Cancer Classification Challenge

Group 9

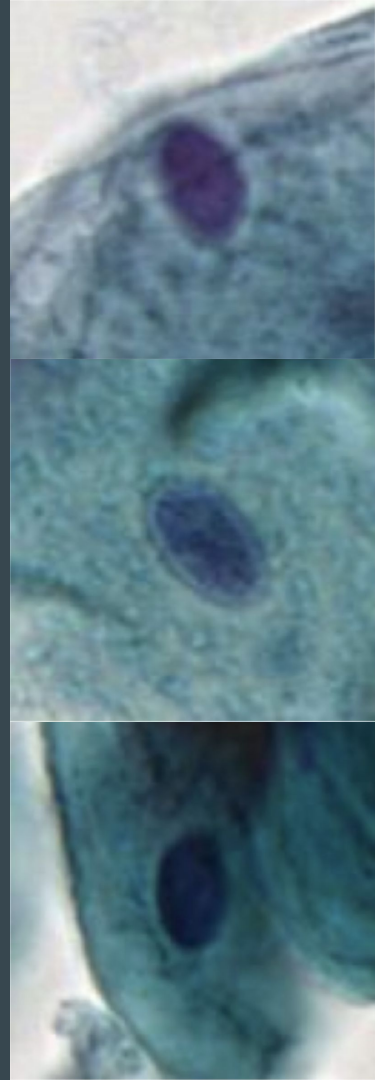
Jonas Dixon
Michail Bakalianos
Georgios Tsouderos
Mohit Uddin Ahmed

Introduction

- Oral cancer is on the rise all over the globe
- The main causes can be linked to alcohol intake, smoking as well as contracting HPV
- Early diagnosis is critical for survival rates
- Utilizing deep learning and neural networks, faster and more accurate detection can be achieved
- We will test and compare several neural network architectures to achieve the best possible classification performance on the given data

Dataset

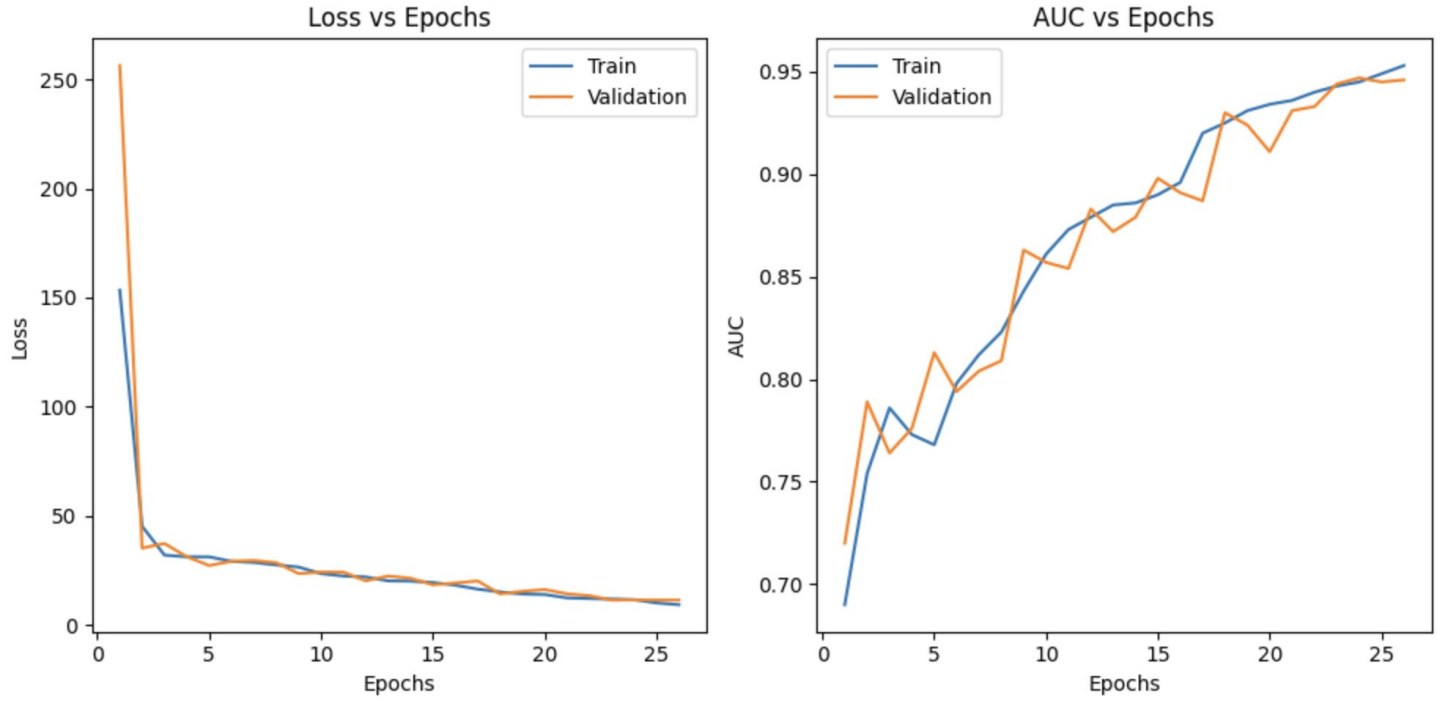
- Dataset is collected through brush scrapes of the mouth
- 19 patients (11 healthy and 8 with cancer)
- Each image contains an individual cell
- 73419 training images - 47684 labeled 0 (healthy) and 25735 labeled 1 (cancer)
- 66530 test images with unknown labels
- 128x128 RGB including CSV file with labels



Initial Attempt - Architecture

- 5 convolutional layers with max pooling
- 2 fully connected layers
- Batch normalization
- Dropout (25%)
- Kaiming He weight initialization
- ADAM optimizer
- ReLU activation function
- Batch size of 64
- 26 training epochs with early stopping conducted
- Learning rate scheduler with an initial value of 0.001 decreasing by a factor of 0.2 every 8 epochs

Initial Attempt - Results



Initial Attempt - Discussion

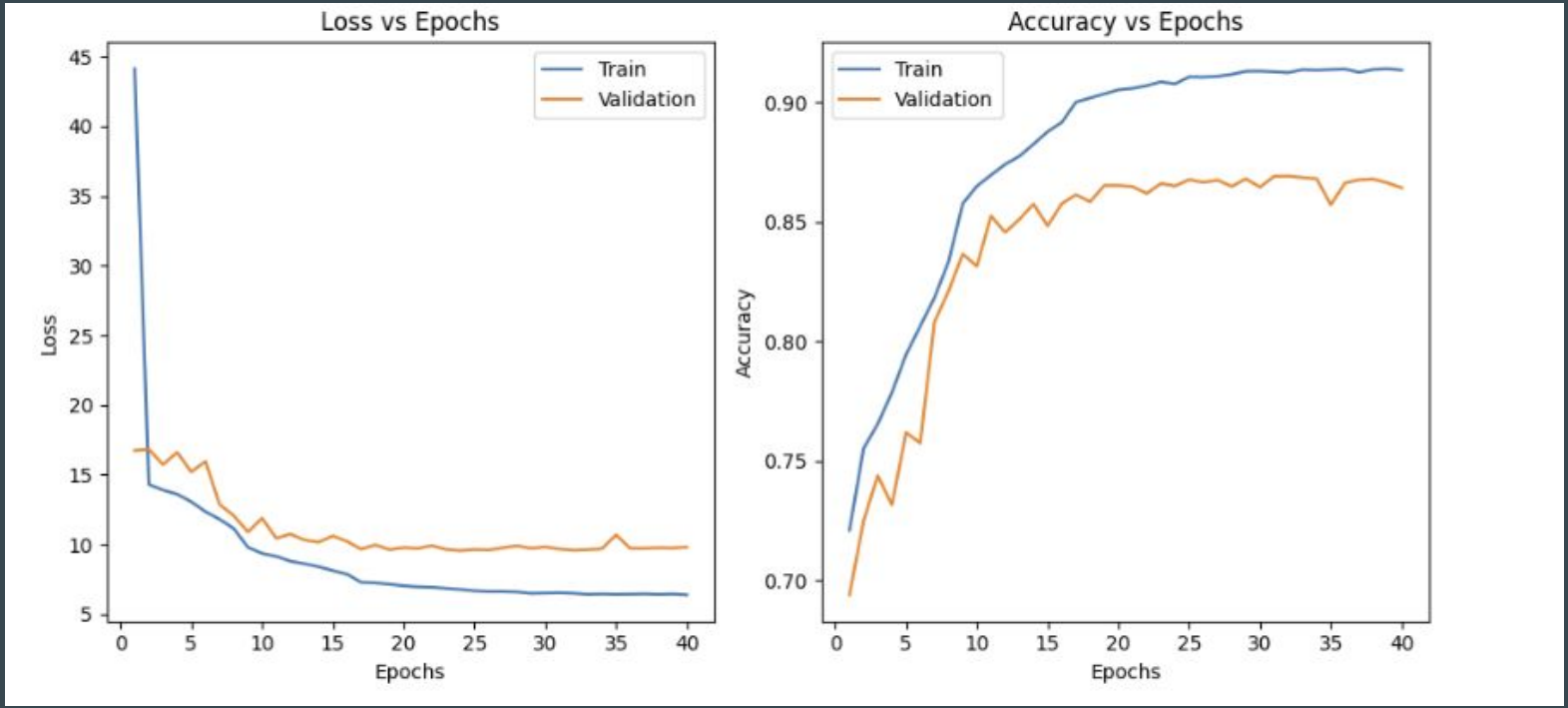
- 78% score on test-dataset
- Training and validation performance indicates underfitting
- Large discrepancy between test and validation performance however indicates overfitting
- Bad generalization can be due to data leakage with patients overlapping in between the train and validation sets
- Possible improvements include augmenting to balance the classes and increasing the model complexity

Attempt after the balancing of the two classes

- Calculate amount of data in each class
- Augment data of class with fewer examples until the difference is covered
- Augment by rotation, flip, brightness, contrast and saturation modifications

5 convolutional layers were used, along with batch normalization and two fully connected layers at the end. Dropout was increased to 0.5 since overfitting was observed

Results



- As we can observe, after around ~15 epochs, train and val accuracy start to differ
- AUC score = 0.94

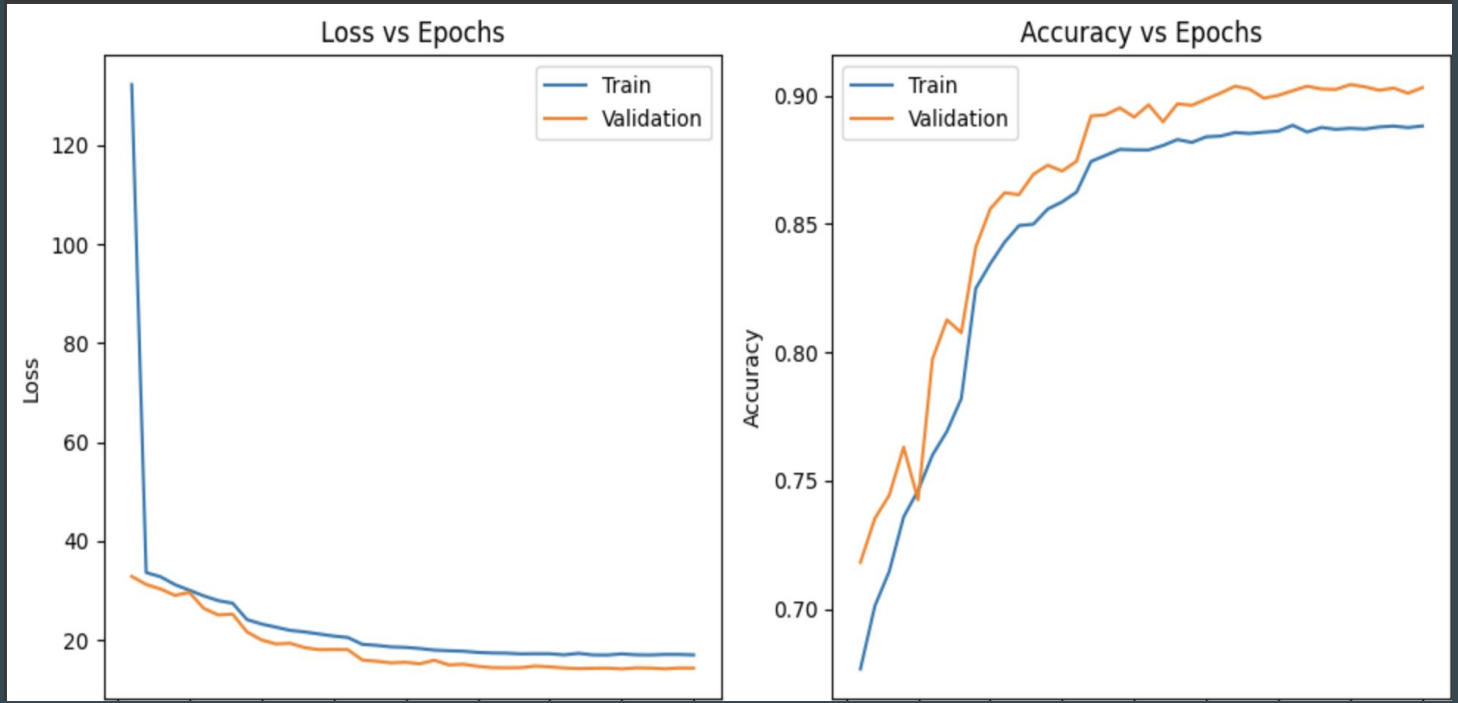
Second Attempt - Architecture (1)

- 5 convolutional layers with max pooling
- 2 fully connected layers
- Batch normalization
- Dropout (40%)
- Kaiming He weight initialization
- ADAM optimizer
- ReLU activation function
- Batch size of 64
- 40 training epochs with early stopping conducted
- Learning rate scheduler with an initial value of 0.001 decreasing by a factor of 0.2 every 8 epochs

Second Attempt - Architecture (2)

- We applied Data Augmentation in order to enhance the dataset.
- Randomly flipped the images horizontally and vertically to increase variability.
- Normalized the images to standardize input data and improve training stability.
- By applying data augmentation we got ~20k more images on the training set.

Second Attempt - Results



Second Attempt - Discussion

- Data Augmentation helped us achieve better accuracy, at least on training/validation dataset. We could say that we expect that this model could achieve better AUC score than the first attempt

```
Epoch [39/40], Train Loss: 17.1362, Train Accuracy: 0.8875, Val Loss: 14.4032, Val Accuracy: 0.9009
1.6000000000000004e-06
Epoch [40/40], Train Loss: 17.0412, Train Accuracy: 0.8881, Val Loss: 14.3806, Val Accuracy: 0.9031
3.200000000000001e-07
```

- Did not check score on test dataset due to GPU timeout issues.

- Validation accuracy is kinda better than training accuracy.

Third attempt - Architecture

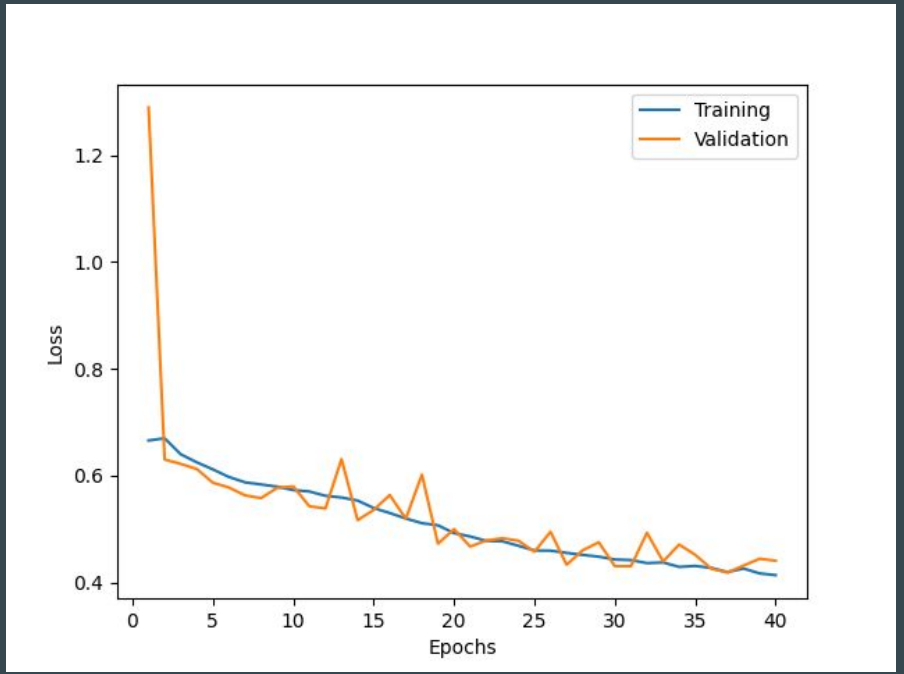
- Deeper network using residual block which consisted of 3 convolution layers each followed by ReLU activation
- Network starts with a convolution layer and ReLU followed by residual blocks
- 12 residual blocks
- Dropout for regularization
- ReLU for activation
- AdaptiveAvgPool3d and Flatten for final classification layer

Training

- Random rotations and flips to add randomness to the dataset during training data load
- 75% split for training and 25% for validation
- Adam optimizer with 0.005 learning rate and 32 batch size.
- 40 epochs

Results

- Prone to overfitting, trained for 100 epochs to get very low score on the test set
- Validation scores higher than test scores
- 84.5% score on the test set



Conclusion

- Deeper model increased score by 6%.
- Using residual layers made it easier to create a deeper network
- Use of transfer learning could prove useful
- Importance/heatmaps or other preprocessing techniques could help bring validation accuracy up for the smaller models
- Quite a difference between validation and test score