CS 6243 Project Report: Microglia Image Analysis

Brendan Winn (ID: SWX320)

Agenda

1. Project Summary
2. Description of Data Set
3. The Proposed Deep Learning Model
4. Results
5. Conclusion

Project Summary

In this project, we seek to provide an automated analysis of microscope images of microglia. This is a computer vision task, involving image segmentation and object categorization.

Due to corruption in the labels associated with the dataset, a smaller subset of the overall dataset has been hand labelled with bounding boxes, leaving out the categorization task.

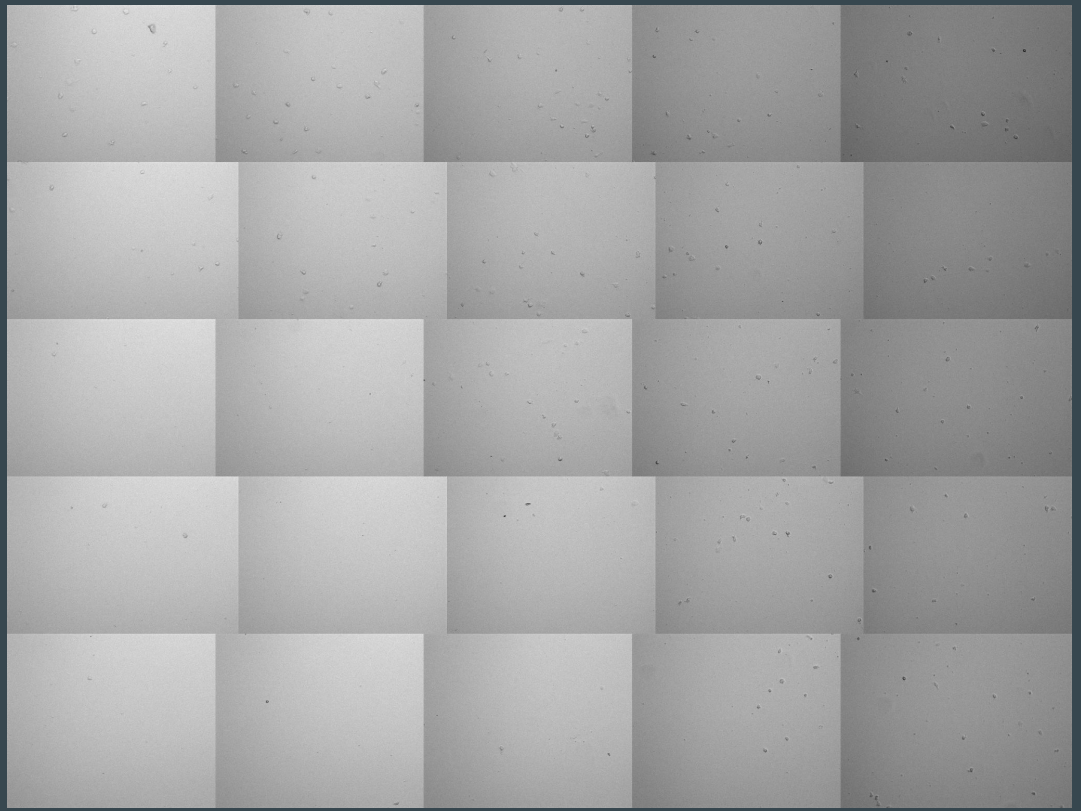
To do this, we utilize transfer learning on a pretrained model – PyTorch's *SSD300_VGG16*. Additional processing and labelling was attempted using OpenCV, but did not produce useful results.

Data Set

The original dataset consists of 133 TIF files of merged microscope images.

Preprocessing of this dataset involved three steps:

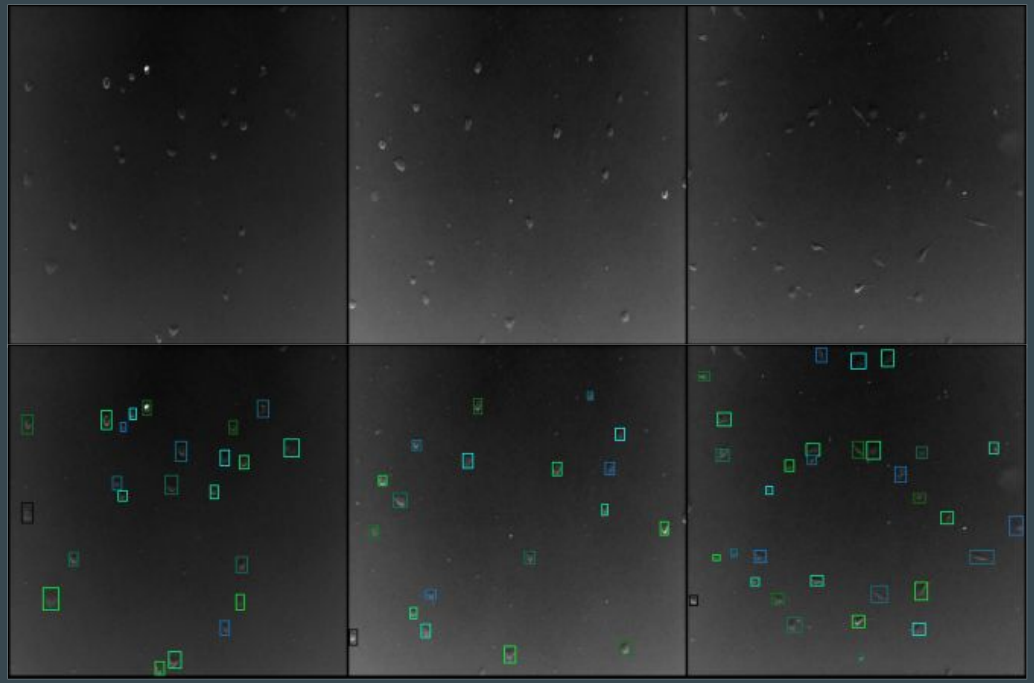
1. Split the merged images into individual images.
2. Denoise the images using bilateral filtering.
3. Normalize the images, and invert the color scheme for easier recognition.



Data Set (Cont.)

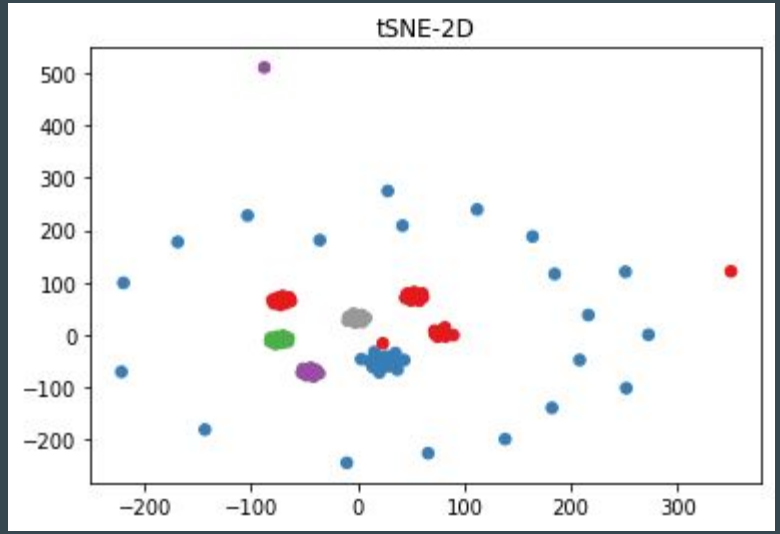
The provided labels for the images – a set of XML files with points for each cell – unfortunately were unusable, as even the program that produced the labels could not read them again.

To deal with the lack of labels, we hand labeled 140 images with bounding boxes, roughly denoting objects of interest in a given image.



Data Set (Cont.)

To analyze the images with t-SNE, we grouped images based on the number of bounding boxes they had, i.e. the number of objects in the image. Analyzing the graph shows that there do seem to be noticeable clusters forming, but a more complete analysis would require far more labelled samples.



Proposed Model

To see results despite the limited number of samples, we utilized transfer learning on *SSD300_VGG16*, one of PyTorch's pretrained segmentation models.

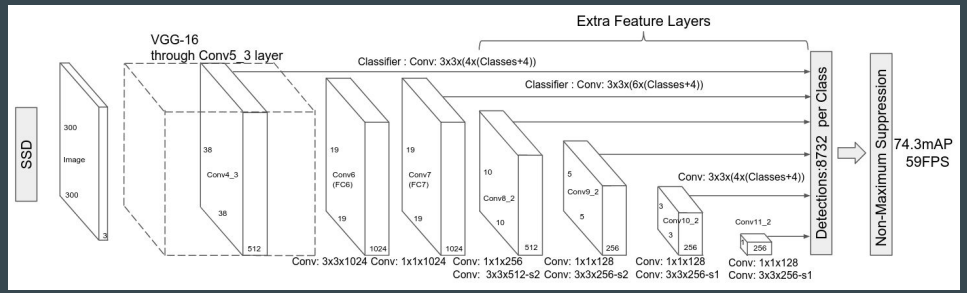
The model consists roughly of:

1. Base Convolution Layers providing lower level feature maps – 22 layers with 7,635,776 parameters, taking a 300x300 RGB image as input.
2. Auxiliary Convolution Layers providing higher level feature maps – 29 layers with 15,308,160 parameters.
3. Prediction Convolution Layers which locate and identify objects – 12 layers with 801,972 parameters between the heads, outputting a tensor of bounding boxes and a tensor of class labels.

After freezing several layers, only 341,084 parameters remained adjustable.

The loss function – Multibox loss – is a combination of regression loss for the predicted boxes and a classification loss for the class predicted for a given box.

For a more in depth explanation of the SSD model, see sgrvinod/a-PyTorch-Tutorial-to-Object-Detection.

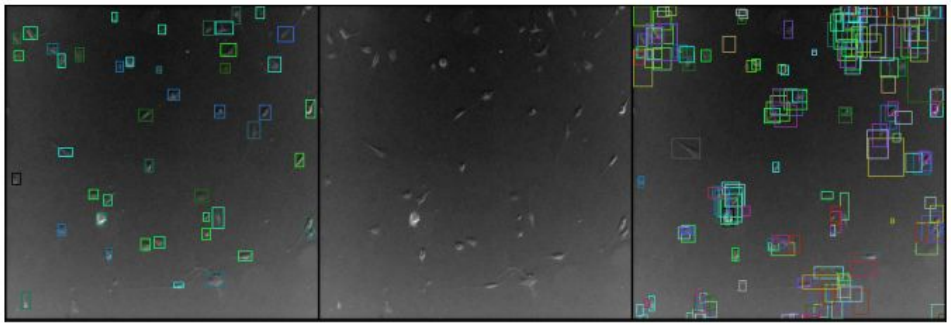


Results

Testing reveals an average loss of about 4.5 between different runs of the model.

Examining the predictions, we notice three issues:

1. The model's predictions overlap significantly, with many boxes completely overlapping
2. The model's predictions are much more sensitive to differences between the foreground and background
3. The edges of the image have a large number of boxes not seen in the ground truth labels.



LEFT: Image with ground truth bounding boxes. CENTER: Raw image.
RIGHT: Image with predicted bounding boxes.

Results (Cont.)

Using WandB, we tracked training and validation loss for one run. Additionally, for this run, we recorded all predictions for bounding boxes, which unfortunately were too visually busy to provide meaningful insight.

Conclusion

For this project, we've gone through eight steps to create, train, and test a model for creating bounding boxes on preprocessed images of microglia. To do so, we have performed numerous conversions and visualizations, utilized transfer learning via a pre-trained *SSD300_VGG16* model, reviewed the *Multibox* loss, trained with WandB logging, and used test images to evaluate the trained model.

While the results obtained are not quite usable, they do show promise as the model does seem to accurately include the objects of interest as in the ground truth. This is encouraging, especially considering the small amount of training data. A larger amount of high quality data would allow further training, including the possibility of retraining the encoder, which could make the model into a more usable tool.