Data Preprocessing

1. Extracted all the bounding boxes out of the provided csv file and formatted it into the JSON file structure followed by COCO2017
2. All data instances with no bounding boxes or invalid bounding boxes are removed (DINO only uses data that contains at least 1 bounding box)
3. Resize images
    1. Determine Scale Factor and Dimensions
        1. It checks whether the image is wider or taller by comparing width and height.
        2. If the image is wider, it scales based on the width (target_width), adjusting the height proportionally.
        3. If the image is taller, it scales based on the height (target_height), adjusting the width proportionally.
    2. Resize:
        1. After calculating the new dimensions (new_width and new_height), it resizes the image using the calculated scale factor, preserving the aspect ratio.
    3. Padding Calculation
        1. It computes padding values (padding_x and padding_y) to center the resized image within the target dimensions.
        2. If the image was scaled by width, padding is added on the vertical axis (padding_y).
        3. If scaled by height, padding is added on the horizontal axis (padding_x).
    4. Create Padded Image
        1. A new image with the target dimensions is created as a black canvas.
        2. The resized image is pasted onto this canvas at the calculated padding position, centering it.
    5. All bounding boxes are adjusted to be correctly positioned for the resized images

4. Area is calculated and added to the JSON files
5. Calculates the mean and standard deviation for each color channel separately (Red, Green, and Blue) . This is used for normalization for both the training/validation and test set.

Custom Data Augmentation Pipeline

Although DINO does include some data augmentation we did not find the augmented images to have sufficient diversity or mimic realistic possibility. Thus we derived our own data augmentation pipeline. Due to the large dataset size and storage restrictions we used online augmentation such that images were randomly augmented during training.

Because of the natural randomness found in blood samples such as size of objects, position, colours (due to various stainings used) we augmented our dataset such that realism was maintained but the model would be exposed to extreme cases for improved generalization.

Our pipeline works as follows:

Our pipeline use a slot approach with probabilities to apply each slot.
100% - Random Rotation of either 0 or 180 degrees (thus not chaning the aspect ratio)

Because blood samples are not structured and the region of interest can be anywhere and surrounded by anything in the sample, rotating allowed us to make the model impartial to position and orientation of the Trophozoite.

We implemented the following spatial augmentation slot with a 50% probability of occurring. Due to the fine detail of Trophozoites we did not want to change the shape and structure of them substantially. Therefore the augmentation chosen focused on positional aspects. If augmentations did change the shape and structure then they were made very minor. This included an equal probability to apply the following augmentations:

1. Random crop: Scale range 0.5 - 0.9
2. Elastic transformation: Alpha range 2 – 6 and sigma range 5 – 9
3. Affine transformation: Shear range -10 – 10 and translate percent 0.05
4. Horizontal flip
5. Vertical flip

The intensity augmentations are a bit more complex. We have two intensity augmentation slots, the first having a 40% chance to apply and the 2nd being a stacking slot that has a 20% chance to apply if the first intensity augmentation was made. Therefore to prevent clashing augmentations or over augmentation we categorized our augmentations such that a category could only be applied once to each image. Meaning the 1nd slot for intensity augmentation would pick from the pool of augmentations categories with out replacement such that the 2nd slot could not apply the same category again.

The categories are as follows:

1. Brightness
    - Standard brightness: Scale factor in range -0.2 – 02
    - Clahe: Clip limit 1 – 4 and tile grid size 8
2. Noise
    - Gaussen Noise: Variance range 20 - 100
3. Sharpness
    - Unsharp Mask
    - Standard Sharpen
4. Colour
    - Hue, saturation and value shift: Hue shift 30, saturation shift 40 and value shift 20
5. Blur
    - Gaussian Blur: limit 3 – 9

Additionally to introduce more extreme augmentations on a irregular basis (5%) we added the following:

1. Random shadow: shadow roi=(0, 0.5, 1, 1), num shadows lower=1, num shadows upper=2, shadow dimension=5
2. Random fog: fog coefficient lower=0.1, fog coefficient upper=0.3, alpha coefficient=0.08
3. Coarse Noise: block size=16, noise level=(0.8, 1.2)

These simulate issues such as poor lens quality, dust and debri.

These augmentation slots were able to stack allowing for a diverse set of augmented images.

Normalization

For all images we applied channel-wise normalization using the following values that were calculated on our training dataset:

Mean: [0.5814, 0.5154, 0.5531]
STD: [0.2783, 0.2569, 0.2610]

with the equation as follows
normalized_pixel = (pixel – mean) / std

Model

- Model Used: DINO (DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection)
- Pre-trained model used: 4 Scale DINO with SWIN-L, which was pretrained of 24 epochs on the COCO 2017 dataset.
- We chose to use the swin-L backbone as it showed superior performance on small fine-grained detailed objects.
- 1/15 of the parameters of larger models like SwinV2-G while still outperforming them.

DINO is an exceptional model choice for detecting malaria in blood samples due to its advanced object detection capabilities tailored for precision and efficiency. Unlike traditional detectors, DINO leverages a unique combination of Contrastive DeNoising and Mixed Query Selection, which allow it to rapidly learn and accurately isolate small objects—critical for pinpointing tiny malaria parasites. Its "Look Forward Twice" technique refines predictions layer-by-layer, ensuring that bounding boxes are both sharp and reliable. Furthermore, DINO outperforms other state-of-the-art models while being computationally efficient, using significantly fewer resources and training time. This blend of high precision, rapid training, and optimized resource use makes DINO not only a powerful but also a practical choice for sensitive, fine-grained medical imaging tasks like malaria detection.

Training

Due to resource constraints we used a 3 phase fine-tuning approach to make training as effective as possible.

Phase 1:
- Epochs: 5
- Learning rate: 0.0001
- Batch Size: 3
- Image resolution: 1000 x 750

Phase 2:
- Epochs: 5
- Learning rate: 0.0001
- Batch Size: 2
- Image resolution:1500 x 1125

Phase 3:

- Epochs 5:
- Learning rate: 0.00005
- Batch Size: 1
- Image resolution: 2000 x 1500

The goal behind this 3 phase training was to improve training times while still giving the model the opportunity to examine high resolution images to learn fine detailed features. Phase 1 allows the model to learn general features found in the lower resolution images. Phase 2 allows the model to work on learning slightly more finer details. And phase 3 allows the model to fine tune for very small features.

Post processing

On inference, non maximum suppression was used with a intersection over union threshold of 0.5 to remove multiple predictions for the same region of interest.

Setup

All code is hosted on git including, DINO connected to our custom data augmentation pipeline, a ipynb file for google colabs for both training and inference.