CSE5DL Assignment Report

This report chronicles answers to questions raised while completing the CSE5DL assignment.

# Task 1

## Task 1a

Data issues

**Question:** What did you check for? What data issues are present in this dataset?

**Answer:** I checked for the NAN values (not a number) and NA (missing values). There are no NAN and NA Values in both training and validation data set. Looking at the training/validation distribution calculated in Model training cell and illustrated through the Task 1a plot, we can see that in the training/validation dataset the distribution is skewed and around 70% labels are of class NV. So, the data set has class imbalance problem. This problem has been addressed in Model training cell using ‘WeightedRandomSampler’ sampler.

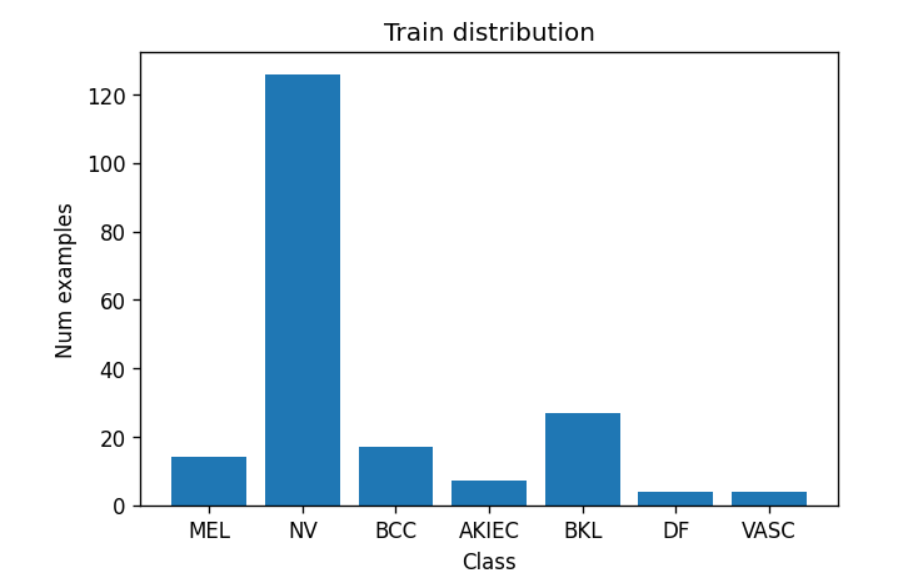
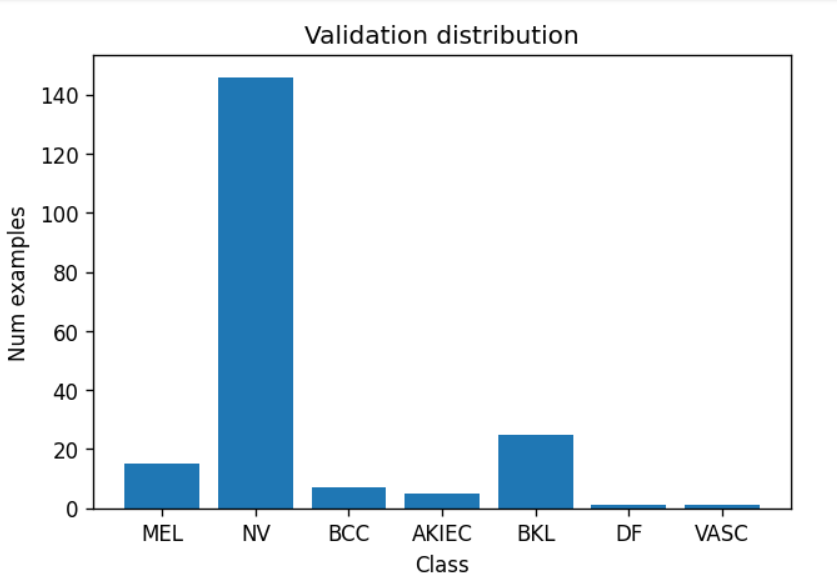
 

Figure 1: Task 1a plot for train distribution and validation distribution

*Training data distribution output in Model Training cell:*

Train data - ML proportion: 0.089 NV proportion: 0.683 BCC proportion: 0.050 AKIEC proportion: 0.038 BKL proportion: 0.113 DF proportion: 0.012 VASC proportion: 0.015

## Task 1b

### **Question:** Why not use random\_split?

**Answer:** We should not use randm\_split in case of an imbalance class data set. It would not be able to do a stratified split based on output labels in case of imbalanced data. Hence, we have used ‘WeightedRandomSampler’ for class to get balanced batches. More importantly, we have already split the data, so using random split would change validation set.

## Task 1c

### Reduce epoch time

**Question:** As training sets get larger, the length of time per epoch also gets larger. Some datasets take over an hour per epoch. This makes it impractical to debug typos in your code since it can take hours after starting for the program to reach new code. Name two ways to significantly reduce how long each epoch takes - for debugging purposes - while still using real data and training a network.

**Answer:** We can try debugging on a small network first and then define a deeper network. Also, we can take a small subset of training and validation dataset for debugging purposes. In this way epoch run time will be less. We can try train for a single (or a few) epochs before progressing to many epochs.

### Confusion

**Question:** Show the confusion matrix and plots of the validation accuracy and UAR in your report and explain what is going wrong. (Right-click a plot and select "save image as..." to save the image to your computer).

**Answer:**

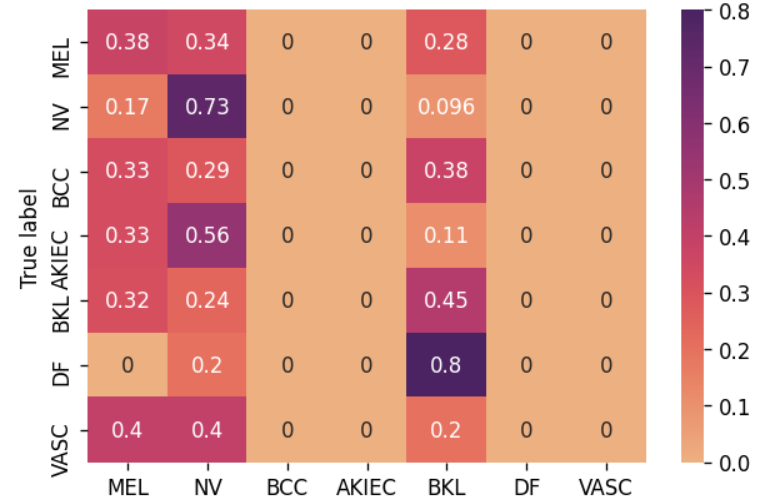


Figure 2: Confusion matrix before fixing class imbalance.

### Run summary:

|  |  |
| --- | --- |
| Loss/train | 0.00054 |
| Accuracy/train | 0.99967 |
| UAR/train | 0.99982 |
| Loss/val | 4.0644 |
| Accuracy/val | 0.5875 |
| UAR/val | 0.22221 |
| \_step | 4 |
| \_runtime | 4363 |
| \_timestamp | 1622309245 |

### Run history:

|  |  |
| --- | --- |
| Loss/train | █▁▂▃▁ |
| Accuracy/train | ▁██▇█ |
| UAR/train | ▁████ |
| Loss/val | ▁▇▃█▂ |
| Accuracy/val | █▆▆▁▆ |
| UAR/val | ▁█▆▅▇ |
| \_step | ▁▃▅▆█ |
| \_runtime | ▁▃▅▆█ |
| \_timestamp | ▁▃▅▆█ |

We can see in the confusion matrix that the model is not predicting any labels for BCC, AKIEC, and VASC classes. This is due to model not trained for these classes as they have less proportion in training data set. We call this class imbalance problem.

## Task 1d

### Account for data issues

### **Question:** How did you account for the data issues? Was it effective? How can you tell? Show another confusion matrix.

### **Answer:** For the class imbalance problem in dataset, I used the ‘WeightedRandomSampler’ as sampler in the ‘DataLoader’ to balance the samples used from each class in the model. Yes, it is effective in training the model in better way as model do better prediction for labels which have less examples in training phase. We can tell that by looking at the validation accuracy and UAR with are higher after applying the sampler. Also, we can see that the confusion matrix shows that correlation between the true labels and predicted labels is higher. That means model is doing well in predicting correct labels for an image.

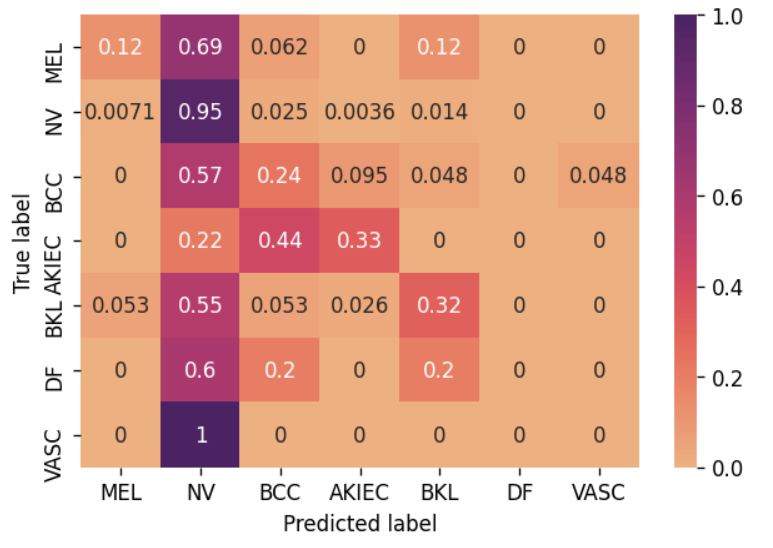


Figure 3: Confusion matrix after fixing class imbalance.

### Run summary:

|  |  |
| --- | --- |
| Loss/train | 0.74569 |
| Accuracy/train | 0.7267 |
| UAR/train | 0.31484 |
| Loss/val | 0.75615 |
| Accuracy/val | 0.735 |
| UAR/val | 0.28034 |
| \_step | 4 |
| \_runtime | 4266 |
| \_timestamp | 1622313914 |

### Run history:

|  |  |
| --- | --- |
| Loss/train | █▄▂▁▁ |
| Accuracy/train | ▁▅▆█▇ |
| UAR/train | ▁▃▇██ |
| Loss/val | █▆▂▃▁ |
| Accuracy/val | ▁▇▇▄█ |
| UAR/val | ▄▁█▄▆ |
| \_step | ▁▃▅▆█ |
| \_runtime | ▁▃▄▆█ |
| \_timestamp | ▁▃▄▆█ |

## Task 1e

### Vertical Flips

**Question**: Are random vertical flips appropriate for this dataset? Why?

**Answer:** Yes, random vertical flips are appropriate in this case because we are using images of skin disease which can appear in any shape or can be in any orientation while taking a picture. So vertical flips can work for this dataset as model will learn upside-down images of skin disease and this will improve the model's performance.

### Effect of Augmentation

**Question**: What effect did Data Augmentation have on performance? Show a screenshot of the relevant graphs from Weights & Biases for evidence.

**Answer:**

With data augmentation the performance of the resnet18 model degraded. I have recorded the parameters as:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0.00049 | 1 | 1 | 2.6965 | 0.545 | 0.2397 |
| Loss/train | Accuracy/train | UAR/train | Loss/val | Accuracy/val | UAR/val |

### [Challenge] 5 crop augmentation

## Task 1f

### Experiments

* <See attached excel document>
* <See attached Weights and Bias Report>
* Write a discussion about the key findings from the experimental results.

Discussion on results:

From the results we can say that small bath size with a small learning rate (0.0001) gives better accuracy for all models. Although I have used 5 epochs due to computational issues, I also observed the accuracy for 10 epochs, and it performed better. Big batch size results in overfitting. Augmentation is also helping model learn as training data is bigger.

I got the best results when I used Shuffle = True and did not used sampler. This gave a 0.79 accuracy and 0.32 UAR which is quite good for validation.

### [Challenge] Batch size

**Question**: Assuming you use the full dataset in a single epoch, if you halve the size of the batch size, what happens to the number of times that you update the weights per epoch? With reference to the gradients, under what circumstances is this good?

**Answer:**

With the full dataset in a single epoch if we half the size of the batch size the computation will reduce, because of not exploiting vectorization to the full extent. However, number of times that you update the weights per epoch will increase. This will also increase the accuracy and in same amount of time. It is better to use small batch size (generally 32) with small gradients to get better accuracy.

# Task 2

## Task 2a

### Data issues

**Question**: What did you check for? What data issues exist, if any? Report anything you checked even if it turned out the data did not have that issue. We want to know what you are checking.

**Answer:**

I checked for NAN values (not a number) and NA (missing values) in the dataset and found that there were no NAN and missing values in train and validation dataset. I also checked for class imbalance problem in the plots. There are no issues in the datasets.

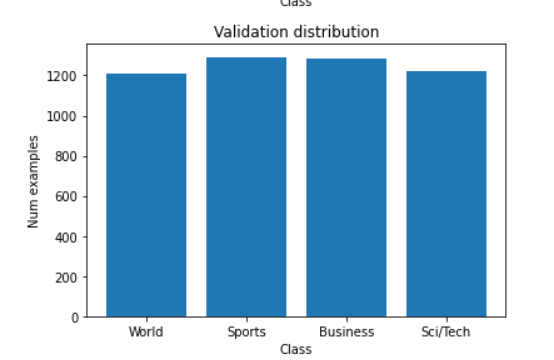
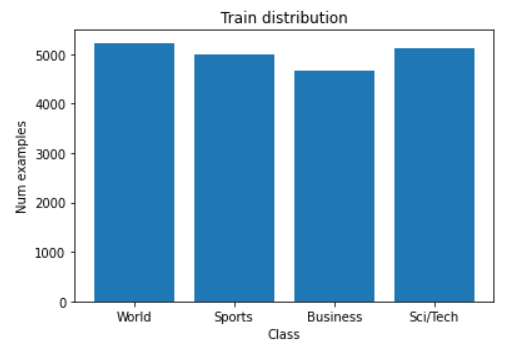
 

Figure 4: Task 2a plot for train distribution and validation distribution.

## Task 2b

### Similar embeddings

…

## Task 2c

### Saved model weights

...

### What do the longs (int64) represent?

….

[CHALLENGE] Visualization of fine-tuned DistillBERT model.

## Task 2d

Class distributions and learning rate

...

Relative performances before and after fixing learning rate

...