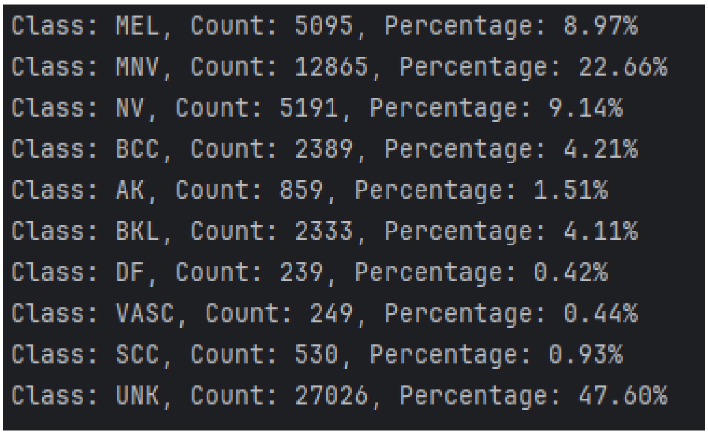
- Class correction, data splitting, data duplication for the preparation of the data augmentation, and minor processes were done locally.

- New class Names of the combined dataset:

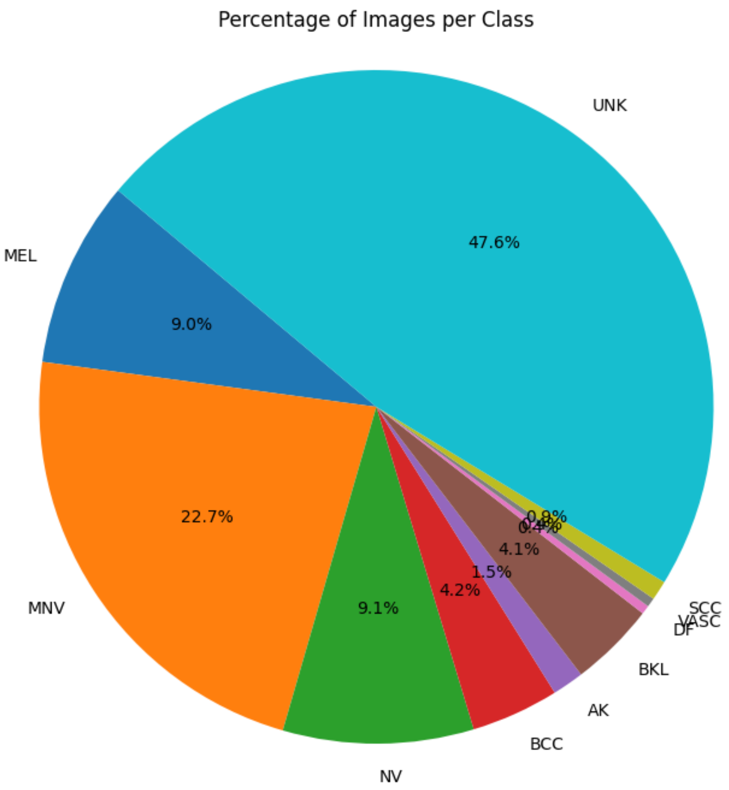
MEL: Melanoma.
MNV: Melanocytic nevus.
NV: Nevus.
BCC: Basal cell carcinoma.
AK: Actinic keratoses and intraepithelial carcinoma / Bowen's disease.
BKL: Benign keratosis-like lesions, including solar lentigines, seborrheic keratoses, and lichen-planus like keratoses.
DF: Dermatofibroma.
VASC: Vascular lesions, including angiomas, angiokeratomas, pyogenic granulomas, and hemorrhage.
SCC: Squamous cell carcinoma
UNK: All unknown cases from ISIC 2020 are diagnosed as **benign**.

- Classes with a very small population of data were combined into larger, more inclusive categories to reduce confusion.

- Dataset distributions for the combined dataset:



```
Class: MEL, Count: 5095, Percentage: 8.97%
Class: MNV, Count: 12865, Percentage: 22.66%
Class: NV, Count: 5191, Percentage: 9.14%
Class: BCC, Count: 2389, Percentage: 4.21%
Class: AK, Count: 859, Percentage: 1.51%
Class: BKL, Count: 2333, Percentage: 4.11%
Class: DF, Count: 239, Percentage: 0.42%
Class: VASC, Count: 249, Percentage: 0.44%
Class: SCC, Count: 530, Percentage: 0.93%
Class: UNK, Count: 27026, Percentage: 47.60%
```



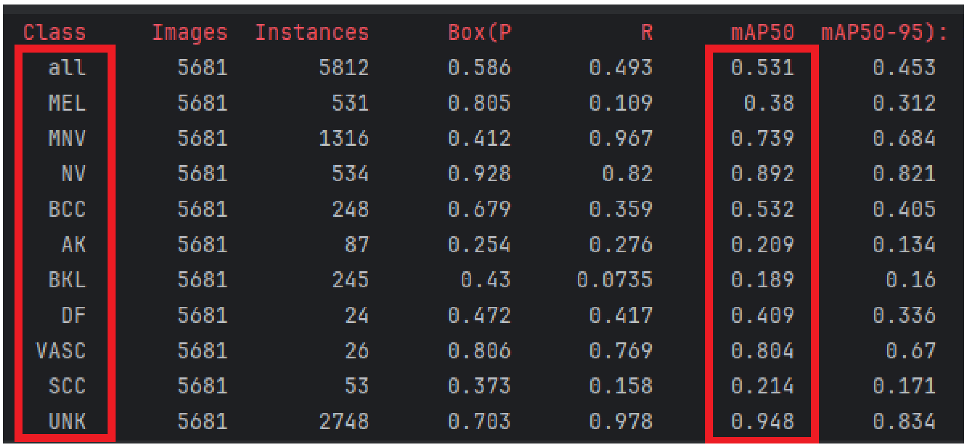
- For data duplication, a magnification setting is applied to each class. For instance, if the setting is 8, each data point in the class should be duplicated 7 additional times.

```
magnification factors = {
    'MEL': 2, 'MNV': 1, 'NV': 2, 'BCC': 2, 'AK': 4,
    'BKL': 2, 'DF': 8, 'VASC': 8, 'SCC': 4, 'UNK': 1
}
```

- The duplication process is applied exclusively to the training split of the dataset. After this, the customized dataset class, with random augmentation, is passed through the data loader to initiate the training phase.

3. **First experiment with 20 epochs**

Model Type: YOLOv8s.pt (detection model)



```
Class     Images  Instances      Box(P          R      mAP50  mAP50-95):
  all       5681       5812      0.586      0.493      0.531      0.453
  MEL       5681        531      0.805      0.109       0.38      0.312
  MNV       5681       1316      0.412      0.967      0.739      0.684
   NV       5681        534      0.928       0.82      0.892      0.821
  BCC       5681        248      0.679      0.359      0.532      0.405
   AK       5681         87      0.254      0.276      0.209      0.134
  BKL       5681        245       0.43     0.0735      0.189       0.16
   DF       5681         24      0.472      0.417      0.409      0.336
 VASC       5681         26      0.806      0.769      0.804       0.67
  SCC       5681         53      0.373      0.158      0.214      0.171
  UNK       5681       2748      0.703      0.978      0.948      0.834
```

1/19/2024 (Week 2)

1. **Magnification rate** of each class and **augmentation probability** adjusted according to the result of the first experiment (20 epochs)

   - **magnification_factors(new) = {**

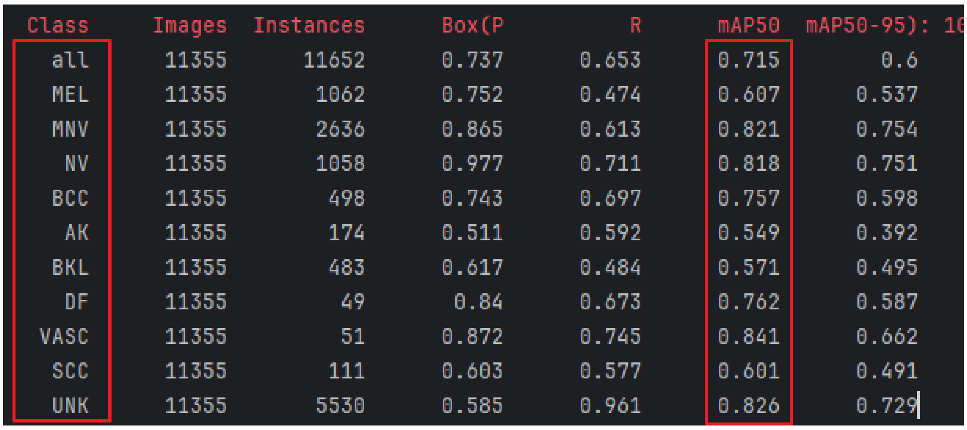
     'MEL': 4, 'MNV': 2, 'NV': 2, 'BCC': 4, 'AK': 16,

     'BKL': 8, 'DF': 16, 'VASC': 8, 'SCC': 16, 'UNK': 1

     }

   - Mosaic augmentation rate for augment specific classes: 0.8 -> 1.0

2. Results of the Evaluation on the Validation Dataset: After completing 50 epochs of training with modified augmentation probabilities and class magnification settings.



```
      Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 10
        all      11355      11652      0.737      0.653      0.715        0.6
        MEL      11355       1062      0.752      0.474      0.607      0.537
        MNV      11355       2636      0.865      0.613      0.821      0.754
         NV      11355       1058      0.977      0.711      0.818      0.751
        BCC      11355        498      0.743      0.697      0.757      0.598
         AK      11355        174      0.511      0.592      0.549      0.392
        BKL      11355        483      0.617      0.484      0.571      0.495
         DF      11355         49       0.84      0.673      0.762      0.587
       VASC      11355         51      0.872      0.745      0.841      0.662
        SCC      11355        111      0.603      0.577      0.601      0.491
        UNK      11355       5530      0.585      0.961      0.826      0.729
```

3. Evaluation of the Error Percentage in the Model's Predictions on the Validation Dataset.

Percentages for MNV:

MNV: 51.05%

UNK: 30.74%

MEL: 2.92%

BKL: 3.05%

NV: 10.57%

BCC: 0.99%

DF: 0.27%

SCC: 0.03%

VASC: 0.03%

AK: 0.34%

Percentages for MEL:

MNV: 14.18%

MEL: 47.79%

BKL: 5.99%

UNK: 27.44%

DF: 0.28%

BCC: 0.92%

NV: 0.46%

AK: 2.39%

SCC: 0.37%

VASC: 0.18%

Percentages for BKL:

UNK: 32.30%

BKL: 55.13%

SCC: 1.55%

MNV: 3.48%

AK: 3.48%

MEL: 2.32%

DF: 0.19%

BCC: 1.55%

Percentages for SCC:

UNK: 17.31%

SCC: 59.62%

AK: 3.85%

MEL: 5.77%

BKL: 4.81%

BCC: 8.65%

Percentages for BCC:

BCC: 67.45%

UNK: 9.55%

SCC: 2.53%

MNV: 4.29%

MEL: 2.92%

BKL: 5.65%

AK: 7.41%

DF: 0.19%

Percentages for VASC:

VASC: 69.64%

UNK: 12.50%

MEL: 1.79%

DF: 3.57%
BKL: 1.79%
MNV: 3.57%
BCC: 7.14%

Percentages for AK:
UNK: 9.83%
BCC: 5.78%
AK: 72.83%
BKL: 7.51%
SCC: 3.47%
MEL: 0.58%

Percentages for DF:
UNK: 19.23%
DF: 73.08%
MNV: 1.92%
BCC: 3.85%
BKL: 1.92%

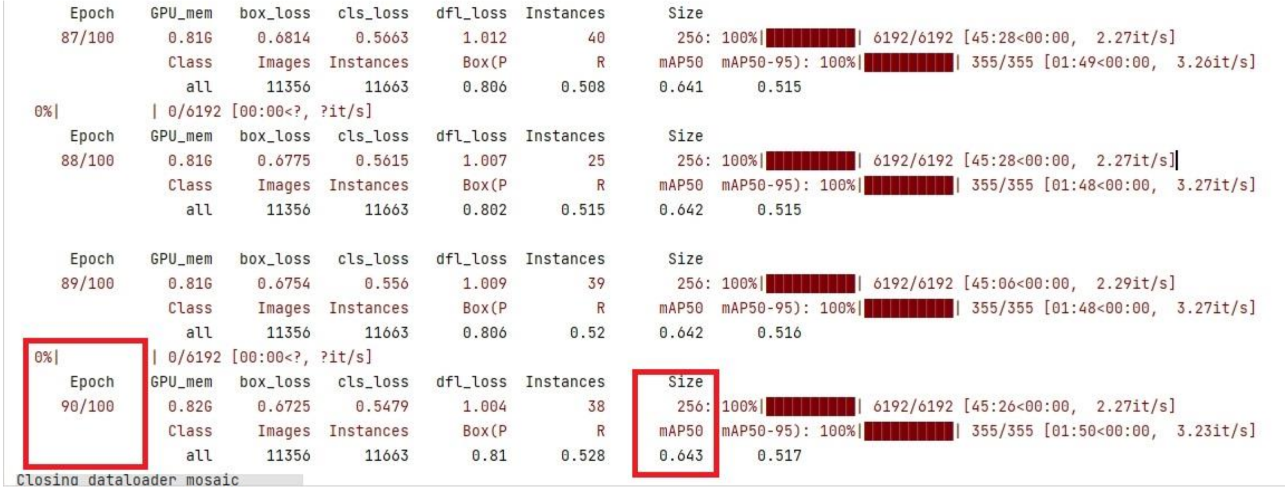
Percentages for UNK:
UNK: 99.93%
BKL: 0.05%
MEL: 0.02%

Percentages for NV:
NV: 63.83%
UNK: 36.09%
BKL: 0.09%

4. Because of the strong correlation observed among the **BKL, NV, MNV**, and **UNK** classes, an attempt was made to amalgamate these class datasets into a single **Benign (BNG)** class. This was coupled with an increase in the number of epochs and an adjustment in class

weighting.

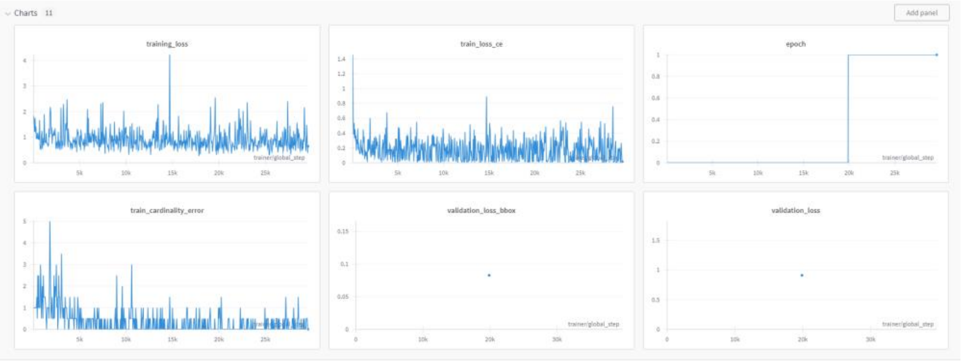
Result: Unfortunately, this approach of class amalgamation and adjustment in magnification did not yield successful results when compared to the original class configurations and weights. Consequently, these modifications will be reverted.



```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     87/100     0.81G     0.6814     0.5663      1.012         40        256: 100%|██████████| 6192/6192 [45:28<00:00,  2.27it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 355/355 [01:49<00:00,  3.26it/s]
                   all      11356      11663      0.806      0.508      0.641      0.515
  0%|          | 0/6192 [00:00<?, ?it/s]
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     88/100     0.81G     0.6775     0.5615      1.007         25        256: 100%|██████████| 6192/6192 [45:28<00:00,  2.27it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 355/355 [01:48<00:00,  3.27it/s]
                   all      11356      11663      0.802      0.515      0.642      0.515

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     89/100     0.81G     0.6754      0.556      1.009         39        256: 100%|██████████| 6192/6192 [45:06<00:00,  2.29it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 355/355 [01:48<00:00,  3.27it/s]
                   all      11356      11663      0.806       0.52      0.642      0.516
  0%|          | 0/6192 [00:00<?, ?it/s]
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     90/100     0.82G     0.6725     0.5479      1.004         38        256: 100%|██████████| 6192/6192 [45:26<00:00,  2.27it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 355/355 [01:50<00:00,  3.23it/s]
                   all      11356      11663       0.81      0.528      0.643      0.517
 Closing dataloader mosaic
```

5. Tried to fine-tune a different type of model that is based on Transformer. **(DETR model)** -> Because of its huge computation requirements, there is time and local device limit to perform such training in the current environments. This model will be deprecated unless there is other efficient way found.



1/26/2024 (Week 3)

1. Web Application Development
2. More detailed data processing is required
3. Research additional method to decide malignant or benign besides transformer

4. (Proposed) Cascade Model

Current problems:

1. Current single-model structure demonstrated inadequate performance
2. The imbalance between different classes is huge. Relying solely on data augmentation is not enough to address this problem

Cascade model

In a cascade model, the second model will only work on samples that are identified as “others” by the first model. The first model will be trained based on majority classes and “others”, which include all minority classes and (a portion of) unknown/others class. The second model will be trained only on minority-class samples and, if it exists, unknown/other-class samples. For an instance, if the first model identifies it as “others”, this instance will be passed to the second model for further analysis; if it is identified as belonging to one of the majority classes, the second model will be skipped. A simple diagram for a cascade model can be found below.

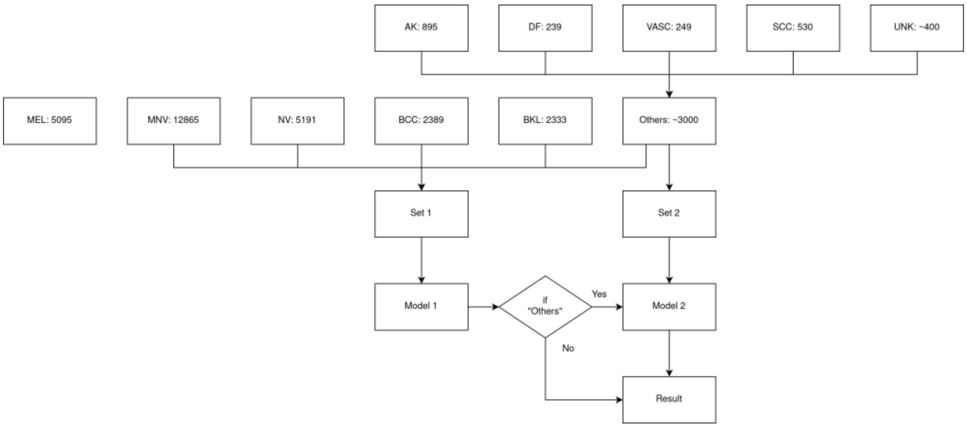


Figure: Cascade Model

Advantages:

1. Reduce the need for data augmentation, thus improving the quality of data
2. (Potentially) allow simpler models to be built, which would help improve efficiency and allow the entire model to be run on reasonable hardware

Disadvantages:

1. Lack of existing research on this approach
2. Low generalizability
3. Explanation could be difficult

4. Choice of architecture will have more impact on overall performance comparing to multi-model bagging ensemble

5. Literature review progress:

Planned: 50 papers in total, 20 about ensemble, 20 about CNN/Transformer, 10 about explainability

Ensemble part: the majority of studies so far used bagging. The other two types of ensemble learning seem to be inadequately investigated.

2/2/2024 (Week 4)

1. Literature review:

    a. Ensemble Part: (From literature review table)

        i. Ensemble learning models can significantly improve the predictive performance of DNN models, especially on imbalanced datasets.

        ii. Ensemble learning models perform better than single-model models on multiclass classification.

        iii. The inclusion of different models in ensemble learning models allows them to capture variations and more complex features in input images. This feature also facilitates residual learning and reduces the effect of vanishing gradient.

        iv. Apart from traditional ensemble techniques like (un)weighted averaging and majority voting, new techniques such as similarity-based were used and showed desirable performance.

        v. Combining CNN and transformer also improves performance.

        vi. All papers used transfer learning and stacking ensemble.

        vii. Imbalanced data, lack of data from population with varying skin tones and racial backgrounds, and lack of collaborations with clinicians are common issues.

        viii. Regularization and noise-removal issues are also mentioned.

ix. All papers used GPUs to train models.

x. Almost all papers used CNN. Only one paper used CNN and transformer.

b. Explainability Part: (From Paper)

i. **Decision-making mechanisms of DNNs are not transparent** due to their complexity and lack of robustness.

ii. There are **legal requirements** that medical devices should exhibit transparency.

iii. **Most articles only used XAI superficially**. Most of this half did not discuss the influence of XAI on model development or validation. Some cases, although also didn't discuss the influence, did specifically incorporated XAI techniques into the model. Many studies that use XAI only superficially did not include keywords like 'XAI' or 'explainability' but rather directly state which XAI algorithm they used.

iv. A few studies are related to technical improvements related to XAI, either through introduction of their own methods, or enhancing the capabilities of existing methods.

v. Some studies conducted elaborate evaluation in the context of XAI. A few studies compared the XAI-generated saliency maps against human-annotated segments. Two studies found that DNN models will learn to detect relevant human-interpretable features.

vi. XAI can **improve the predictive performance of human users** by a significant amount. Additionally, XAI also **increases users' confidence in prediction**; although this is unclear as sometimes users will report to be confident about the predicted results whether they are correct or not, and sometimes only when they are correct. **Very little is known about the influence of XAI on the predictive performance, confidence and model trust of dermatologists** in an artificial setting and nothing is known about its effects in a clinical setting. Sometimes, XAI results are even being ignored by clinicians.

vii. **Model fidelity** is a characteristic of an XAI method that describes how accurately its outputs reflect the inner workings of the explained classifier, i.e. how 'true' the explanations are to the explained model. Some XAI methods, such as Guided Backpropagation and GuidedGradCAM, are known not to be true to the explained model.

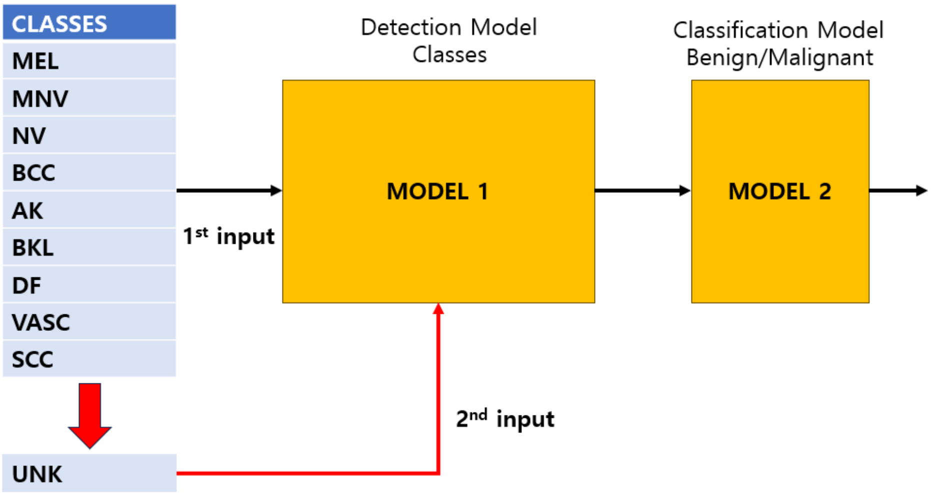
viii. However, due to the explained model being a black box, it cannot be determined how large the deviation between the approximated explanation and the real decision is when using post-hoc methods. One solution is **using inherently interpretable models**.

ix. **Post-hoc methods, heatmaps and prototypical explanations require human interpretation**: although a region is highlighted or similar images are shown, it is still up to the human evaluator to interpret why the region is relevant or why the images are similar. This likely introduces a **confirmation bias**.

x. **Similarity-based explanation was not investigated**.

2/9/2024 (Week 5)

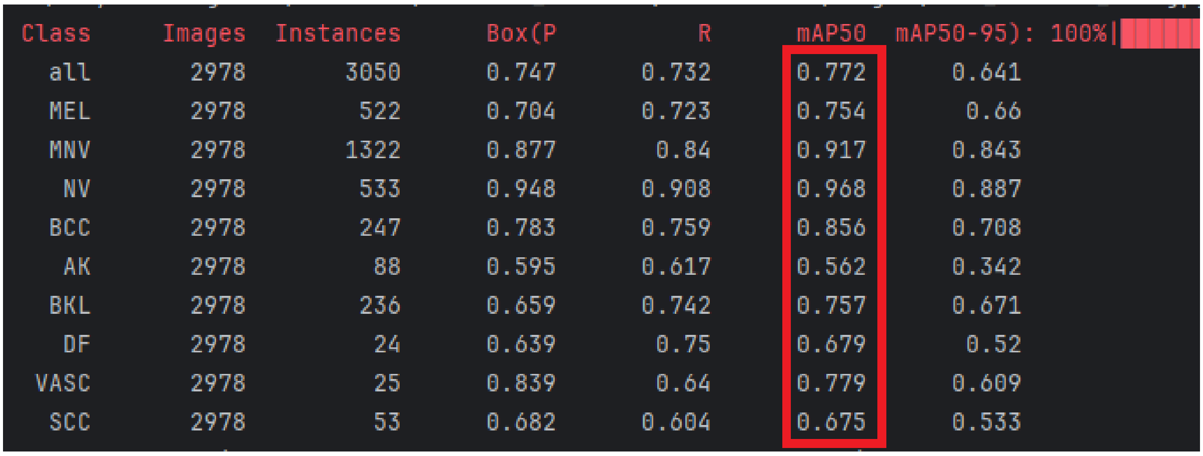
1. Data Seperation



The UNK (Unknown) class has been separated from the original dataset due to its unique characteristics. This class comprises skin diseases that have been verified as non-cancerous, but the specific type of disease has not been identified. The UNK class demonstrated a high correlation with other benign classes, which led to its exclusion from the initial training process of the first model. Instead, it will be used to train the second model.

This model takes the output from the first model and performs a classification task to determine whether the input is benign or malignant. At this stage, the UNK class can be included in the input and processed by the first model to train the second model.

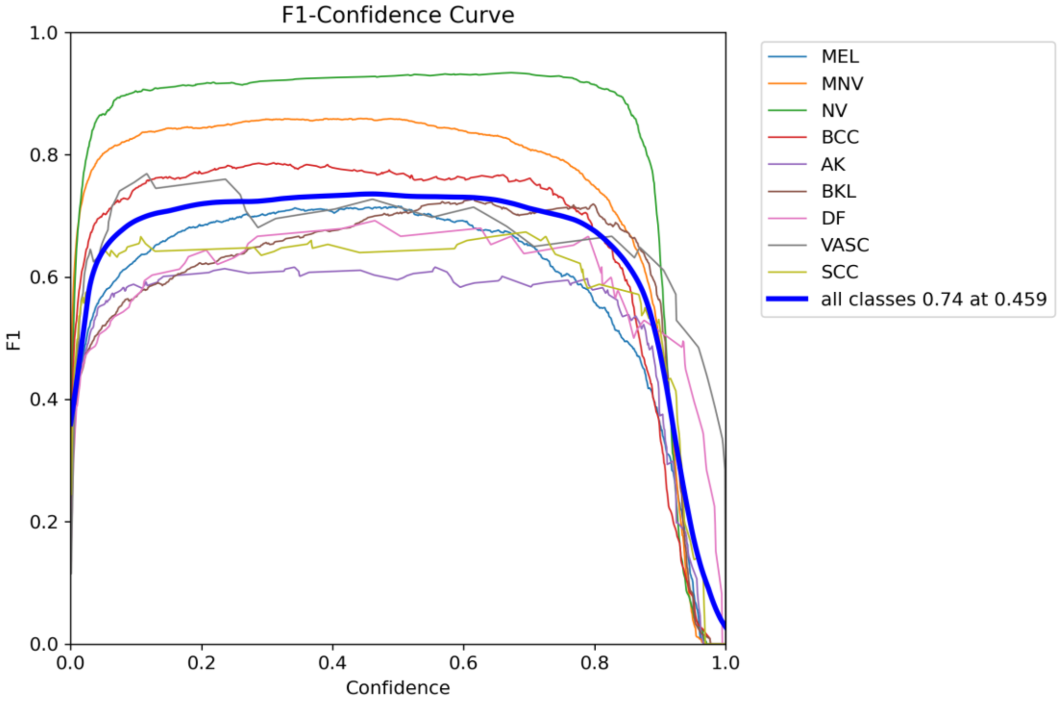
Results:



```
  Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|█████
    all       2978       3050      0.747      0.732      0.772      0.641
    MEL       2978        522      0.704      0.723      0.754       0.66
    MNV       2978       1322      0.877       0.84      0.917      0.843
     NV       2978        533      0.948      0.908      0.968      0.887
    BCC       2978        247      0.783      0.759      0.856      0.708
     AK       2978         88      0.595      0.617      0.562      0.342
    BKL       2978        236      0.659      0.742      0.757      0.671
     DF       2978         24      0.639       0.75      0.679       0.52
   VASC       2978         25      0.839       0.64      0.779      0.609
    SCC       2978         53      0.682      0.604      0.675      0.533
```

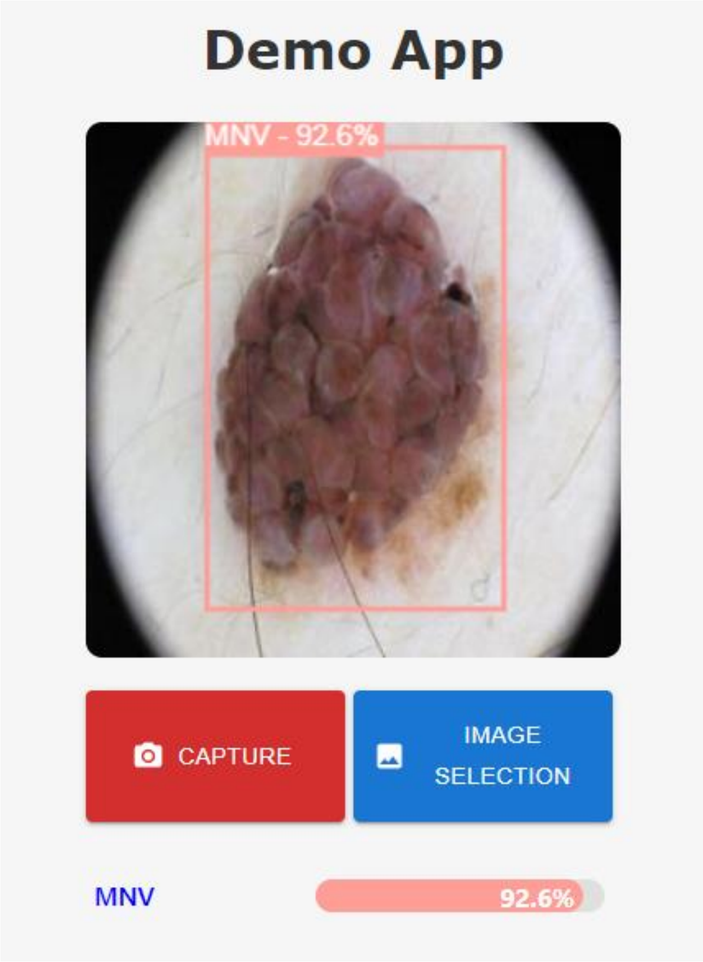
As a result, the average mAP50 scores across all classes have significantly increased to **0.772.** However, classes such as AK, DF, and SCC exhibited weaker performance compared to the others.

Experiments involving various settings and magnification factors were conducted, but the model 'train9', which excludes the UNK class, emerged as the best performing model to date.



The F-1 Score for all classes is **0.74**, achieved at a **0.459** confidence threshold. (train 9)

2. Progressive Web Application

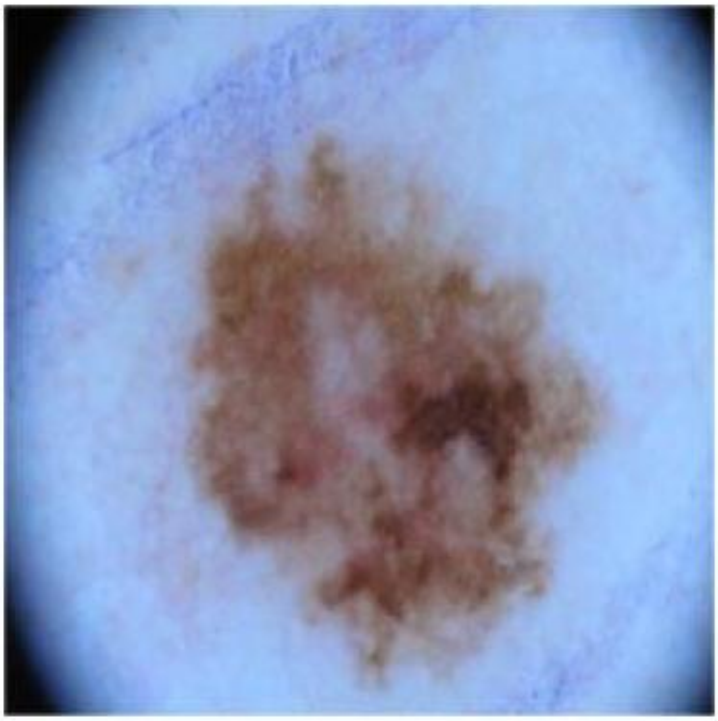


The demo version of the PWA is complete. Further development updates will be provided when there are significant advancements between the completed version of the model and the app. The design will also be updated.

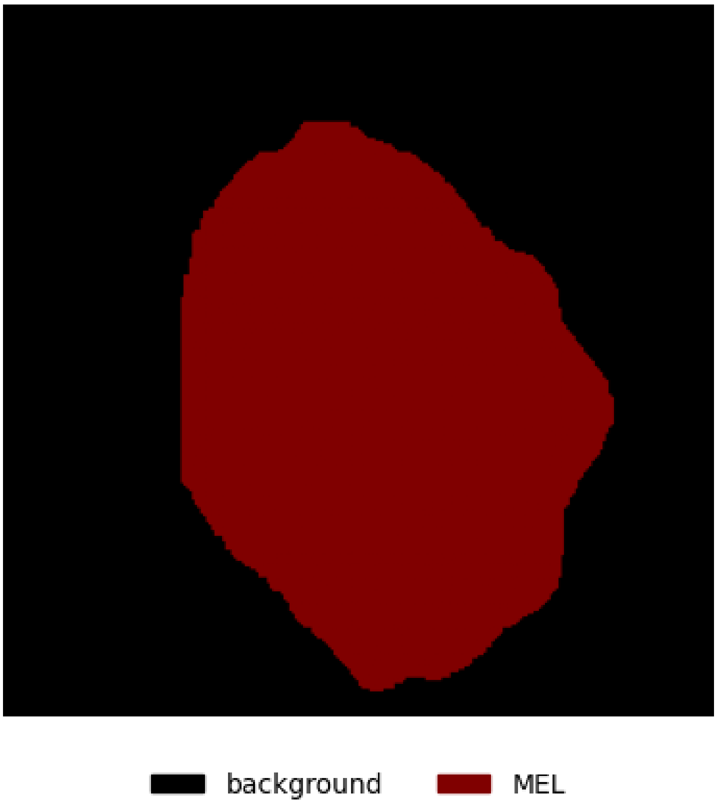
3. Transformer model progress (One Former, Mask2Former)

Two transformer models were trained on Vast.ai. Mask2Former exhibited poor compatibility and performance, leading to its deprecation. Consequently, only the **OneFormer** model is utilized for the final evaluation. This model was trained for up to 16 epochs with a customized sampler designed to reinforce data classes with fewer instances, such as the AK and SCC classes.

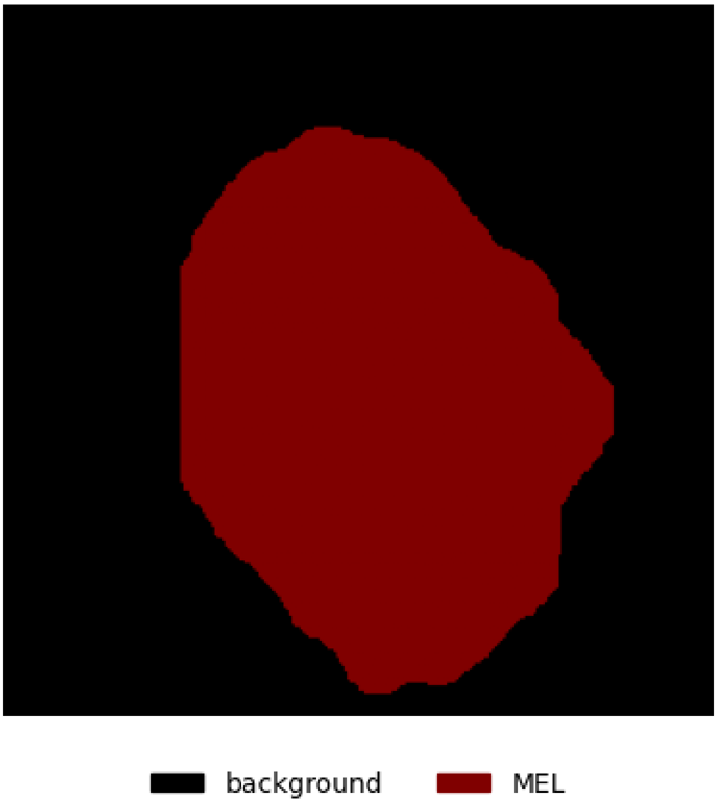
Sample Image



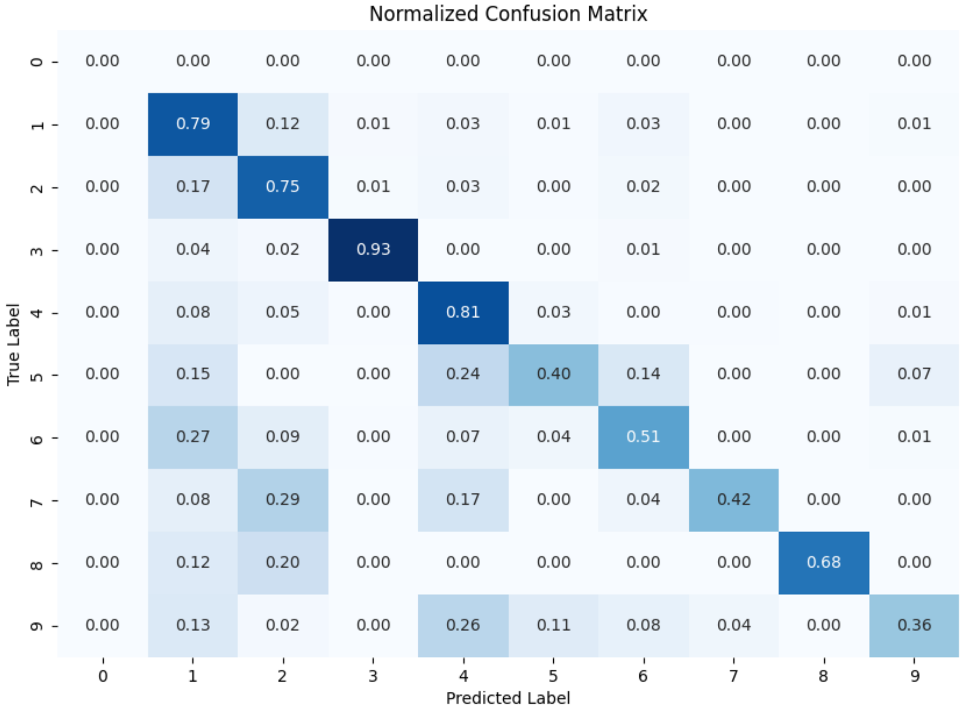
Actual Mask



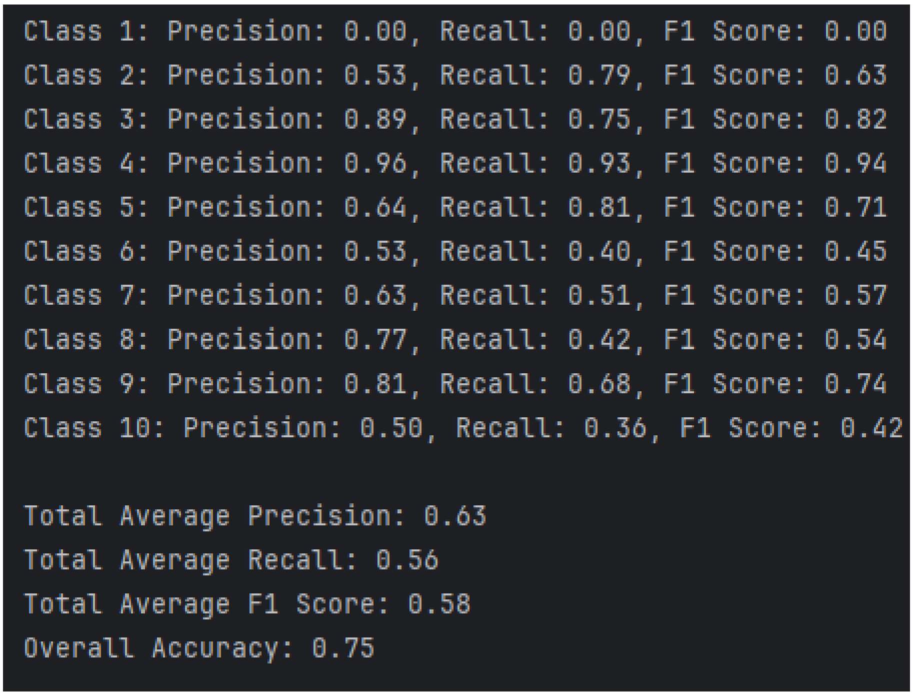
Predicted Mask



Confusion Matrix & Scores



	0	1	2	3	4	5	6	7	8	9
0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	0.79	0.12	0.01	0.03	0.01	0.03	0.00	0.00	0.01
2	0.00	0.17	0.75	0.01	0.03	0.00	0.02	0.00	0.00	0.00
3	0.00	0.04	0.02	0.93	0.00	0.00	0.01	0.00	0.00	0.00
4	0.00	0.08	0.05	0.00	0.81	0.03	0.00	0.00	0.00	0.01
5	0.00	0.15	0.00	0.00	0.24	0.40	0.14	0.00	0.00	0.07
6	0.00	0.27	0.09	0.00	0.07	0.04	0.51	0.00	0.00	0.01
7	0.00	0.08	0.29	0.00	0.17	0.00	0.04	0.42	0.00	0.00
8	0.00	0.12	0.20	0.00	0.00	0.00	0.00	0.00	0.68	0.00
9	0.00	0.13	0.02	0.00	0.26	0.11	0.08	0.04	0.00	0.36



```
Class 1: Precision: 0.00, Recall: 0.00, F1 Score: 0.00
Class 2: Precision: 0.53, Recall: 0.79, F1 Score: 0.63
Class 3: Precision: 0.89, Recall: 0.75, F1 Score: 0.82
Class 4: Precision: 0.96, Recall: 0.93, F1 Score: 0.94
Class 5: Precision: 0.64, Recall: 0.81, F1 Score: 0.71
Class 6: Precision: 0.53, Recall: 0.40, F1 Score: 0.45
Class 7: Precision: 0.63, Recall: 0.51, F1 Score: 0.57
Class 8: Precision: 0.77, Recall: 0.42, F1 Score: 0.54
Class 9: Precision: 0.81, Recall: 0.68, F1 Score: 0.74
Class 10: Precision: 0.50, Recall: 0.36, F1 Score: 0.42

Total Average Precision: 0.63
Total Average Recall: 0.56
Total Average F1 Score: 0.58
Overall Accuracy: 0.75
```

Class 1 represents the background and can be disregarded.

Similar to YOLOv8's difficulties with minority classes, the transformer model also faced challenges with classes that have smaller datasets, such as AK (6), BKL (7), DF (8), and SCC (10). A solution to mitigate this issue needs to be proposed.

4. Cascade Model Plan:

The Cascade Model Design will be the initial approach adopted to address the challenges presented by the imbalanced dataset for further progress. First implementation will be briefly done with YOLOv8 model.

5. GNN Model Plan:

Further research into the GNN models will be done.

2/16/2024 (Week 6)

1. **Search of GNN implementation of image classification:**

GNN architecture may not be feasible for image classification task
Currently, we have 2 most popular and up-to-date GNN libraries:
GNN and **DGL** libraries

PyG DGL DEEP GRAPH LIBRARY

The two frameworks mentioned do not support image classification tasks, nor do they facilitate the conversion of images into graph structures. Therefore, there is a limitation in implementing Graph Neural Network (GNN) models using these frameworks.

Vision GNN: An Image is Worth Graph of Nodes
https://arxiv.org/pdf/2206.00272.pdf
Graph Neural Network (GNN) in Image and Video Understanding Using Deep Learning for Computer Vision Applications
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9532631

"Vision GNN" and the other papers discuss the potential of GNN models for image classification tasks. However, their unique methods do not seem to be compatible with the most recent version of CUDA and related machine learning libraries, which would require downgrading and may not be preferable. Attempting to implement their methods, I encountered difficulties due to compatibility issues and the lack of detailed parameters and hyperparameters, making it inefficient to start the project from scratch without certainty.

Moreover, the most critical reason is that the transformer model concept is actually an upgraded version of the GNN architecture, according to the following article:

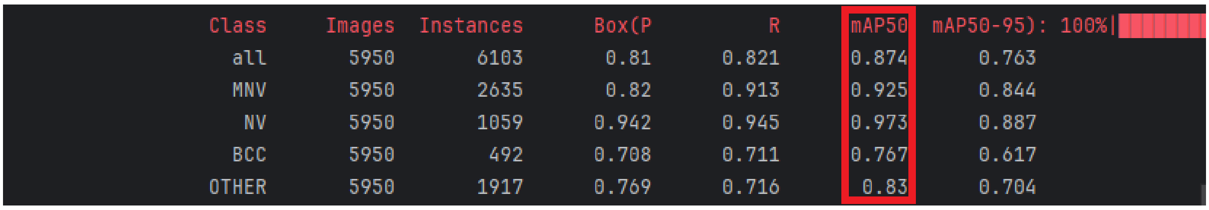
Transformers are Graph Neural Networks

https://thegradient.pub/transformers-are-graph-neural-networks/

Transformers can be considered an upgrade to Graph Neural Networks (GNNs) because they generalize the concept of neighborhood aggregation (a key component of GNNs) to the entire input sequence, treating sentences as fully connected word graphs. While GNNs aggregate features from immediate neighbors to update a node's representation, Transformers apply multi-head attention mechanisms to aggregate information from all words in the sentence, regardless of their positional relationship.

2. **Cascade Model implementation:**

First, its first trial went with the implementation of the cascade model

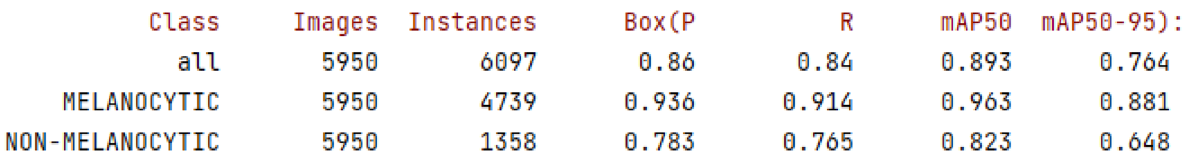


```
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|
                   all       5950       6103       0.81      0.821      0.874      0.763
                   MNV       5950       2635       0.82      0.913      0.925      0.844
                    NV       5950       1059      0.942      0.945      0.973      0.887
                   BCC       5950        492      0.708      0.711      0.767      0.617
                 OTHER       5950       1917      0.769      0.716       0.83      0.704
```

The first setting is to division of top 3 map score classes against the union of other smaller division classes. However, the result was not satisfactory since map score of OTHER class was still comparatively lower than other three classes.

Therefore, it came up with the idea to separate two big classes with common features before training separate classes. Instead of using TOP 3 map score classes, group the classes with: melanocytic and non-melanocytic according to the skin lesion grouping introduced by the paper:

https://www.sciencedirect.com/science/article/pii/S0031320320302168

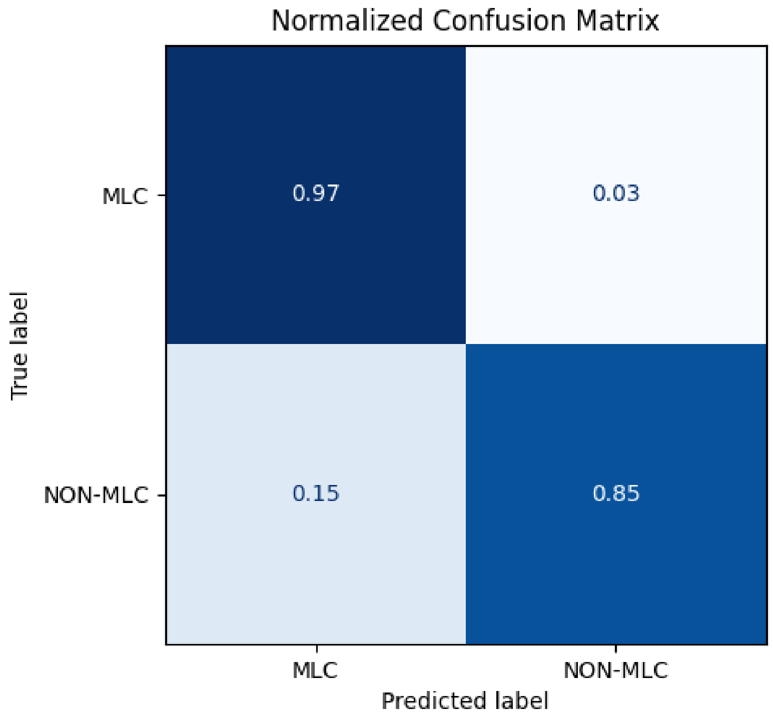


```
            Class     Images  Instances      Box(P          R      mAP50  mAP50-95):
              all       5950       6097       0.86       0.84      0.893      0.764
      MELANOCYTIC       5950       4739      0.936      0.914      0.963      0.881
  NON-MELANOCYTIC       5950       1358      0.783      0.765      0.823      0.648
```

However, the results were not satisfactory enough to address the low performance of the smaller division group (non-melanocytic). This class must achieve a higher score to improve performance in subsequent, more detailed classification tasks.

It's worth recalling that transformer models demonstrated superior performance in the decisive selection than the YOLO model. They managed to classify this broader category using a transformer image classification model, then later used the YOLO model to detect the more detailed classes for providing a more comprehensive detection representation later on.

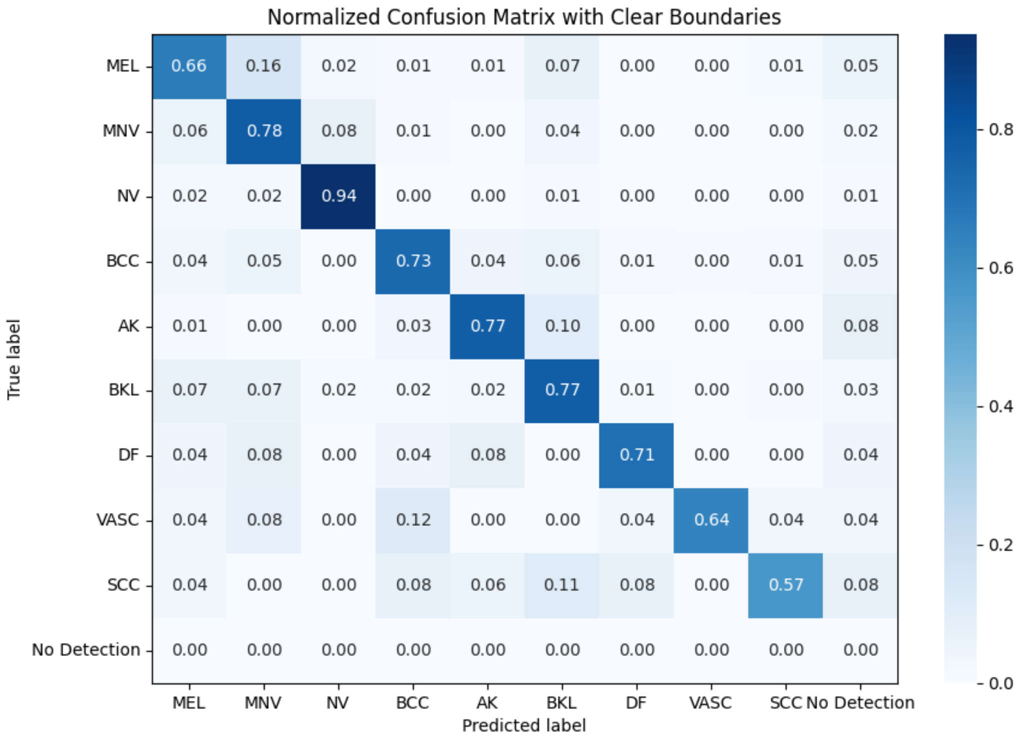
The SwinV2 model, one of the most recent image classification transformer models, was applied over 100 epochs for classifying melanocytic and non-melanocytic classes.



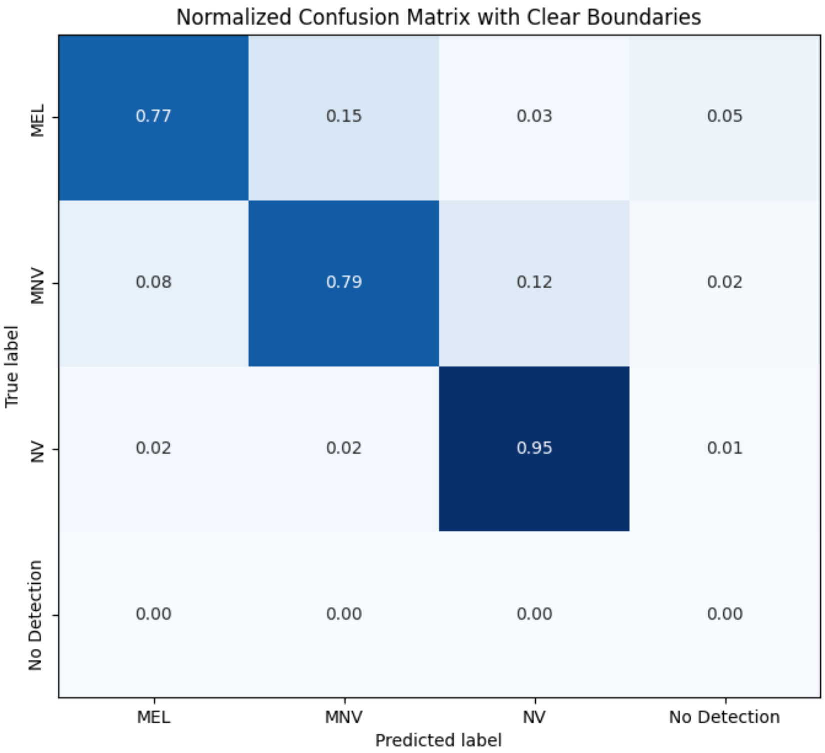
The results, however, were not feasible. The model still failed to show greatly improved performance in classifying minority classes from the larger classes.

Nonetheless, to verify the validity of this method, we will attempt to train more divided classes with the YOLOv8 model. This will help us determine if the model performs better when the dataset is refined by better separating the larger and smaller groups and thus mitigating dataset imbalance to enhance model training performance.

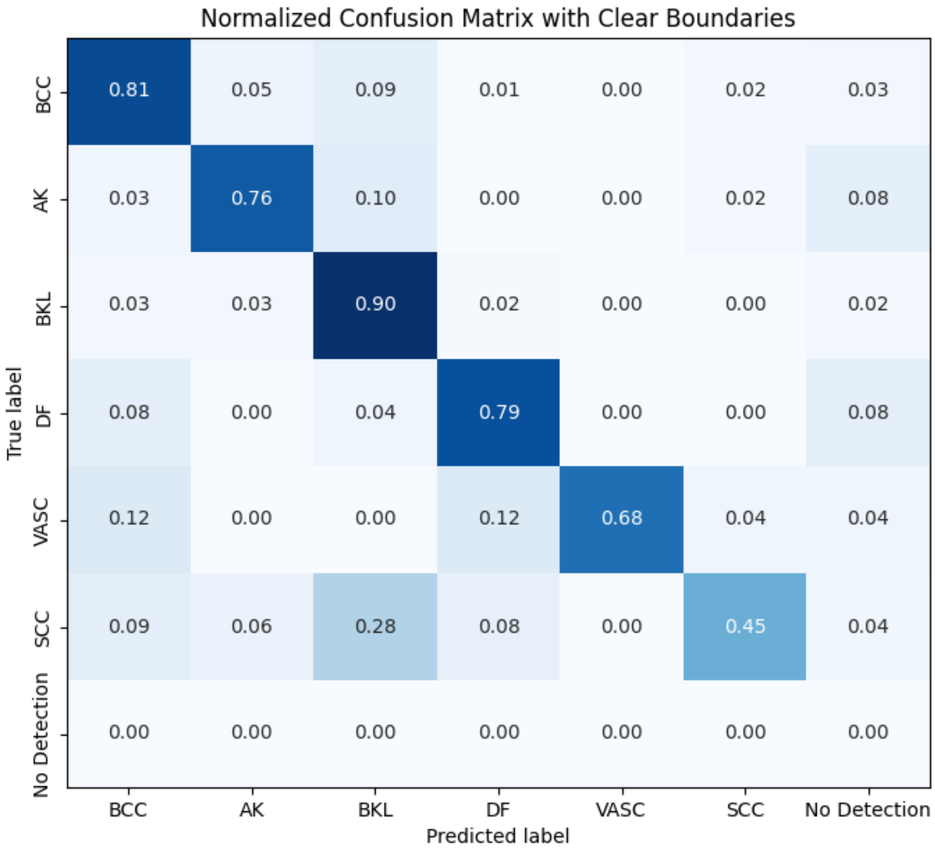
The YOLO original confusion matrix (confidence 0.35):



The YOLO Melanocytic division confusion matrix (confidence 0.35):



The YOLO Non-Melanocytic division confusion matrix (confidence 0.35):



	BCC	AK	BKL	DF	VASC	SCC	No Detection
BCC	0.81	0.05	0.09	0.01	0.00	0.02	0.03
AK	0.03	0.76	0.10	0.00	0.00	0.02	0.08
BKL	0.03	0.03	0.90	0.02	0.00	0.00	0.02
DF	0.08	0.00	0.04	0.79	0.00	0.00	0.08
VASC	0.12	0.00	0.00	0.12	0.68	0.04	0.04
SCC	0.09	0.06	0.28	0.08	0.00	0.45	0.04
No Detection	0.00	0.00	0.00	0.00	0.00	0.00	0.00

However, the results from the confusion matrix tell a differently. It indicates that the classes which showed weak performance in the scenario with the entire dataset displayed equal or even poorer performance in the divided environment. Therefore, we can infer that the issue lies within the dataset itself, rather than coming from the dataset's imbalance.

Summary of Models Used So Far:

- **YOLOv8 (**classification -> object detection + data augmentation**)** (**In Use**): Currently the most viable option.

- **GNN** (**Deprecated**): Due to a lack of compatibility and framework support, transformers are considered a better alternative.
- **OneFormer** (**Deprecated**): Exhibited poorer performance compared to the YOLOv8 model.
- **Mask2Former** (**Deprecated**): Demonstrated poorer performance than the OneFormer model.
- **Cascade Model** (YOLOv8 + SwinV2) (**Deprecated**): Classification between majority and minority groups does not perform well with the current settings. Additionally, classification in the divided environment also shows poor performance, making its intended purpose (to remove dataset imbalance) irrelevant. The poor performance is not attributed to dataset imbalance.

These trials seem sufficient for testing various model types and configurations. Implementation will cease from this week forward for a while. We will shift our focus to exploring methods utilized in other research for our study. Our approach will be more centered on search and analysis rather than implementation for the future improvement and implementation.

2/20/2024 (Week 7)

Meeting notes:

1. Some cancer types are not biopsied. Therefore, doctors propose that focus on MEL and MNV
2. Use real dataset (Amy's) to validate
3. **Time-series data**
    a. **Stability over time**
    b. **Difference between time stamps**
4. Target: Rural areas – primary care & Tele-demorscopy

2/26/2024 (Week 8)

Project focus: YOLOv8 + Multi-explanation (**Deprecated**)

Kim's writing on innovation (as reference):

Developing an explanatory machine learning framework that can provide explanations for its decisions based on the concepts and features of the input is crucial. This can be achieved in various ways; however, a simple post hoc method is commonly used to analyze machine decisions to understand the rationale behind them. Nonetheless, merely applying a post hoc method to the model's final decision may not always be sufficiently trustworthy, as it sometimes lacks the information needed to explain why the decision exhibited a particular tendency. The model could often misinterpret specific parts of the data, and it lacks information on how humans can understand the information from the interpretation result it provides.

Therefore, instead of merely applying interpretability techniques to the final result, we will develop a novel architecture that can subdivide the feature maps from the machine learning models into concepts of feature maps to perform concept-based interpretation. This architecture will analyze each concept of feature map to determine how likely it is that the feature map contains the characteristics of such a concept The decisions made by the thresholds in each concept will not only help the machine make the final decision but also assist users in comprehending why the information is presented in a certain way. There has been few research in this project, so our goal is to add more detailed information on each concept with models of better performance.

In addition, developing a lightweight architecture to enable this model in computationally limited environments is also a critical aspect since the models must be usable in various settings, including mobile devices with low computational power. Therefore, using machine learning architectures that require substantial computation and memory, such as transformers, is discouraged. Instead, a moderately sized CNN architecture that can make accurate decisions should be adopted. The structure of the system or application should be appropriately designed to integrate this architecture efficiently.

We will utilize state-of-the-art computer vision machine learning models for various purposes. First, it is mandatory to have a machine learning model capable of image segmentation to precisely identify which section of the image is a skin lesion when the image is passed to the model. This model could be a state-of-the-art CNN model, such as YOLO. Additionally, the backbone of another feature extraction model can be used to analyze the features of the detected section with more detailed information from a different dataset. Then, these extracted features will be passed to interpretation architecture, which can detail the image into each concept helping us gain interpretability information.

Datasets: PH2 and Derm7pt

3/5/2024 (Week 9)

Explainability update:

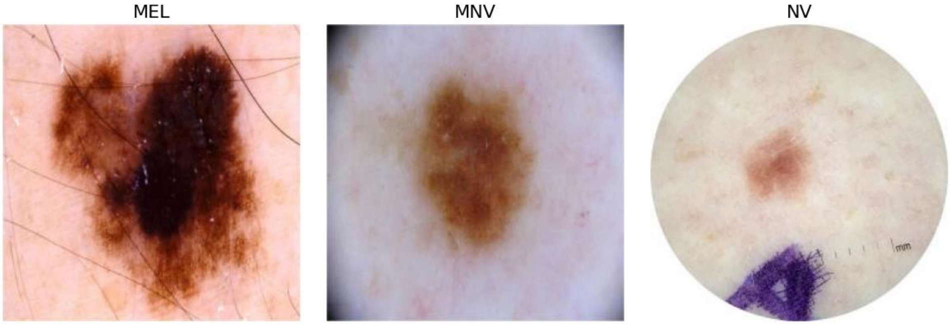
- Although smoothgrad is the preferred choice, no existing library supports it. For pytorch models, the PAIRML library supports smoothgrad. However, the example it provided is only for Inception V3. Since YOLOv8's architecture is completely different from Inception V3, simply changing the model won't work. More time is needed to do further experiments because the library requires Softmax activation layer, which YOLOv8 does not have.
- ~~Right now, the only working library is the YOLOv8-Explainer. It has GradCAM and some variations, but no smoothgrad.~~ YOLOv8-Explainer, for some unknown reason, ~~doesn't work with certain images. Unfortunately, our skin cancer data belongs to the group that doesn't work with this library.~~ any method but EigenCAM does not work because of gradient-related issue.
- Pytorch-grad-cam library does not work with YOLOv8 since it is not uploaded to pytorch hub. It only supports YOLOv5, which exists on pytorch hub.
- SHAP also doesn't work. It looks like the main problem is caused by the use of pre-trained model. Maybe we need to build a model from scratch
- Build-from-scrach YOLOv8 using Keras: https://www.kaggle.com/code/fitztata/yolov8-asl-recognition-model-explainability

3/14/2024 (Week 10)

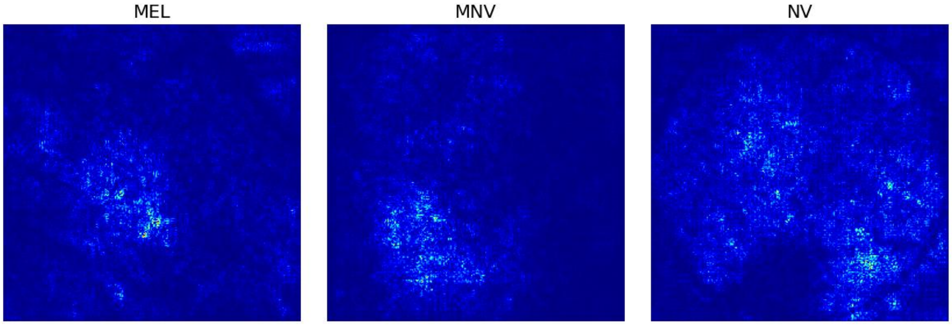
Explainability update:

- Customized model based on YOLOv8 backbone: https://colab.research.google.com/drive/1-F3mElXO3rokO913AsqCqCPeOs38uY76?usp=sharing
- Metrics are implemented
- tf-explain libraries didn’t work. tf-explain haven’t been updated since 2021.
- tf-keras-vis works without using image-loading function from TensorFlow library. **It also reveals that the model is relying on completely irrelevant features. The problem is likely to be caused by the lack of supervision on feature extraction (i.e. layer.trainable = true)**

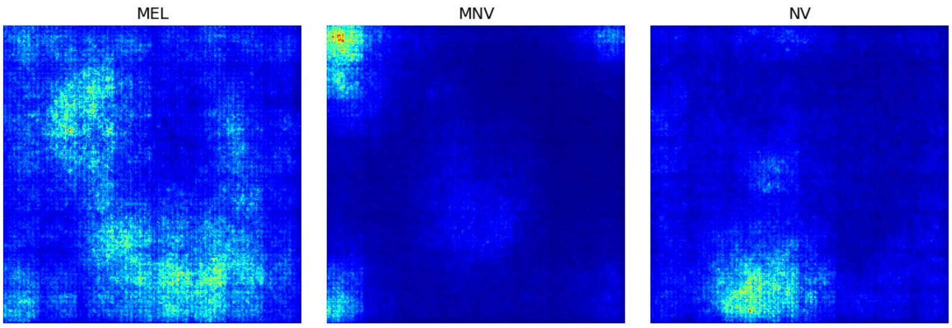
Test Images:



Saliency Map/Integrated Gradients:



Smoothgrad:



- Customized GradCAM in the notebook should work. Modifications can be done to change it to smoothgrad. Currently, it ran out of available memory.

3/22/2024 (Week 11)

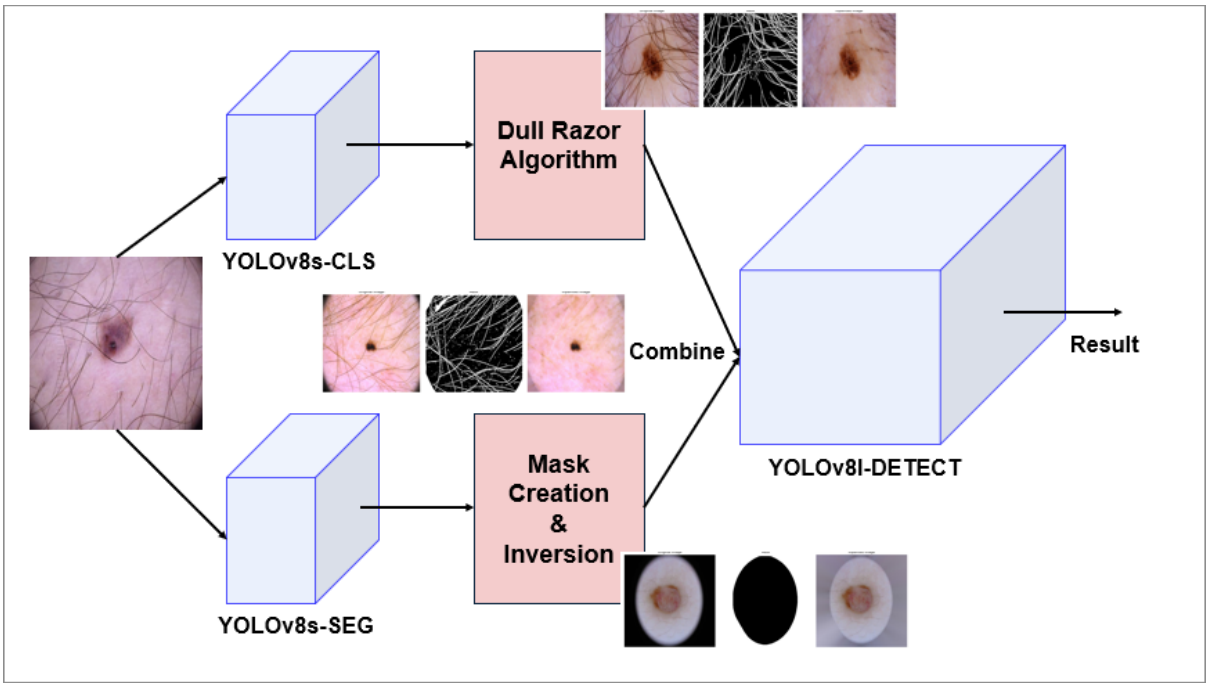
- GradCAM and GradCAM++ were tested. Results are contradictory.
- SwinTransformerV2, ResNetXt50, EfficientNetV2 were also tested. It seems that changes in model do not have significant impact on explanations.
- Paper (https://www.sciencedirect.com/science/article/pii/S095741742301549X) used several image segmentation techniques, namely 1) threshold method, 2) edge detection, 3 region-growing method, 4) clustering method, 5) U-Net, and 6) RP-Net, in an ensemble setup.
- Paper (https://dl.acm.org/doi/abs/10.1007/978-3-030-80432-9_1) simply cropped images.
- Some papers used hand-crafted features or noise-removal techniques.
- A technique called HR-CAM seems to be able to avoid noise in images.
- Most papers don’t show explanations on noisy images.

3/29/2024 (Week 12)

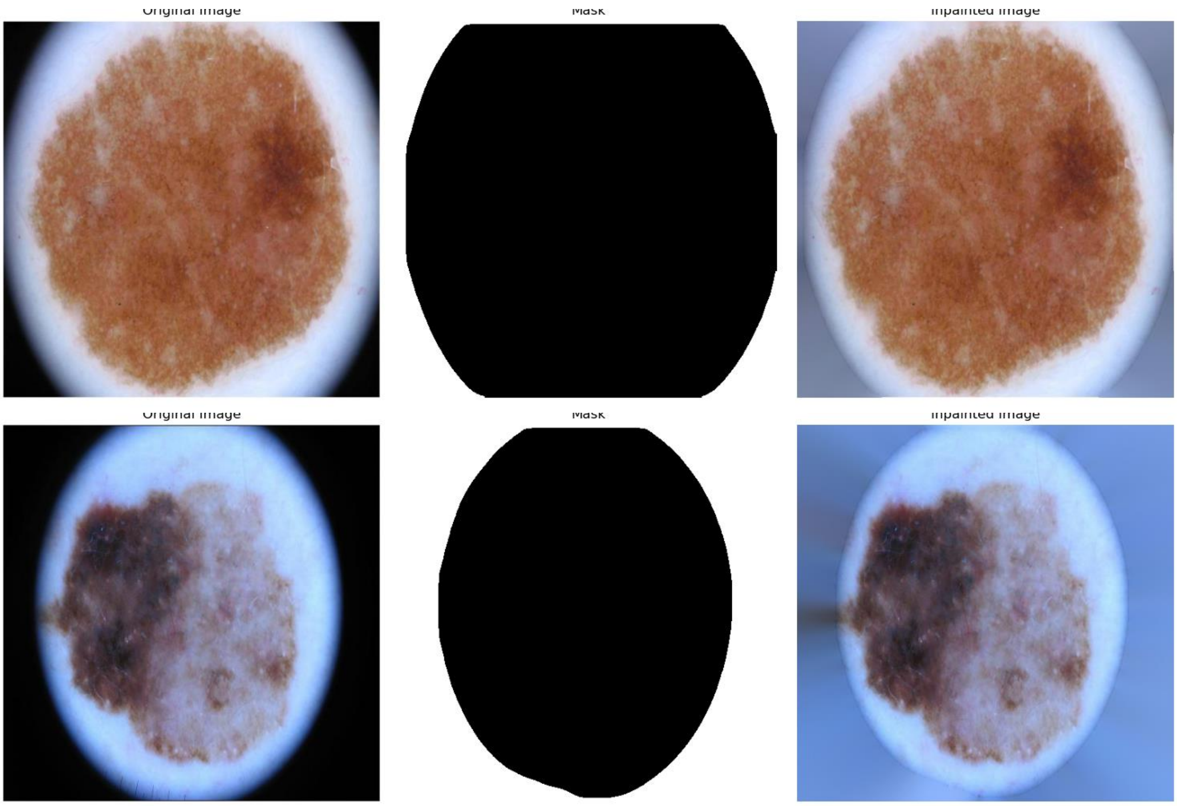
- After using DullRazor to remove artifacts from images, smoothgrad can correctly recognize skin lesion regions. However, other methods are broken. GradCAM and GradCAM++ both failed to recognize regions, which is curious as both can recognize regions without removing artifacts. This effect could be due to the blurring caused by artifact removal.

4/12/2024 (Week 14)

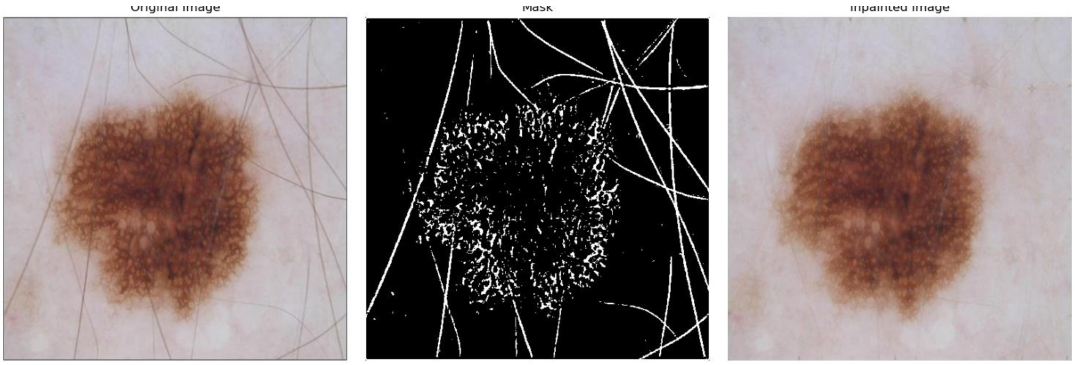
Confounding Factor Removal pipelines are successfully implemented. It utilizes the structures which are visualized below. Yet, there is still room for improvements, but it will be detailed later after the second stage of the research is completed.

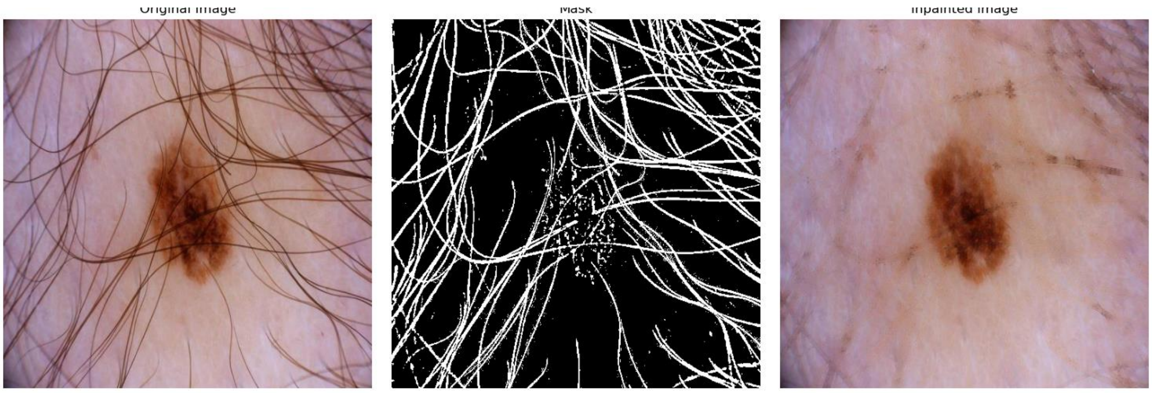


DCA removal sample

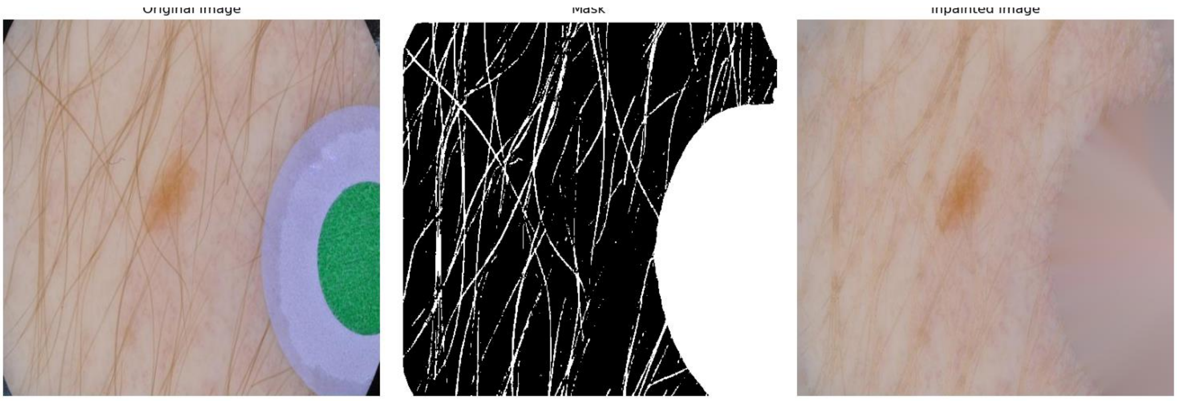


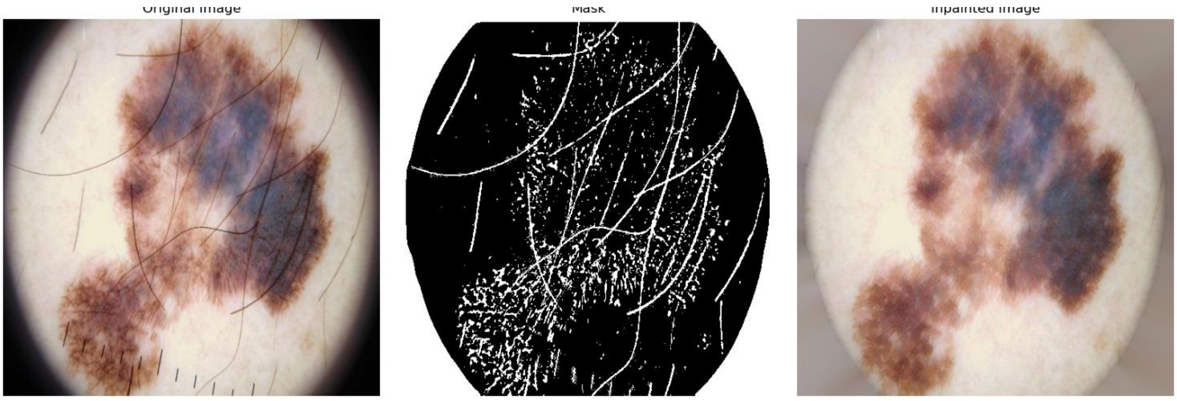
Hair removal sample



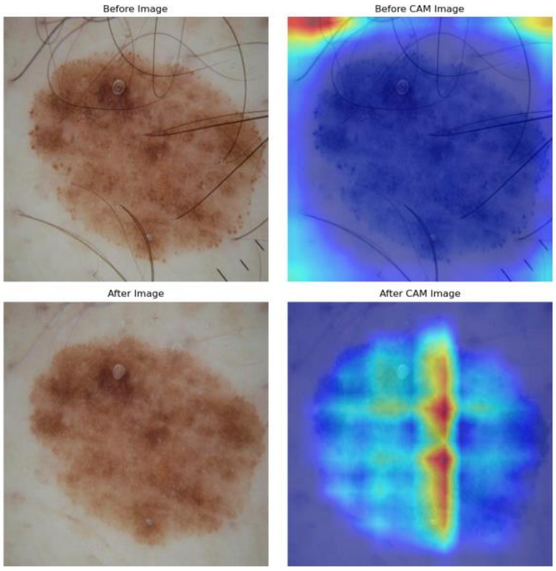


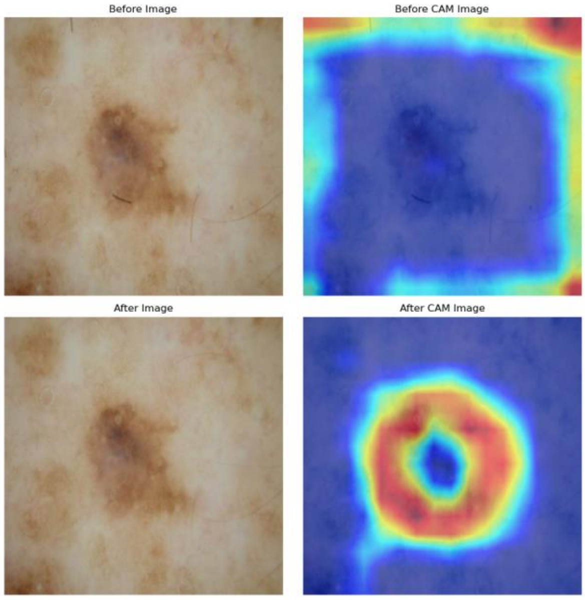
DCA+Hair removal sample



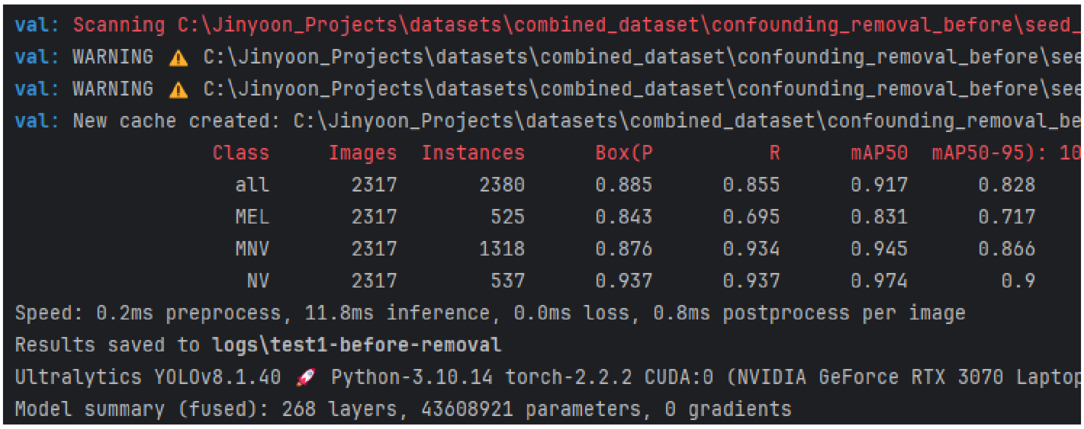


Cam Sample (before, after)

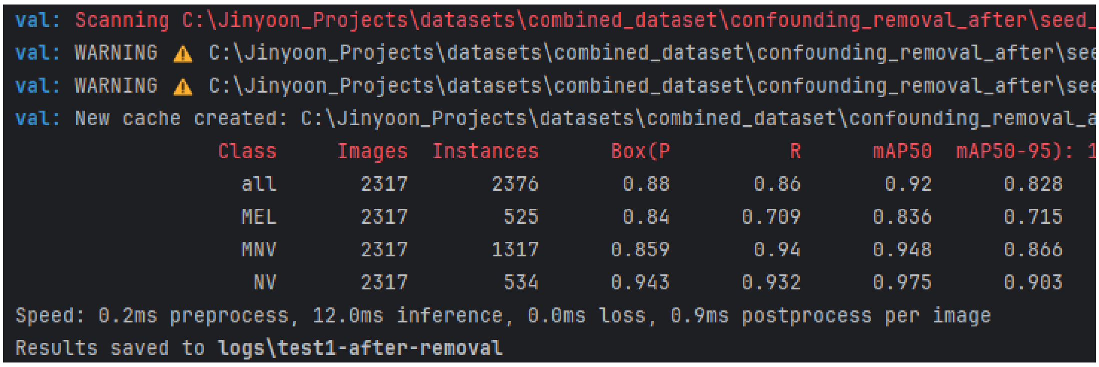




Model Performance Change Result



```
val: Scanning C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_before\seed_
val: WARNING ⚠ C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_before\see
val: WARNING ⚠ C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_before\see
val: New cache created: C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_be
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 10
                   all       2317       2380      0.885      0.855      0.917      0.828
                   MEL       2317        525      0.843      0.695      0.831      0.717
                   MNV       2317       1318      0.876      0.934      0.945      0.866
                    NV       2317        537      0.937      0.937      0.974        0.9
Speed: 0.2ms preprocess, 11.8ms inference, 0.0ms loss, 0.8ms postprocess per image
Results saved to logs\test1-before-removal
Ultralytics YOLOv8.1.40 🚀 Python-3.10.14 torch-2.2.2 CUDA:0 (NVIDIA GeForce RTX 3070 Laptop
Model summary (fused): 268 layers, 43608921 parameters, 0 gradients
```



```
val: Scanning C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_after\seed_
val: WARNING ⚠ C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_after\see
val: WARNING ⚠ C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_after\see
val: New cache created: C:\Jinyoon_Projects\datasets\combined_dataset\confounding_removal_a
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 1
                   all       2317       2376       0.88       0.86       0.92      0.828
                   MEL       2317        525       0.84      0.709      0.836      0.715
                   MNV       2317       1317      0.859       0.94      0.948      0.866
                    NV       2317        534      0.943      0.932      0.975      0.903
Speed: 0.2ms preprocess, 12.0ms inference, 0.0ms loss, 0.9ms postprocess per image
Results saved to logs\test1-after-removal
```