# Quarterly Report of Oraiclebio Project

Index Page

Project Execution Plan (PEP File)

RoadMap :

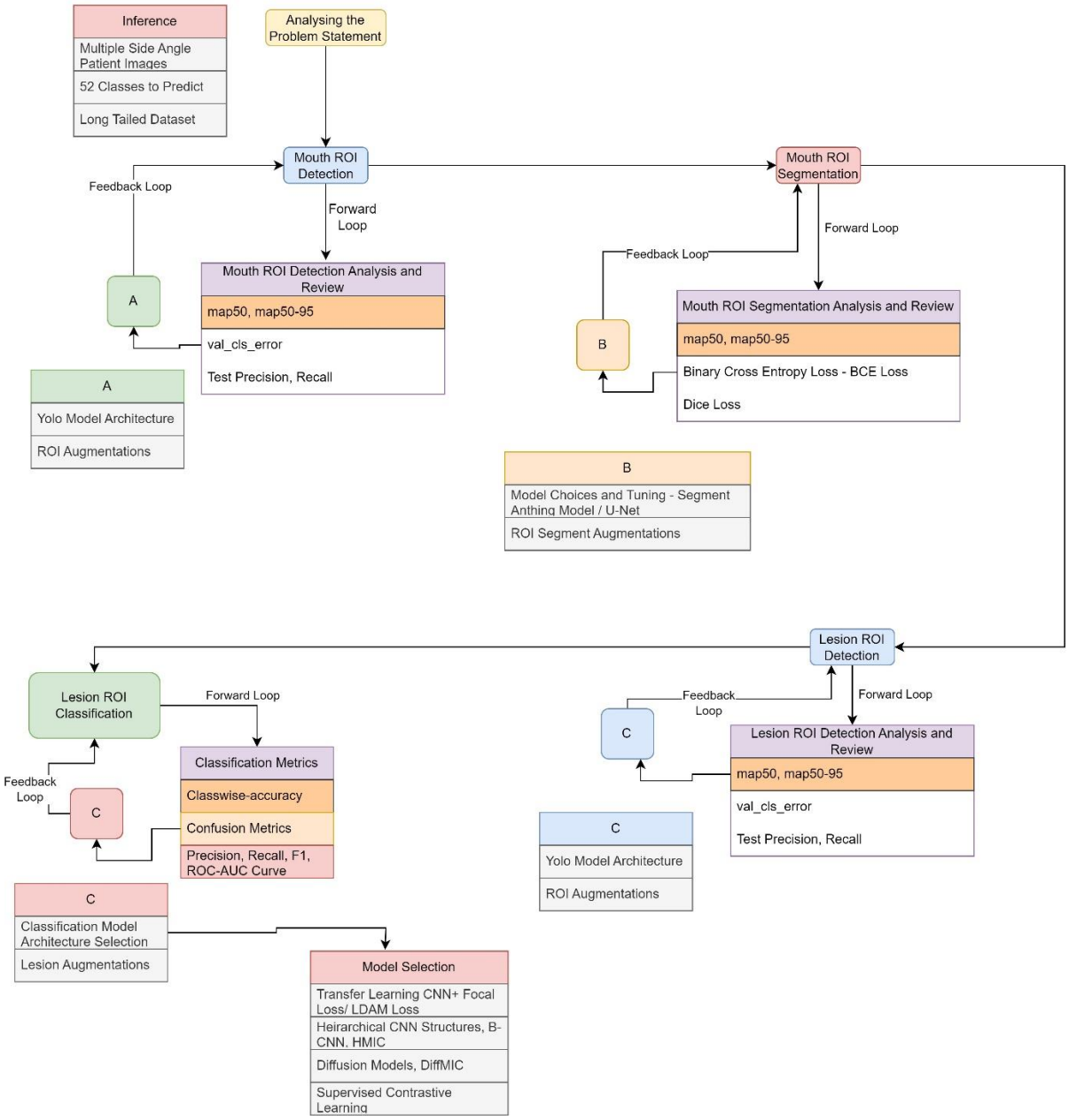



Fig. 1 Roadmap for Implementing the Mouth ROI Project

Problem Analysis

1) Analysing the Problem Statement

Problem Statement: Annotating oral-precancerous and cancerous lesions through Machine Learning methods

## 1.a. Analysing the Classes

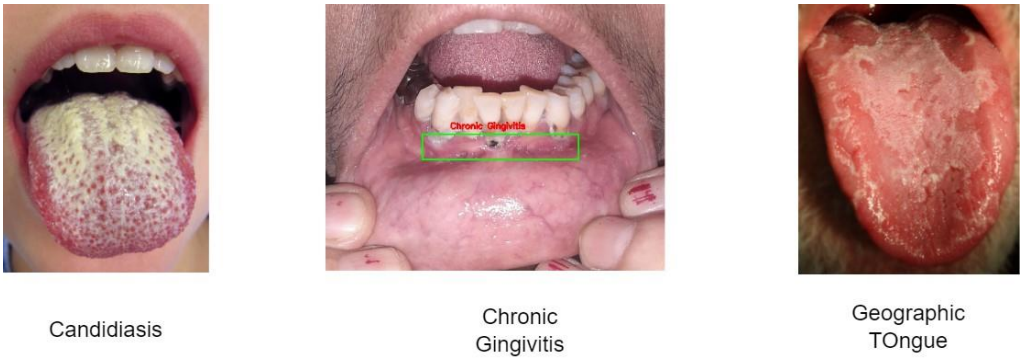



Fig. 2 Sample Classes

List of Classes with short descriptions

1. Chronic Gingivitis: Persistent inflammation of the gums.
2. Candidiasis: Fungal infection causing white lesions.
3. Traumatic Ulcer: Injury-induced open sore.
4. Geographic Tongue: Benign, migratory patches on the tongue.
5. Gingival Enlargement: Abnormal gum growth.
6. Frictional Hyperkeratosis: Thickening of the oral mucosa due to friction.
7. Periapical Cyst: Cyst near the root of a tooth.
8. Erythroplakia: Red, potentially precancerous lesion.
9. Fordyce Granules: Yellowish spots, often on lips.
10. Lichen Planus: Inflammatory condition affecting oral tissues.
11. Denture Stomatitis: Inflammation under dentures.
12. Aphthous Stomatitis: Recurrent, painful mouth ulcers.
13. Proliferative Verrucous Leukoplakia: Precancerous, thick white patches.
14. Varicosities: Enlarged blood vessels.
15. Torus Mandibularis: Bony growth on the lower jaw.
16. Physiologic Pigmentation: Normal oral discoloration.
17. Squamous Cell Carcinoma: Aggressive form of oral cancer.
18. Melanoplakia: Dark patches on oral mucosa.
19. Pyogenic Granuloma: Noncancerous growth due to irritation.
20. Actinic Cheilitis: Sun-induced lip inflammation.
21. Hairy Tongue: Abnormal coating, often due to poor oral hygiene.
22. Atrophic Glossitis: Inflammation causing a smooth tongue.
23. Pericoronitis: Inflammation around a partially erupted tooth.
24. Angular Cheilitis: Cracks or sores at the corners of the mouth.
25. Parulis: Inflamed gum tissue opening on the skin.
26. Mucocele: Fluid-filled cyst, often on the lip.
27. Osteogenic Tumors: Tumors originating from bone tissue.
28. Epulis Fissuratum: Overgrowth of tissue due to denture irritation.
29. Hemangioma: Benign blood vessel tumor.
30. Pemphigus: Autoimmune blistering disorder.
31. Erythema Migrans: Red, migratory patches on oral mucosa.
32. Tobacco Pouch Keratosis: White patches due to smokeless tobacco.
33. Peripheral Ossifying Fibroma: Gum tumor containing bone.

34. Speckled Leukoplakia: White and red patches, potentially precancerous.
35. Fissure Tongue: Deep grooves on the tongue surface.
36. Submucous Fibrosis: Progressive fibrosis affecting the oral mucosa.
37. Exostosis: Bony growth on the surface of bone.
38. Nicotine Stomatitis: Inflammation due to tobacco smoke exposure.
39. Oral Lichenoid Reaction: Oral lesions resembling lichen planus.
40. Leukoplakia: White patches, often precancerous.
41. Oral Nevi: Benign pigmented lesions.
42. Contact Stomatitis: Inflammation due to contact with irritants.
43. Circumvallate Papillae: Large taste buds on the back of the tongue.
44. Herpes Gingivostomatitis: Oral herpes infection.
45. Squamous Papilloma: Benign growth, often on the tongue.
46. Tobacco Induced Melanosis: Dark pigmentation due to tobacco use.
47. Irritation Fibroma: Overgrowth of tissue due to irritation.
48. Smoker's Melanosis: Oral discoloration due to smoking.
49. Localized Aggressive Periodontitis: Rapid, localized gum disease.
50. Leukoedema: Benign, whitish appearance of the oral mucosa.
51. Linea Alba: Raised white line on the inner cheek.
52. Foliate Papillae: Grooved structures on the sides of the tongue.

Number of Images per class.

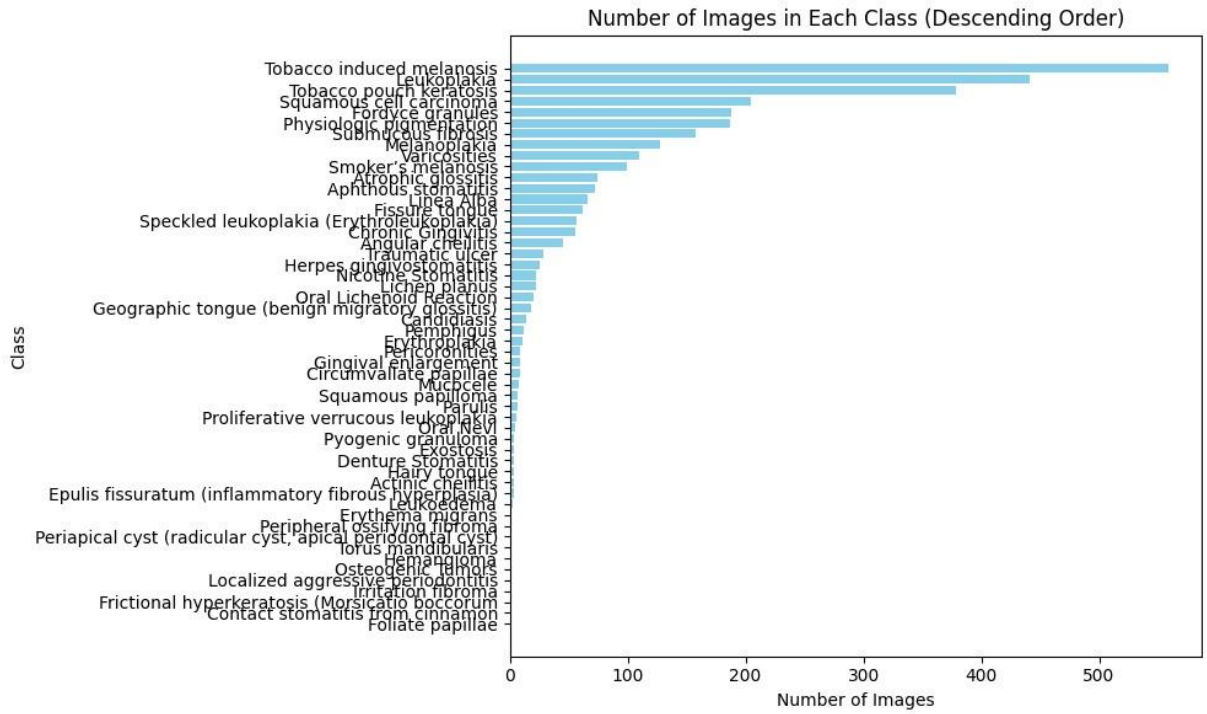



Fig. 3 Long tailed Dataset

This shows it's a long tailed dataset.

## 1.b. Analysing the Images

Variation among the Images:-

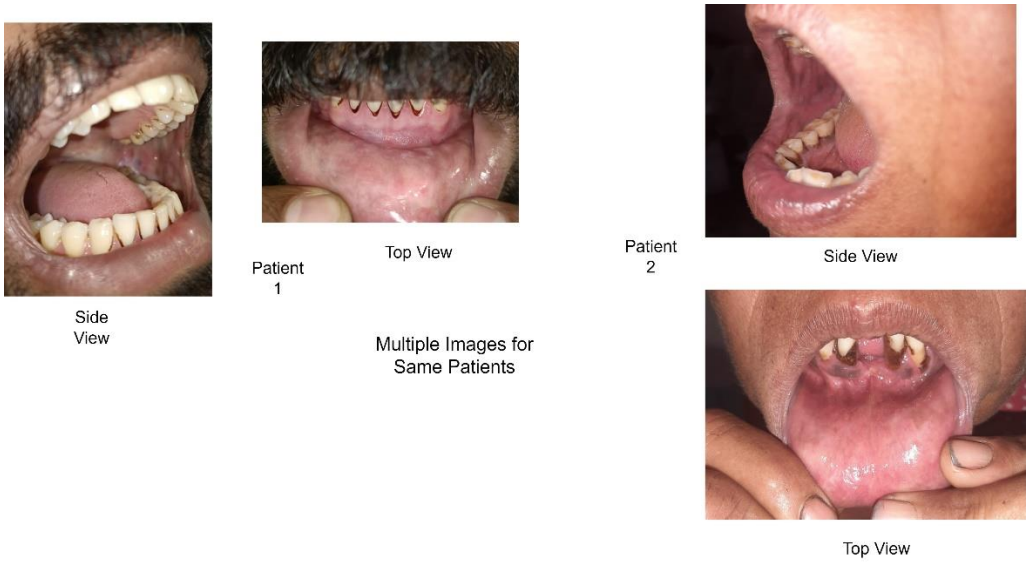



Fig. 4 Multiple Angles for Same Patient

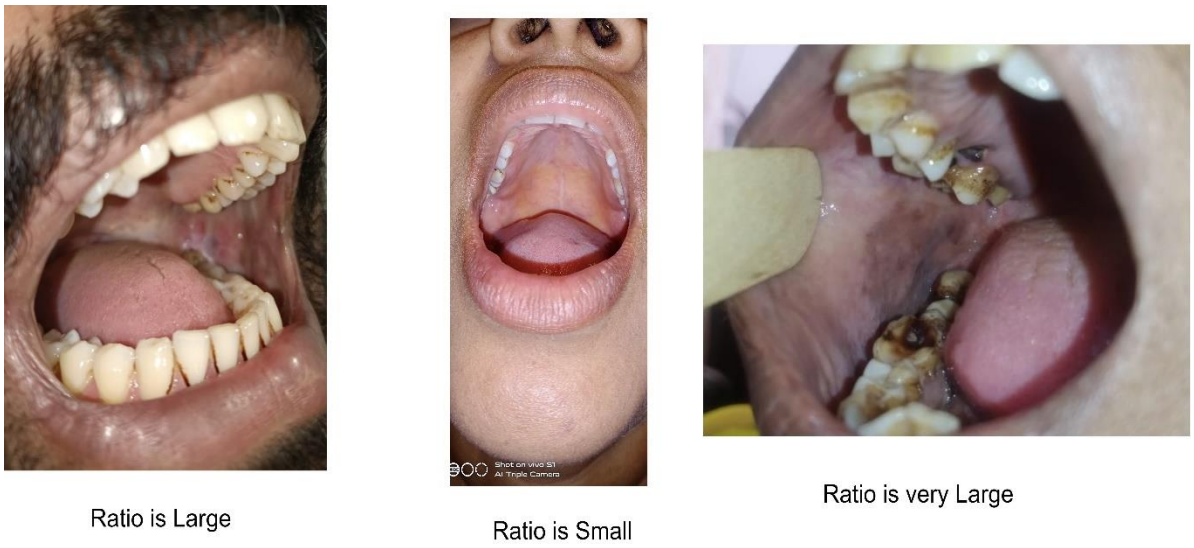



Fig. 5 Mouth to Image Aspect Ratio Variation

After iterating throughout all the images, observed in the dataset, problems with image predictions:

1. Inclusion of Non-Oral Cavity Areas:

The presence of non-oral cavity areas in multiple images, when considered along with the mouth to image aspect ratio, poses a significant challenge for detection algorithms. These non-oral cavity areas introduce additional complexity and variation into the images, potentially leading to difficulties in accurately locating and identifying the classes of interest. The varying sizes and shapes of these areas, combined with the diverse lighting angles present in the images, further exacerbate the detection problems.

2. Blurry Photos / Light source direction variations

In general the light sources vary from multiple directions, leading to blurriness, and general image distortions, which could cause multiple problems during object localization and classification.

3. Undetected Multiple Lesions:

The variability of light sources in images, emanating from multiple directions, introduces challenges such as blurriness and general distortions that can impact object localization and classification tasks. These variations can result in inconsistent image quality, making it difficult for algorithms to accurately identify and delineate objects of interest. The presence of blurriness and distortions can obscure important features, leading to misinterpretation or misclassification of objects within the image.

After this analysis, it is clear that the data we are working with has a lot of randomness, and is naturally long tailed. Therefore a one-shot model will not work, as it wont take into account these intricacies required for model prediction.

We propose the following algorithm. First, we detect the mouth ROI (Region of Interest). This is done using detection and segmentation models. Then, we perform legion detection on these images, to extract the Lesion ROIs. These lesion ROI's are used for training and testing on multiple classification models, specifically tailored for the long tailed cause. We will discuss these in detail later on. Lets discuss the first model, i.e. the mouth ROI Detection.

## Mouth ROI Detection

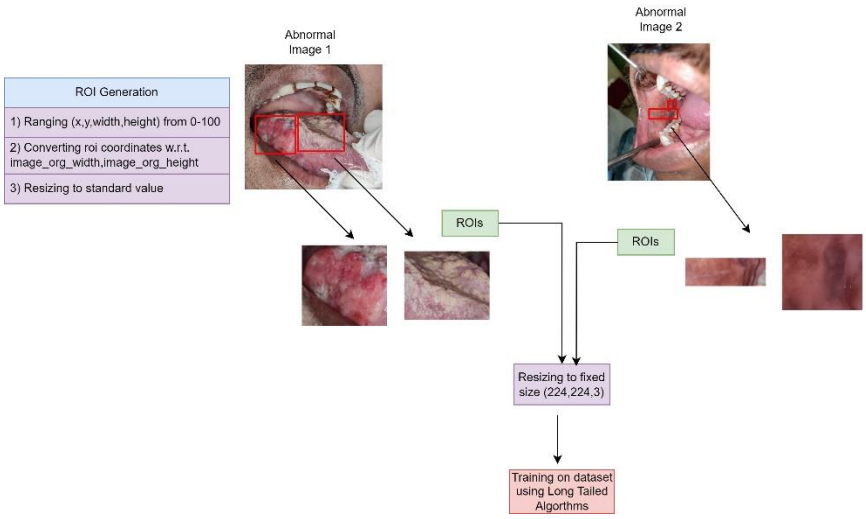



Fig. 6 Data Standardization to take care of Uneven ROI sizes

## Model Used:

Yolo v8. Some of the reasons include:

1) Speed: YOLO models are known for their speed, as they can detect objects in real-time or near real-time on a GPU.

2) Single-stage detection: YOLO is a single-stage detector, which means it makes predictions on the full image in one evaluation, making it faster than two-stage detectors like Faster R-CNN in some scenarios.

3) Good performance: YOLO models have shown good performance on object detection benchmarks in terms of accuracy and speed.

4) Ease of use: YOLO models are relatively easy to implement and use, especially with popular frameworks like Darknet, PyTorch, and TensorFlow.

YOLO (You Only Look Once) is preferred over Faster R-CNN for several reasons. Firstly, YOLO is faster due to its single-pass approach, making it suitable for real-time applications like video processing. Secondly, its simplicity makes it easier to implement compared to the multi-stage architecture of Faster R-CNN. These factors make YOLO a compelling choice for tasks that require speed and simplicity without compromising much on accuracy.

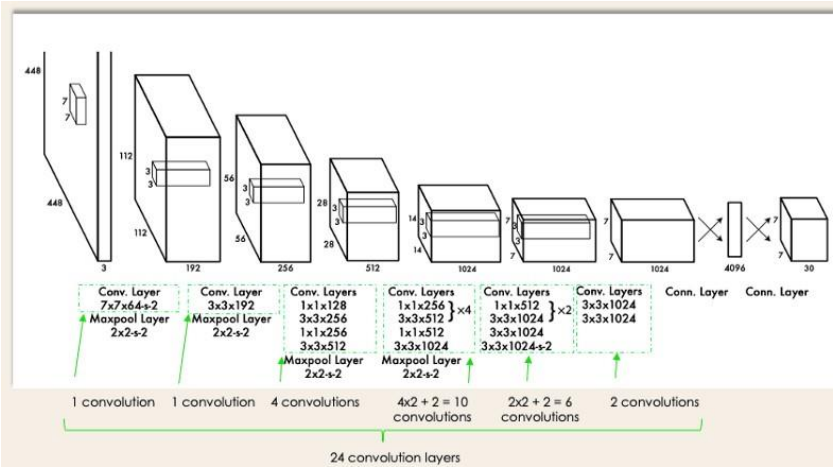



Fig. 7 Yolo Model Architecture

The YOLOv8 model contains out-of-the-box support for object detection, classification, and segmentation tasks, accessible through a Python package as well as a command line interface.

Compared to YOLOv8's predecessor, YOLOv5, YOLOv8 comes with:

1. A new anchor-free detection system.

2. Changes to the convolutional blocks used in the model.

3. Mosaic augmentation applied during training, turned off before the last 10 epochs.


Fig. 8 Yolo v8-s(small architecture) Preds on Test Data

## Model Results:

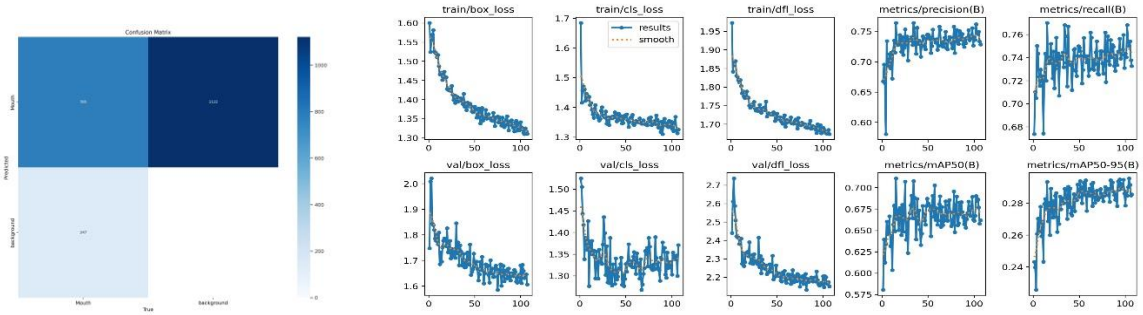



Fig. 9 Confusion Matrix and Metrics for Yolo v8-s best val scorer

Image Augmentations used:

1) Random Cropping: Randomly crops the image to the specified dimensions

2) Horizontal Flipping: Flips it horizontally.

3) Random Brightness Contrast: Randomly Increases the brightness contrast of image.

Model Problems:

1) Larger prediction ROIs due to variety of training data images close up distances.

2) Low confidence scores due to variety in images, with respect to angles, lighting etc.

Thus, we decide on integrating it with Mouth ROI Segmentation.

## Mouth ROI Segmentation

Some of the SOTA segmentation models which we are going to use, are as follows,

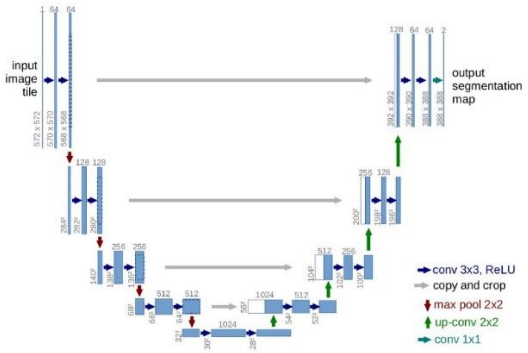
1) U-Net Model

2) SAM (Segment Anything Model)

3) Mask RCNN

## 1) U-Net

The U-Net architecture is widely used in biomedical image analysis for its effectiveness in image segmentation tasks. Its distinctive U-shaped design consists of a contracting path that captures context and reduces image dimensions, followed by an expansive path that enables precise localization using up-sampling and skip connections. These skip connections preserve spatial information, aiding in object localization. Downsampling and upsampling operations in the network allow for learning both global and local features. U-Net uses small receptive fields in convolutional layers for feature extraction, capturing high-level semantic features and reconstructing segmented images at the pixel level. The final layer typically employs a sigmoid activation function to produce a binary mask indicating object presence in each pixel. The network is trained using pixel-wise binary cross-entropy loss, comparing predicted masks with ground truth masks. Overall, U-Net excels in scenarios requiring accurate object segmentation and localization.




Fig, 10 Unet Architecture

U-Net inputs, images and it’s corresponding masks.

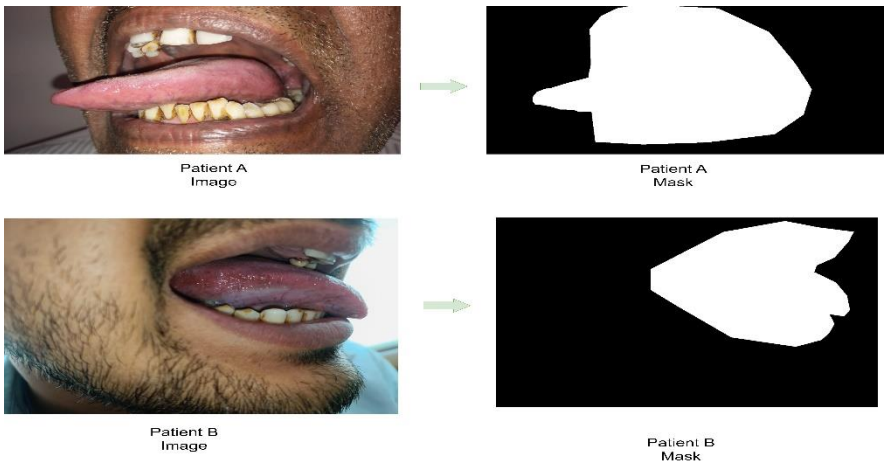


Fig. 11 U-Net Inputs

The masks were labelled and extracted from CVAT.ai, an amazing tool for Instance Segmentation.

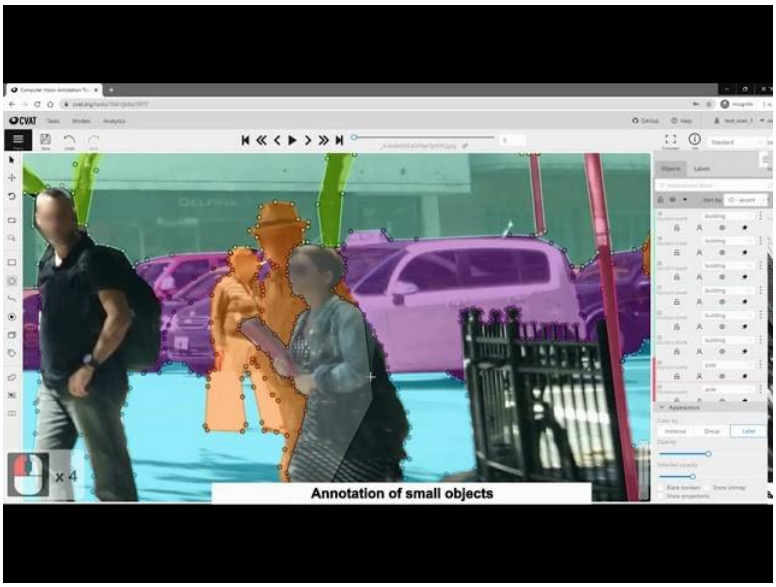



Fig. 12 CVAT Annotations

## 2) Segment Anything Model

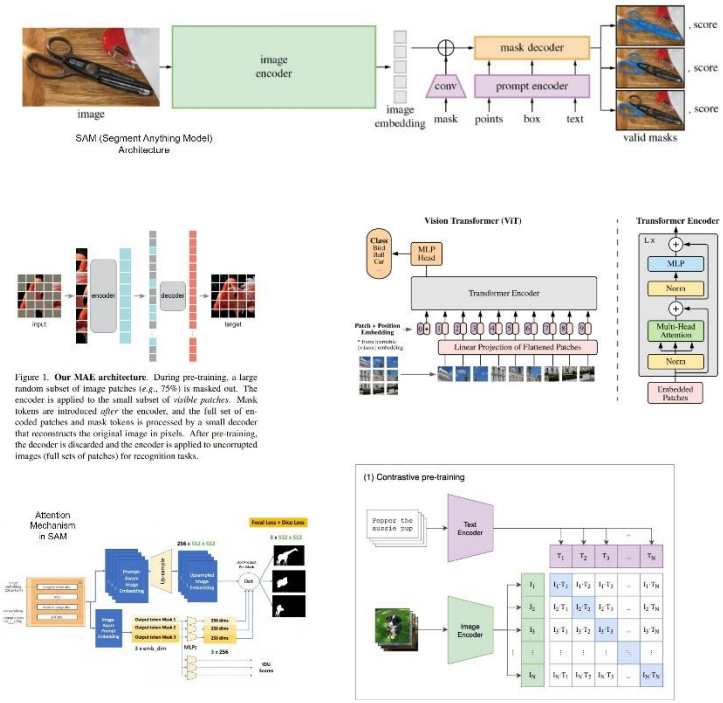



Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

Fig. 13 SAM Architecture

The Segment Anything Model (SAM) is a state-of-the-art segmentation model designed for promptable segmentation tasks, allowing it to generate accurate segmentation masks based on

prompts. It features an advanced architecture with an image encoder, prompt encoder, and mask decoder, enabling flexible prompting and real-time mask computation. SAM is trained on the SA-1B dataset, which contains over 1 billion masks on 11 million images, making it the largest segmentation dataset. SAM demonstrates impressive zero-shot performance, making it suitable for various segmentation tasks with minimal prompt engineering.

### 3) Mask R-CNN

Mask R-CNN is a deep learning model used for instance segmentation, which is a combination of object detection and semantic segmentation. It extends the Faster R-CNN model by adding a branch for predicting segmentation masks on each Region of Interest (RoI), in addition to the existing branch for classification and bounding box regression.

Mask R-CNN's key features include a Region Proposal Network (RPN) for efficient region proposals, ROI Align for accurate feature extraction from RoIs, and two-stage training for both classification and mask prediction. Overall, Mask R-CNN is widely used for tasks requiring detailed object segmentation, such as medical image analysis, autonomous driving, and video understanding.

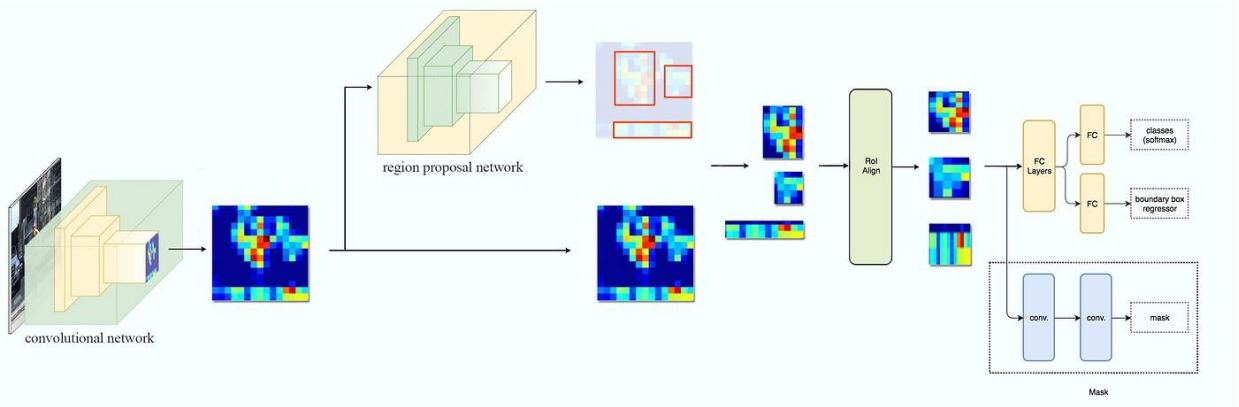



Fig. 14 Mask RCNN Architecture

## Lesion ROI Detection

Model used: Yolo-v8

Fig. 15 Validation (Batch of 16) Preds

Results of Predictions:

Problems encountered:

1. Inclusion of Non-Oral Cavity Areas:

- Issue: The segmentation model considers areas beyond the oral cavity.

- Implication: It may lead to inaccurate segmentation, affecting the precision of the model for the intended task.

2. Repeated Low Confidence Predictions in Close Proximity:

- Issue: Multiple low-confidence predictions in nearby regions.

- Implication: This can introduce uncertainty and reduce the model's reliability in distinguishing between different regions, impacting the accuracy of segmentation.

3. Blurry Photos:

- Issue: Image blurriness in the dataset.

- Implication: Blurred images can hinder the model's ability to accurately identify boundaries and details, leading to imprecise segmentation results

4. Undetected Multiple Lesions:

- Issue: Failure to detect multiple lesions.

- Implication: The model may miss critical information, potentially resulting in incomplete segmentation and an inadequate representation of the actual pathology.

Image augmentations are used for YOLO training to improve the model's generalization and robustness. By augmenting the training images with various transformations such as rotation, scaling, flipping, and changing brightness or contrast, the model learns to recognize objects under different conditions. This helps prevent overfitting and enables the model to perform better on unseen data, which is crucial for real-world applications. Additionally, augmentations can help increase the diversity of the training data, which is particularly beneficial when the original dataset is limited in size. For this reason, we perform the following augmentations.

Augmentations Used:

Blur(p=0.01, blur_limit=(3, 7)),

MedianBlur(p=0.01, blur_limit=(3, 7)),

ToGray(p=0.01),

CLAHE(p=0.01, clip_limit=(1, 4.0), tile_grid_size=(8, 8))

# Lesion ROI Classification

When it comes to image classification, Convolutional Neural Networks (CNNs) are the most commonly used models due to their ability to effectively learn spatial hierarchies of features. Here are some key reasons why CNNs are commonly used for image classification:

1. Local Connectivity: CNNs use convolutional layers that are locally connected, meaning each neuron is connected to only a small region of the input volume. This allows the network to focus on local patterns and features in the image.

2. Shared Weights: In CNNs, the weights of the convolutional filters are shared across the entire input image. This reduces the number of parameters in the network and helps in learning translational invariance, meaning the network can recognize patterns regardless of their position in the image.

3. Hierarchical Features: CNNs typically have multiple layers, with each layer learning increasingly complex features. Lower layers may learn simple features like edges and textures, while higher layers may learn more abstract features like shapes and objects.

4. Pooling Layers: CNNs often include pooling layers, which reduce the spatial dimensions of the feature maps. This helps in making the network more robust to variations in the position and size of the objects in the image.

5. Convolutional and Activation Layers: The convolutional layers in CNNs apply filters to the input image, extracting features that are relevant for classification. The activation layers (e.g., ReLU) introduce non-linearities, allowing the network to learn complex relationships in the data.

6. Transfer Learning: CNNs trained on large-scale datasets (e.g., ImageNet) can be fine-tuned for specific image classification tasks. This leverages the pre-trained network's ability to extract generic features, which can be useful when the target dataset is small.

7. Efficient Architecture: CNNs are designed to efficiently process images, with architectures that leverage the spatial structure of images. This makes them more computationally efficient compared to fully connected networks for image-related tasks.

8. State-of-the-Art Performance: CNNs have demonstrated state-of-the-art performance on various image classification benchmarks, making them a natural choice for many image-related tasks.

Hence we will be using CNNs and models mentioned henceforth will have CNNs as backbones for their network. There are multiple ways to tackle long tailed classification problems, some of the most common being stated below:

## 1) Under and Over sampling

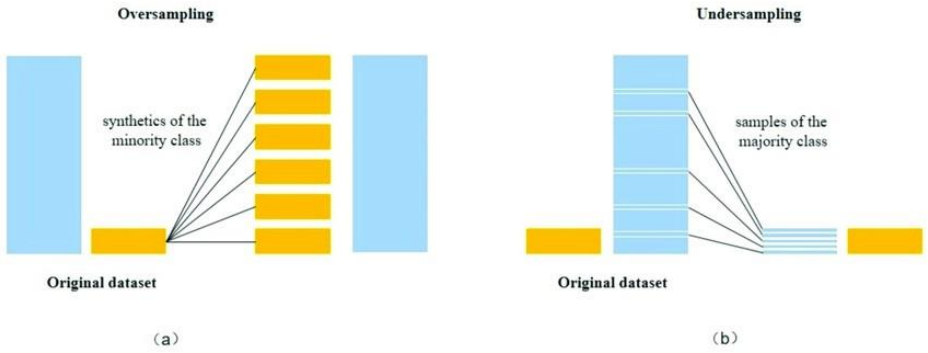



Fig. 16 Over and Under sampling of original dataset

Although these techniques work primarily on structured data, where data is organized in a tabular format, img pixel over/under sampling can be sometimes used as a natural data augmentation tool. Common methods for these are:

-Random Over/ Under Sampling

-Synthetic Minority Oversampling Technique (SMOTE) and Adaptive Synthetic Sampling (ADASYN)

Random oversampling causes generalization of data, and thus poor results on test set, and random undersampling can lead to loss of potential information from data. For this reason, we considered SMOTE and ADASYN, for smart data synthesis..

SMOTE basically works in the following steps:

It identifies the minority samples,  selects a random instance, find it's k nearest neighbours and then uses these instances to generate a new instance via interpolation. It keeps repeating the procedure till the desired balance between the majority and minority classes is achieved.

ADASYN is basically an extension of SMOTE, with modifications to tackle the problems of SMOTE like skewness of minority classes. It generates a synthetic sample from the imbalance ratio of the neighbours, where the imbalance ratio of a sample is the number of majority samples to the total instances in the neighoburhood of the sample. Thus, instances with higher imabalance ratio, are given more weightage for generation of the synthetic sample.

For under sampling, a standard random sampler from the imblearn library is used.

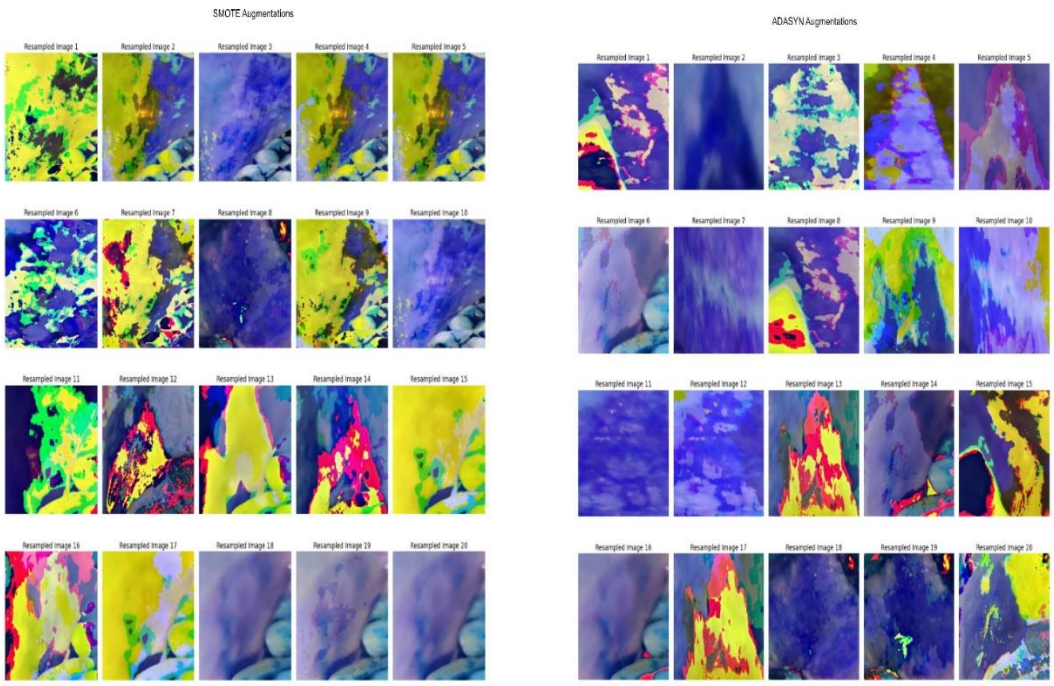



Fig. 17 SMOTE and ADASYN Results after minority oversampling strategy

After training and testing, it is seen that the image augmentations produced from the sampling techniques, are indeed random and cant be used. The specified augmnetations used after parameter tuning and testing, are :

Augmentations Used:

Blur(p=0.01, blur_limit=(3, 7)),

MedianBlur(p=0.01, blur_limit=(3, 7)),

ToGray(p=0.01),

CLAHE(p=0.01, clip_limit=(1, 4.0), tile_grid_size=(8, 8))

## 2) Class weights and Modified Losses

Another solution of tackling long tailed problems was using class weights and modified loss functions. Class weights are used for increasing the contribution of the minority classes and thus prevent bias of model to the majority classes. Modified loss functions, tailored to penalize the model stronger for the minority classes, also serve the same purpose.

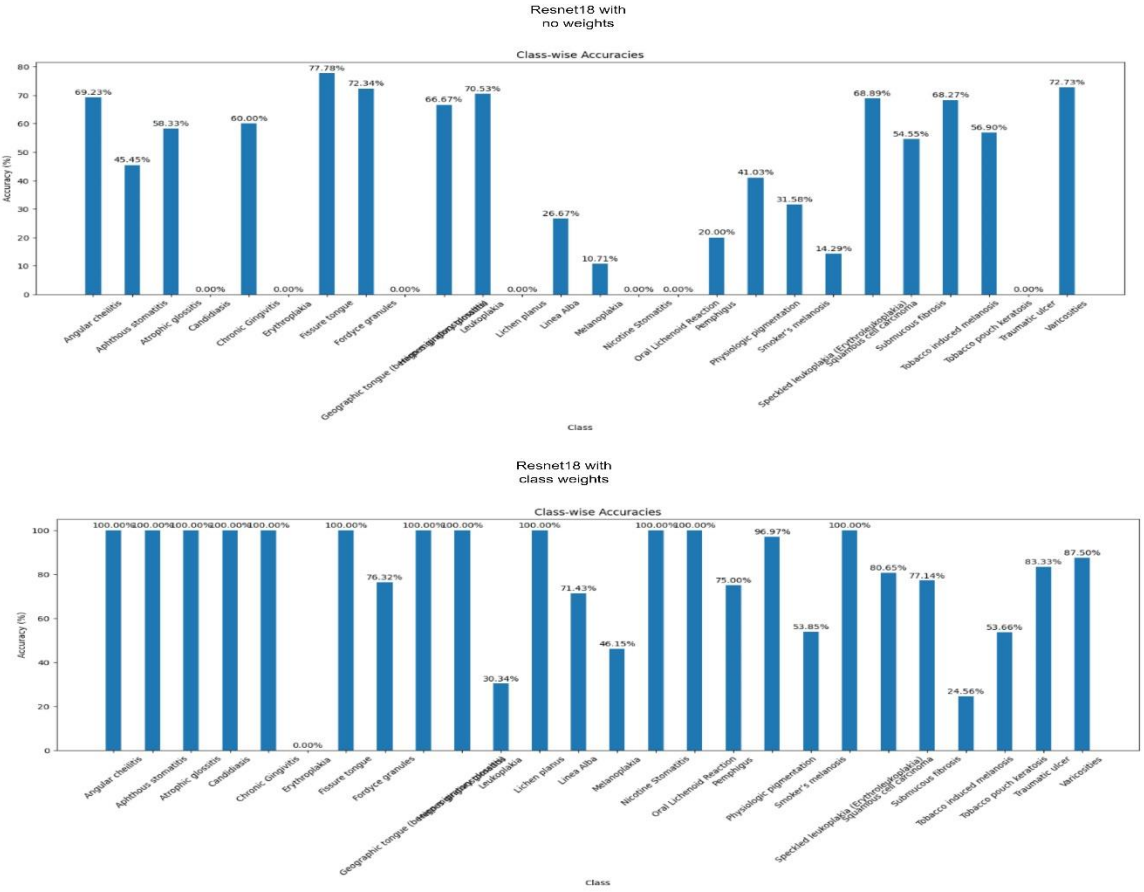



Fig. 18 Renet18 (Sample CNN Transfer Learning Model) with and without weights class-wise accuracies

Thus, class weights highly impact the minority classes' contribution to model training. Going by this, all models henceforth will be using class weights, inversely proportional to the number of image for that class.

Focal Loss

Using this loss, you can focus the model's attention on the hard examples (the minority classes) by down-weighting the loss assigned to well-classified examples (the majority classes).

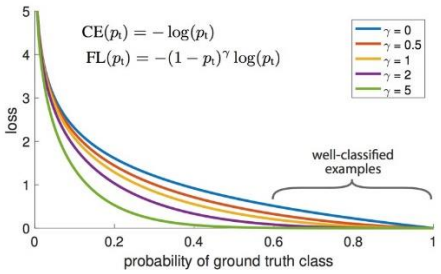



Fig. 19 Focal Loss, playing with gamma parameter

The γ parameter can be set to 2, as it is commonly done in practice. Adjusting this parameter allows you to effectively address the long-tailed nature of the dataset by making the model focus more on hard examples, (with $p_t$ close to 0.5), than the easy ones.

LDAM Loss

$$\mathcal{L}_{\text{LDAM}}((x,y);f) = -\log \frac{e^{z_y-\Delta_y}}{e^{z_y-\Delta_y}+\sum_{j\neq y} e^{z_j}}$$

$$\text{where } \Delta_j = \frac{C}{n_j^{1/4}} \text{ for } j \in \{1,\ldots,k\}$$

Fig. 20 LDAM Loss

LDAM aims to mitigate this problem by adjusting the margin in the softmax function based on the inverse of the square root of the class frequency. This adjustment effectively increases the penalty for misclassifying minority classes, making the model more sensitive to these classes during training. Using LDAM helps improve the model's performance on imbalanced datasets by giving more emphasis to minority classes during training.

These losses are integrated with standard CNN Models like Resnet, InceptionNet etc. Here we use Transfer Learning for the following reasons:

1) It works well for long tailed, where the feature learning is low for minority classes, thus we could reuse the features learned through using a pretrained model on a large dataset.

2) It will improve the generalization due to the model having seen multiple images, and also increase the model convergence to loss function.

The Models considered were Resnet18, Resnet50, VGG16, VGG19 and InceptionNet.

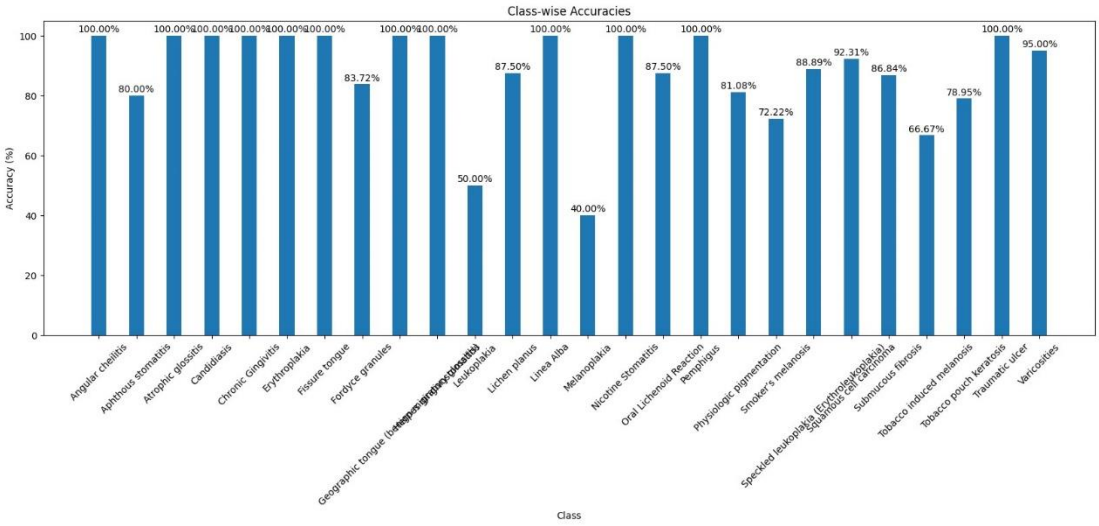



Fig. 21 Resnet50 + Class Weights + Specified Augmentations + Cross Entropy Loss Class Accuracies

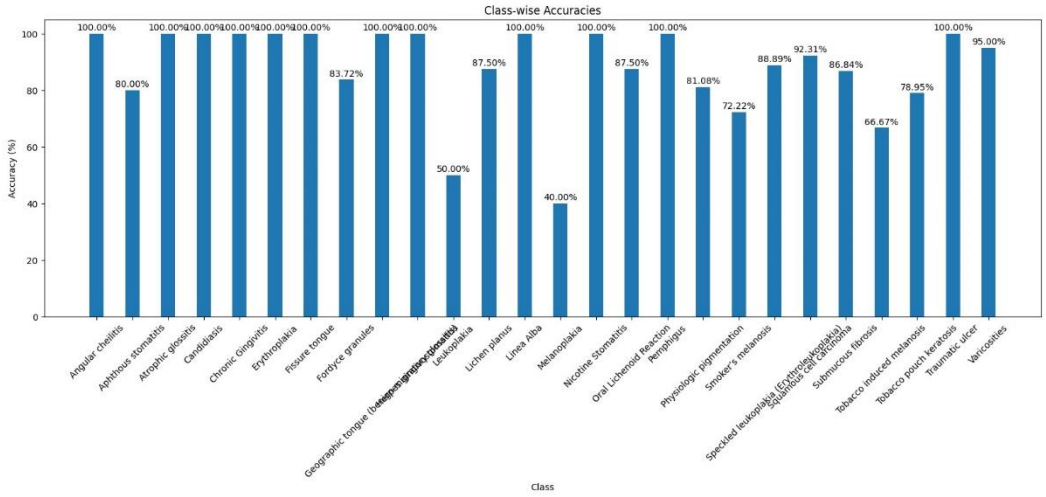



Fig. 22 Resnet50 + Class Weights + Specified Augmentations + Focal Loss Class Accuracies (*Best Transfer Learning Model)

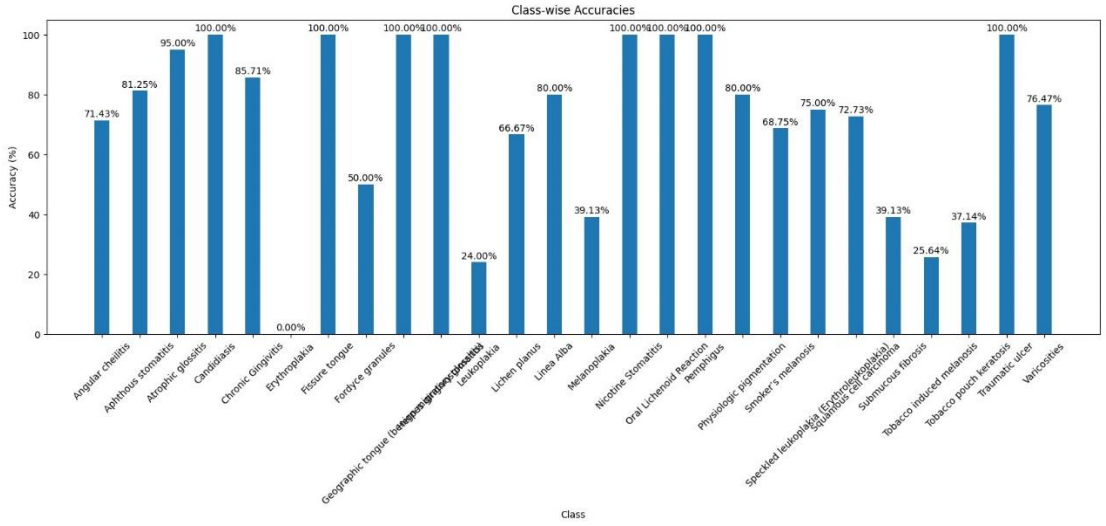



Fig. 23 VGG16 + Class Weights + Specified Augmentations + Cross Entropy Loss Class Accuracies

The step by step process, allowed us to first select the bes architecture for the dataset, which was Resent50, then use hyperparameter tuning to get the best augmentations, applying class weights, iterating through the SOTA long tail models, and finally selecting the best model.

## 3) Contrastive Learning

Contrastive learning has emerged as a powerful technique for learning representations in deep neural networks. However, its application to long-tailed problems, where the class distribution is heavily skewed towards a few classes, remains relatively unexplored. In contrastive learning, the model learns to pull together representations of similar examples while pushing apart representations of dissimilar examples. This property makes it well-suited for addressing the imbalance in long-tailed datasets, where the model can learn more

discriminative representations for minority classes by emphasizing their differences from the majority classes.

One key challenge in applying contrastive learning to long-tailed problems is the design of the contrastive pairs. Since there are fewer examples for minority classes, forming balanced pairs (i.e., pairs containing examples from both the majority and minority classes) can be challenging. One approach is to use hard negative mining, where the model focuses on examples that are difficult to distinguish, potentially including more examples from the minority classes. Another approach is to use class-aware sampling, which biases the sampling towards the minority classes to ensure that they are adequately represented in the contrastive pairs. Supervised contrastive learning, has been frequently used for the same.

In addition to the pair sampling strategy, the choice of the contrastive loss function is crucial.

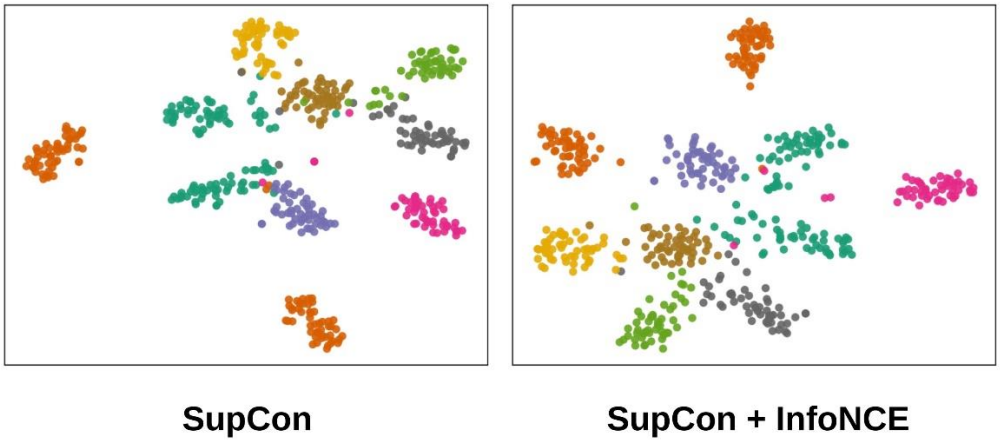



Fig. 24 InfoNCE significance

While the standard contrastive loss (e.g., InfoNCE loss) has been widely used, it may not be optimal for long-tailed problems. In this case, to specifically tackle the condition of long tailed problem, we suggest the following two models:

a. Supervised Contrastive Learning

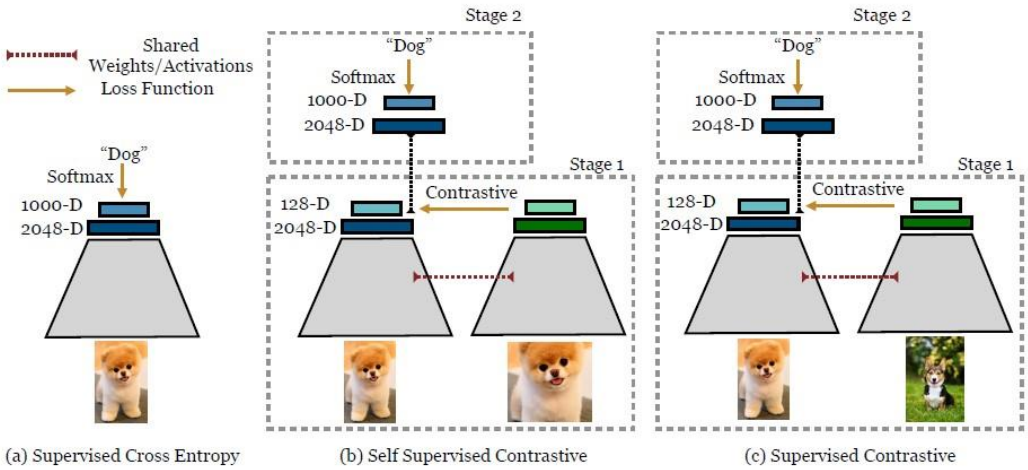



Fig. 25 Supervise Contrastive Algorithm

Supervised Contrastive Learning (SupCon) has emerged as a powerful technique for learning representations that are both discriminative and invariant to variations within the same class. The fundamental principle of SupCon is to maximize the similarity between augmented views of the same image (positive pairs) while minimizing the similarity between augmented views of different images (negative pairs). This approach encourages the model to learn embeddings that bring together semantically similar instances while pushing apart instances that are dissimilar.

Formally, given a batch of images, SupCon first generates two augmented views for each image, treating them as positive pairs. It then forms negative pairs by considering one view of an image as the anchor and selecting views from other images in the batch. The similarity between pairs is computed using a similarity function, often the cosine similarity, which measures the cosine of the angle between the embeddings of the views.

The contrastive loss, typically formulated as the InfoNCE loss, is then employed to encourage the model to learn embeddings that increase the similarity of positive pairs and decrease the similarity of negative pairs. The InfoNCE loss is a variant of Noise-Contrastive Estimation (NCE) adapted for neural networks, which has been shown to effectively learn representations in a self-supervised or semi-supervised setting.

During training, the model's parameters are updated using backpropagation to minimize the contrastive loss, resulting in embeddings that capture the underlying structure of the data. By learning such representations, SupCon enables the model to achieve state-of-the-art performance on various downstream tasks, including classification, clustering, and retrieval, even in scenarios with limited labeled data.

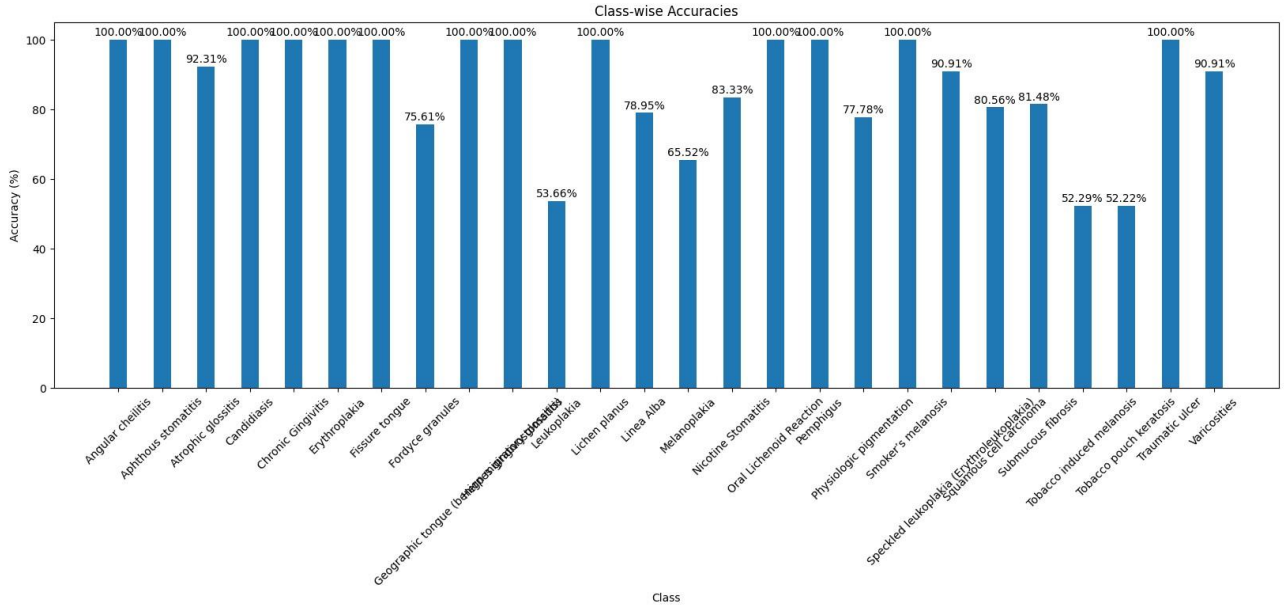



Fig. 26 Renset50 + Class Weights + Supcon Loss Class Accuracies

b. Parametrized Contrastive Learning

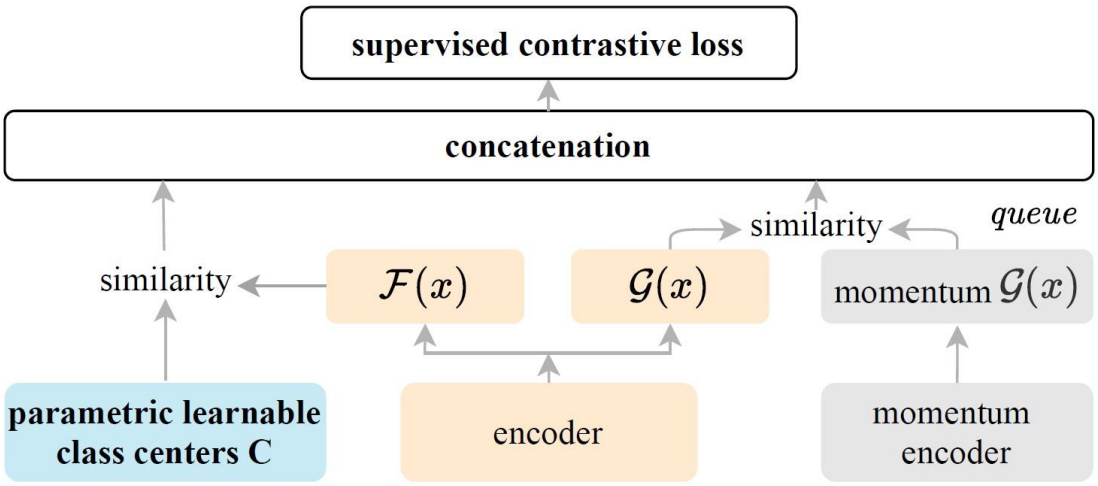



Fig. 27 Parametrized Supcon

Parametrized Contrastive Learning (PCL) represents a significant advancement in the field of representation learning, offering a flexible framework that adapts the notion of similarity to the specific requirements of a given task or dataset. Unlike traditional contrastive learning methods that rely on fixed similarity metrics such as cosine similarity, PCL introduces a parameterized similarity function that is learned alongside the main model. This parameterization enables the model to dynamically adjust the similarity function during training, leading to more task-specific and effective representations.

At the core of PCL is the parameterized similarity function, typically implemented as a neural network. This function takes as input the embeddings of two instances and produces a scalar value representing their similarity. During training, the parameters of this function are updated to maximize the similarity of positive pairs (instances that are semantically similar) and minimize the similarity of negative pairs (instances that are dissimilar). This optimization is achieved through a contrastive loss function, which encourages the model to learn embeddings that reflect the underlying structure of the data.

By allowing the similarity function to be parameterized, PCL can capture complex and nuanced relationships between instances, leading to more informative representations. This adaptability is particularly valuable in scenarios where the notion of similarity is context-dependent or varies across different parts of the data space. Experimental results have shown that PCL can outperform traditional contrastive learning methods in various tasks, highlighting its effectiveness and versatility in learning representations from data.

## 4) Diffusion Models

Diffusion models are gaining attention for addressing long-tailed problems due to their unique ability to generate realistic samples from complex data distributions. In the context of long-tailed datasets, where the majority of classes have limited samples, diffusion models offer several advantages:

1. Data Generation: Diffusion models can generate synthetic samples that closely resemble real data, especially in regions of the data space where training samples are sparse. This capability is particularly beneficial for minority classes in long-tailed datasets, as it helps in augmenting the dataset and improving the model's ability to generalize to these classes.

2. Data Augmentation: By generating new samples through diffusion, models can effectively augment the dataset, increasing the effective size of the minority classes. This augmentation can help in mitigating the class imbalance problem and improving the model's performance on minority classes.

3. Feature Learning: Diffusion models learn rich representations of the data distribution, capturing both local and global structure. This can be advantageous for long-tailed problems, where learning discriminative features for minority classes is challenging. The learned representations can enable the model to better distinguish between different classes, even with limited samples.

Here, we will be considering two models, namely:

a. DiffMIC: Dual-Guidance Diffusion Network for Medical Image Classification

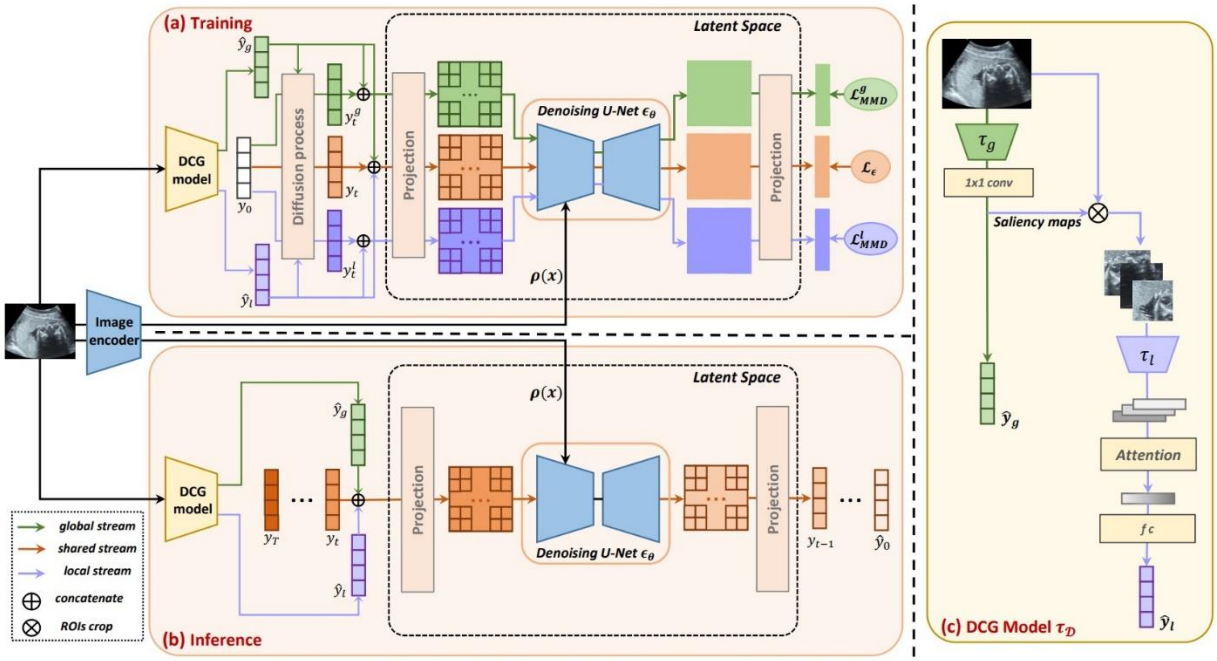



Fig. 28 DiffMIC Architecture

The key innovation of DiffMIC lies in its dual-guidance strategy, which leverages both contrastive learning and generative modeling to learn robust and discriminative representations from medical images.

At the core of DiffMIC is a dual-guidance diffusion network, which consists of two key components: a contrastive learning module and a generative modeling module. The contrastive learning module is responsible for learning representations that are discriminative and invariant to variations within the same class. It achieves this by maximizing the similarity between augmented views of the same image (positive pairs) and minimizing the similarity between augmented views of different images (negative pairs). This module helps in capturing the underlying structure of the data and improving the model's ability to generalize to unseen examples.

The generative modeling module, on the other hand, focuses on generating realistic samples from the learned representations. It utilizes a diffusion process, where noise is iteratively added to the representations to generate new samples. By training the generative model to reconstruct the original images from the noisy representations, DiffMIC encourages the model to learn representations that capture the essential features of the data distribution. The dual-guidance strategy of DiffMIC enables the model to learn representations that are both discriminative and semantically meaningful.

b. LDLMR: Latent-based Diffusion Model for Long-tailed Recognition

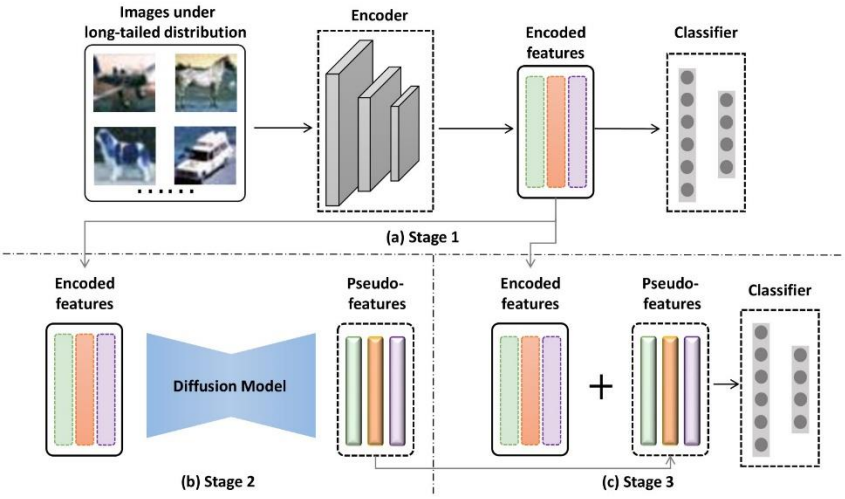



Fig. 29 LDLMR Architecture

The key innovation of this approach lies in its use of a latent space to model the data distribution, allowing for more effective representation learning and recognition of rare classes.

At the core of the Latent-based Diffusion Model is a two-stage training process. In the first stage, a diffusion model is trained to generate realistic samples from the data distribution. The diffusion model iteratively applies noise to a latent representation, gradually transforming it into a realistic sample. This process allows the model to learn a latent space that captures the underlying structure of the data, with a focus on preserving the characteristics of rare classes.

In the second stage, a recognition model is trained using the latent representations learned by the diffusion model. The recognition model takes a latent representation as input and predicts the corresponding class label. Importantly, the recognition model is trained using a novel contrastive loss that encourages the model to learn discriminative features for both the majority and minority classes. This helps to mitigate the effects of class imbalance and improve the model's performance on rare classes.

The Latent-based Diffusion Model offers several advantages over existing approaches. By learning a latent space that captures the data distribution, the model can generate synthetic samples that are more representative of rare classes. Additionally, the use of a contrastive loss

for training the recognition model helps to improve the model's ability to distinguish between different classes, even in the presence of imbalanced class distributions.

## 5) Heirarchical CNN structures

Heirarchical Clustering can be very useful, for long tailed problems with multiple classes, as the clustering groups similar classes together, thus increasing prediction accuracy, by narrowing the predictions to the sub-classes the model should predict.

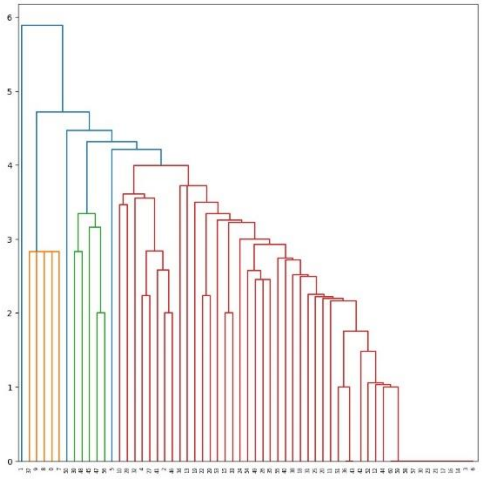


Fig. 30 Lesion Similarity Dendogram

A lesion similarity dendrogram is a hierarchical clustering visualization that shows the similarity between different lesions based on their features. It organizes lesions into clusters based on their similarity, with closely related lesions appearing closer in the dendrogram. This dendrogram helps in understanding the relationships between different types of lesions and can be useful in medical image analysis for grouping similar lesions and identifying patterns or correlations.

a. B-CNN: Branch Convolutional Neural Network for Hierarchical Classification

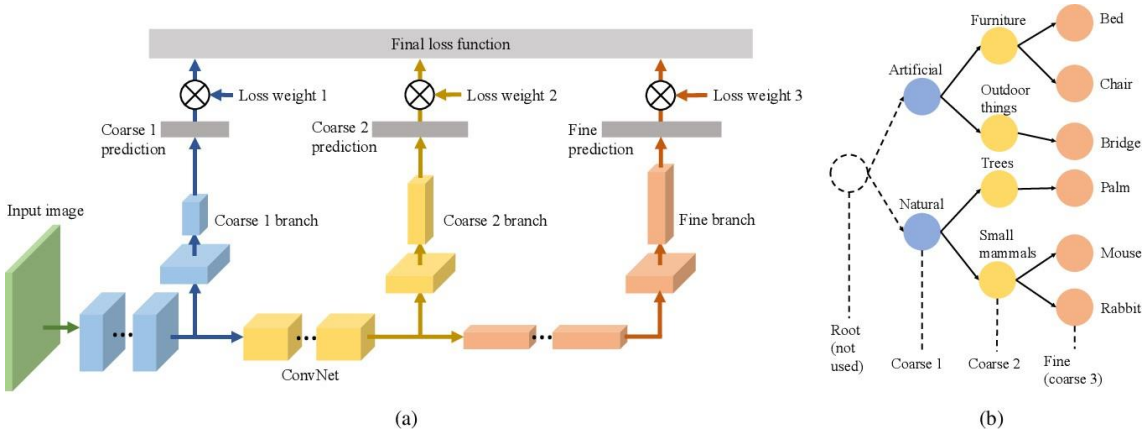



Fig. 31 B-CNN Architecture

The key innovation of this approach lies in its hierarchical branching structure, which allows the model to learn hierarchical representations of the input data at different levels of abstraction.

Branch CNN is a series of parallel branches, each responsible for learning representations at a specific level of the hierarchy. The branches are connected in a hierarchical manner, with lower branches capturing low-level features and higher branches capturing more abstract and hierarchical features. Each branch consists of a series of convolutional layers followed by pooling layers, which help in extracting hierarchical features from the input data.

The hierarchical branching structure of the Branch CNN enables the model to learn representations that are not only discriminative but also hierarchical in nature. This allows the model to effectively capture the hierarchical relationships between classes in the dataset, which is crucial for hierarchical classification tasks. During training, the model is trained using a multi-task learning approach, where each branch is trained to predict the class labels at its respective level of the hierarchy. The loss from each branch is then combined using a weighted sum to form the overall loss function, which is used to update the model parameters using backpropagation.

The Branch CNN offers several advantages over traditional CNN architectures for hierarchical classification. By explicitly modeling the hierarchical structure of the classes, the model can learn more meaningful representations and improve classification performance.

b. HMIC: Hierarchical Medical Image Classification

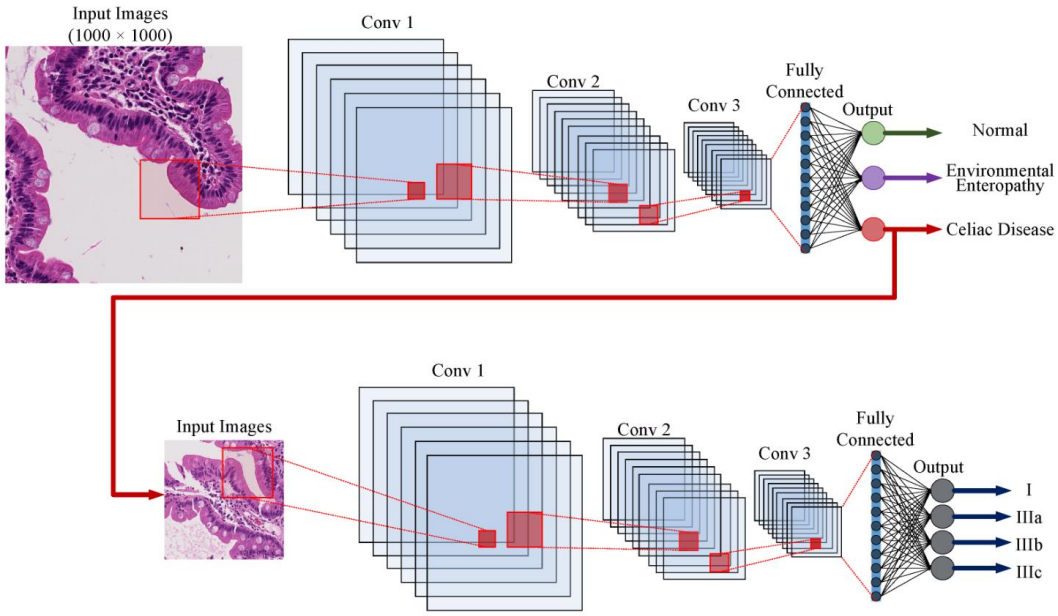



Fig. 32 HMIC Architecture

Hierarchical Medical Image Classification is a specialized task that involves categorizing medical images into a hierarchical structure of classes based on their content and characteristics. This ensures that the model doesn’t consider those classes that are not in the

coarse categories, thus making classification easier and more accurate. However, this will not work on very lowly represented classes well, thus it needs some tuning and good data augmentation for proper results.

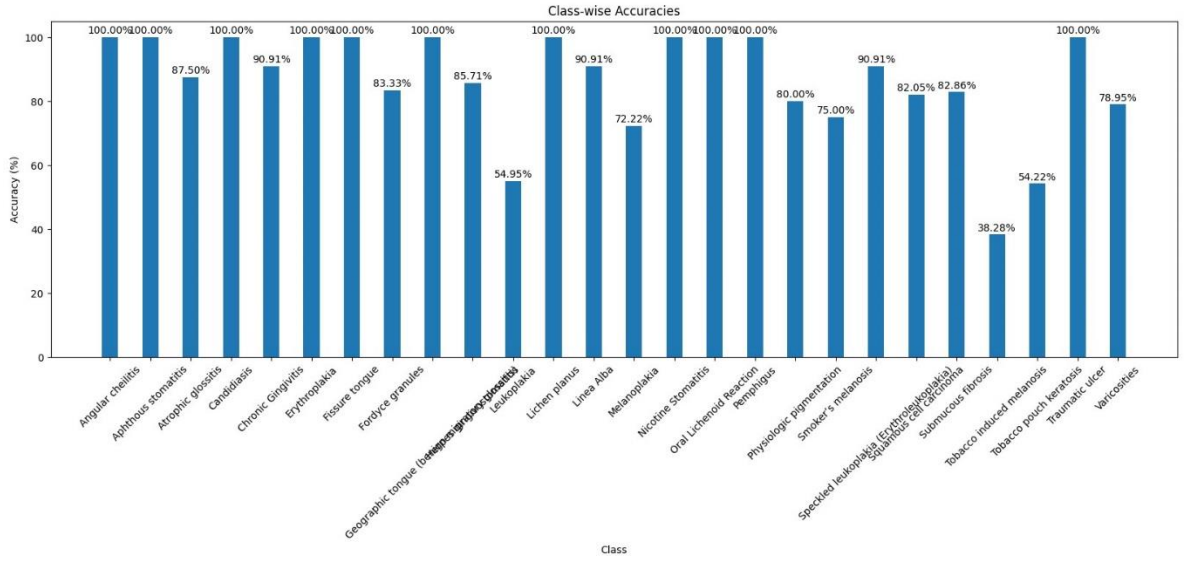



Fig. 33 HMIC with Resnet18 as CNN Model Class Accuracies

# Pipeline Results

| Segmentation Models | mAP for IOU Thresholding = | |
|---|---|---|
| | 50 % | 90 % |
| U-Net | 0.74 | 0.6 |
| SAM | 0.5 | 0.4 |
| Mask R-CNN | - | - |

| Detection Models | mAP for IOU Thresholding = | |
|---|---|---|
| | 50 % | 90 % |
| Yolo v8 for Mouth ROI | 0.76 | 0.4 |
| Yolo v8 for Lesion ROI | 0.4 | 0.2 |
| Fast R-CNN | - | - |

$$\text{IoU} = \frac{Area\ of\ overlap}{Area\ of\ union} = \frac{\text{Prediction} \cap \text{Ground truth}}{\text{Prediction} \cup \text{Ground truth}}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

| Classification Models | Model Accuracies (in %) |
|---|---|
| SMOTE / ADASYN Synthetic Minority Sampling + Random Majority Sampling | x |
| Resnet18 without Class Weights and No Augmentations | 56 |
| Resnet18 with Class Weights and No Augmentations | 60 |
| Resnet50 + Class Weights + Specified Augmentations + Focal Loss | 66 |
| Resnet50 + Class Weights + Specified Augmentations + Cross Entropy Loss | 72 |
| VGG16 + Class Weights + Specified Augmentations + Cross Entropy Loss | 52 |
| VGG19 + Class Weights + Specified Augmentations + Cross Entropy Loss | 51 |
| Resnet18 + Class Weights + Specified Augmentations + LDAM Loss | 49 |
| Resnet18 + Class Weights + Specified Augmentations + LDAM Loss | 52 |
| Resnet50 + Class Weights + Specified Augmentations + Supcon Loss | 76 |
| Resnet50 + Class Weights + Specified Augmentations + Parametrized Supcon Loss | - |
| DiffMIC (Contrastive + Generative Model) | 75 |
| LDLMR (Diffusion Based (VAE)) | 70 |
| Branch CNN (Dynamic) | 71 |
| HMIC + Resnet18 + Class Weights + Specified Augmentation + Cross Entropy Loss | 72 |

Fig. 34 Pipeline Results

Yolo v8, U-Net, and Resnet50 + Contrastive Learning with class weights and specified augmentations seem to be the best model so far.

# Future Scope:

1) Integrating Multimodal Embeddings

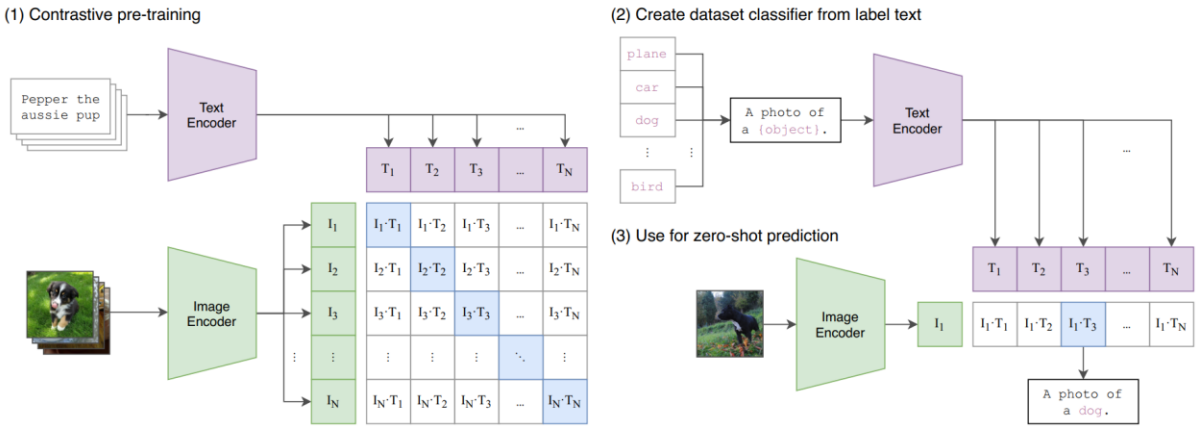



Fig. 35 Clip Architecture

- CLIP Model (Currently Giving 55% Accuracy)

2) Special Input Prompting for better Segmentation of Lesion

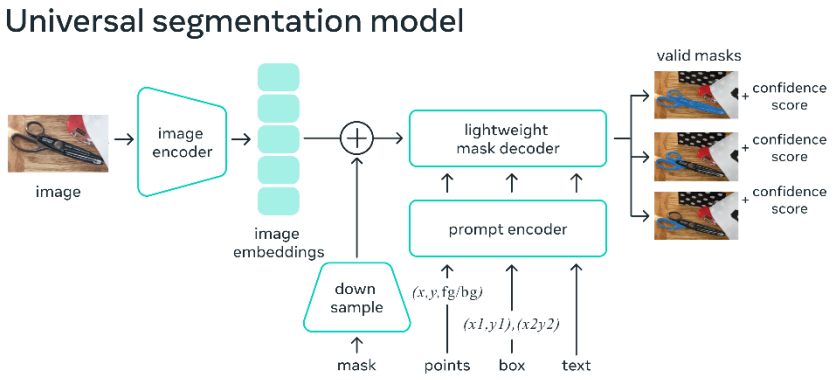



Fig. 36 Segment Anything Model (SAM)

- SAM Model with Bounding Box Input (Currently Giving about 0.5 IoU Thresholding)

Project Report by Avdhoot Lendhe

Supervisor: Dr. Kshitij Jadhav