CSE5DL Assignment Report

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

**Author:** Matthew Finster

**Student ID:** 21702717

Table of Contents

[Task 1 1](#_Toc168133119)

[Task 1a 1](#_Toc168133120)

[Data issues 1](#_Toc168133121)

[Task 1b 2](#_Toc168133122)

[Why not use random\_split? 2](#_Toc168133123)

[Task 1c 2](#_Toc168133124)

[Reduce epoch time 2](#_Toc168133125)

[Confusion matrix 3](#_Toc168133126)

[Task 1d 5](#_Toc168133127)

[Account for data issues 5](#_Toc168133128)

[Task 1e 6](#_Toc168133129)

[Vertical Flips 6](#_Toc168133130)

[Effect of Augmentation 6](#_Toc168133131)

[[Challenge] 5 crop augmentation 8](#_Toc168133132)

[Task 1f 9](#_Toc168133133)

[Experiments 9](#_Toc168133134)

[[Challenge] Batch size 10](#_Toc168133135)

[References 10](#_Toc168133136)

# Task 1

## Task 1a

### Data issues

Upon exploring the training set, data issues were checked for. In particular, the data was checked for any null values. All data in both the training and validation sets were complete. The training set had 3004 non-null values while the validation set had 400 non-null values.

After viewing a summary of the data, a small sample of images were identified to check that the images were labelled correctly. Although this was difficult to assess with an untrained eye, it does appear that the sample of images selected were labelled correctly, providing confidence to continue. Further confidence is attained after reading that [“disease diagnoses were histopathologically confirmed”](https://challenge.isic-archive.com/landing/2018/47/) (with medical imaging and consensus of three expert dermatologists) in relation to this dataset.

Lastly, the distribution of classes was checked. The following information was acquired:

|  |  |  |
| --- | --- | --- |
| Class | Proportion of training dataset | Proportion of validation dataset |
| Melanoma (MEL) | 8.89% | 8.00% |
| Melanocytic nevus (NV) | 68.31% | 70.25% |
| Basal cell carcinoma (BCC) | 5.03% | 5.25% |
| Actinic keratosis (AKIEC) | 3.79% | 4.50% |
| Benign keratosis (BKL) | 11.28% | 9.50% |
| Dermatofibroma (DF) | 1.20% | 1.25% |
| Vascular lesion (VASC) | 1.50% | 1.25% |

There is a clear issue in the distribution of this dataset. It is heavily unbalanced with 68.31% of the training data belonging to the NV class. Theoretically, this could mean that a classifier could predict the NV class on all images and record an accuracy of 70.25% on the validation set. Obviously, this is an issue and requires further techniques to balance the data. Going forward, it is pertinent to consider data augmentation to generate additional samples for the other classes, or to consider assigning higher weights to samples from the minority classes to penalise misclassifications more heavily.

## Task 1b

### Why not use random\_split?

Random\_split splits the dataset into training and validation partitions randomly. As identified in Task 1a, the classes are imbalanced with 68.31% of the images in the training dataset belonging to the melanocytic nevus (NV) class. This is an issue. For example, if we randomly partition the dataset into a 68/32 training/test split using random\_split(), it is possible that we end up with a training set that is composed only of NV images and a test set that is composed of the other classes!

It is ideal that the training and validation datasets have already been partitioned in a proportional manner.

To further ensure fair representation of all classes during training, we will also need to consider how the batches are constructed. For example, we may need to oversample classes with fewer samples in the batches to result in a balanced representation and reduce bias. This will be addressed in Task 1d.

## Task 1c

### Reduce epoch time

To reduce epoch time, one can reduce the sample size as has been done in this project. For instance, the full dataset includes over 3000 images, but for debugging purposes, smaller subsets of 200 images were used. These smaller subsets are still representative of the class distribution found in the set of 3000 images.

Another way to reduce epoch time is to use a larger initial learning rate (for example, 0.01 as opposed to 0.001) which will cause the model to attempt to minimise loss at a larger quantity. This approach can, however, lead to error and/or a lack of proper convergence if the learning rate is too large.

Additionally, I would theorise that reducing image size from the 450x600 used in this program to a size used in other image processing tasks such as 224x224 could be beneficial for reducing epoch times as there will be less pixels to process. This can be done through resizing or cropping.

One could also save the weights and reload them to save time in retraining after the model has been run once, however this option is only useful after the model has been finely tuned.

### Confusion matrix

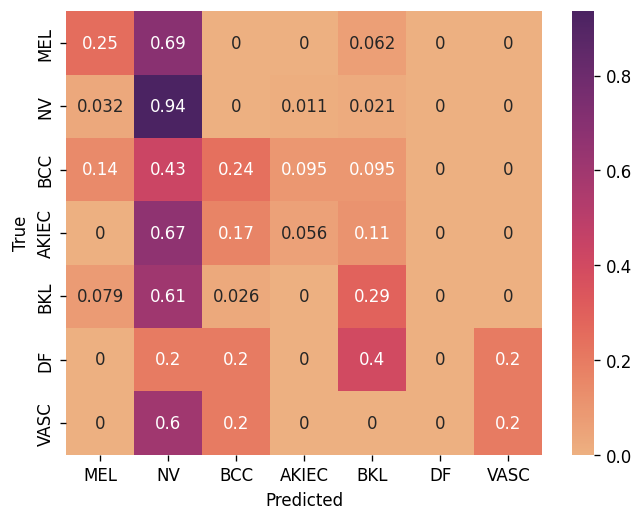


Figure 1. Confusion matrix

After the initial training run with a simple baseline convolutional neural network with 5 convolutional layers, a Confusion Matrix was produced. As shown in Figure 1,

* MEL *(n = 32)* was predicted 25% of the time. Given that only 8.89 per cent of the train dataset is MEL images, it is unsurprising that the classifier did not learn its features well. Of the 32 images of the validation set, it was incorrectly predicted as the biggest class, NV, in 69% of instances.
* NV *(n = 281)* was correctly predicted 94% of the time. Given that 68.31 per cent of the train dataset is NV images, it is unsurprising that the classifier learned its features better than other types of skin lesions. Many classes were also incorrectly labelled as NV, likely due to the large proportion of images belonging to the class, causing a bias in the classifier.
* BCC *(n = 21)* was correctly predicted 24% of the time. Given that 5.03 per cent of the train dataset is BCC images, it is unsurprising that the classifier did not learn its features well. It was most often incorrectly predicted as NV, 43% of the time.
* AKIEC *(n = 18)* was almost never correctly predicted. Given that 3.79 per cent of the train dataset is AKIEC images, it is unsurprising that the classifier did not learn its features well. It was incorrectly predicted as the biggest class, NV, 67% of the time.
* BKL *(n = 38)* was predicted correctly 29% of the time. 11.28 per cent of the train dataset is BKL images, making it the second biggest class. When BKL images were incorrectly predicted, they were predicted as the biggest class, NV, 61% of the time.
* DF *(n = 5)* were never correctly predicted. Given just 1.2 per cent of the train data set is DF images, it is unsurprising.
* Similarly, VASC *(n = 5)* was correctly predicted only once. With only 1.5 per cent of the train dataset being VASC images, this result is also unsurprising.

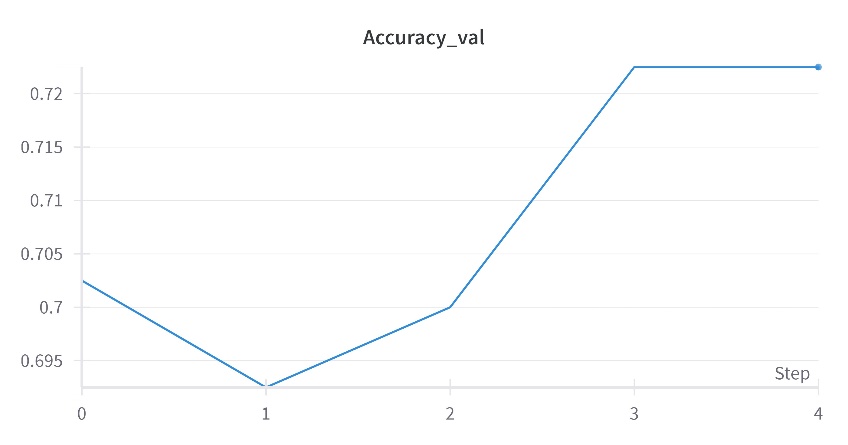


Figure 2. Validation accuracy

As seen in Figure 2, accuracy seems to rise quickly to around 70% when the model learns that it can predict the NV in most instances and be correct. 68.31% of the training data is NV images. This is because accuracy does not consider class imbalance, and is instead calculated as:

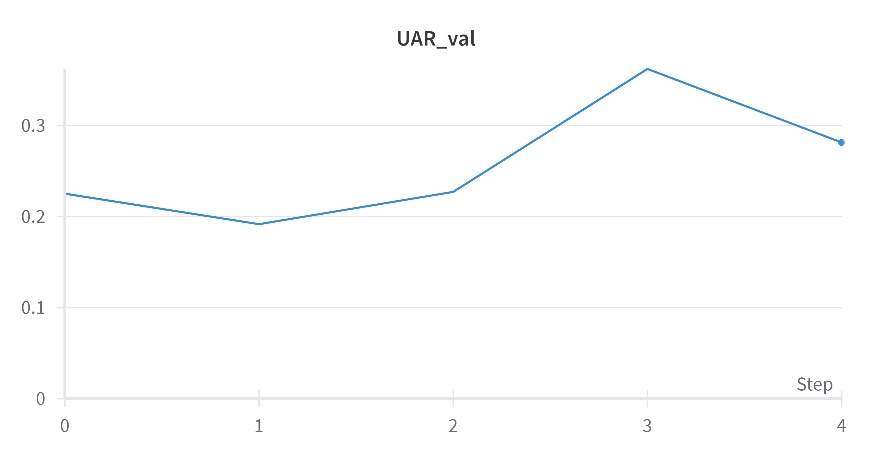


Figure 3. Validation UAR

As seen in Figure 3, Unweighted Average Recall (UAR) shows that the average recall across all classes plateaued around 30%. This is a better indicator of model performance than Accuracy as there is a certain class imbalance. UAR can be interpreted as:

It is clear that the classifier is heavily biased towards predicting the largest class, the NV class.

## Task 1d

### Account for data issues

It is pertinent to address the class imbalances.

To account for the class imbalance, which is resulting in the classifier being biased towards the large proportion of NV samples, a WeightedRandomSampler was used. Specifically, the weight for each class was calculated using the formula

where is the proportion of the class. For example,

The weights were placed in a list that corresponds with each example in the dataset. The examples in the dataset were selected by the DataLoader based on the probabilities assigned to them by the weights. Based on the new Confusion Matrix in Figure 4, this method was effective.

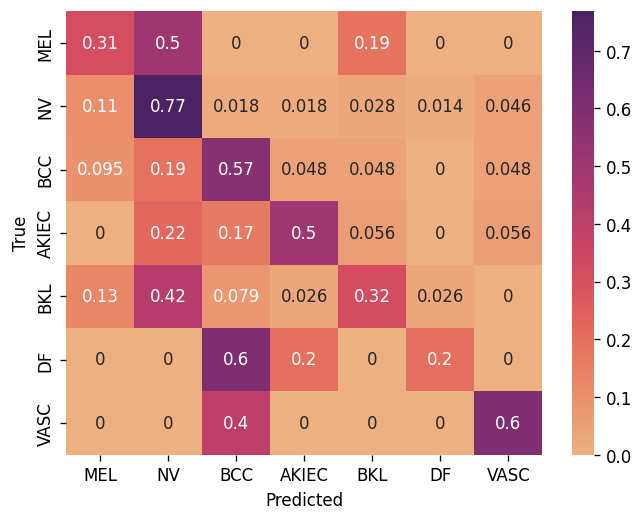


Figure 4. Confusion Matrix for training run with weighted sampler

As can be seen from Figure 1 to Figure 4,

* MEL *(n = 32)* correct predictions increased from 25% to 31%.
* NV *(n = 281)* correct predictions reduced from 94% to 77%.
* BCC *(n = 21)* correct predictions increased from 24% to 57%.
* AKIEC *(n = 18)* correct predictions increased from 5.6% to 50%.
* BKL *(n = 38)* correct predictions increased from 29% to 32%.
* DF *(n = 5)* correct predictions increased from 0% to 20%.
* VASC *(n = 5)* correct predictions increased from 20% to 60%.

Overall, the use of the weighted sampler resulted in a lower overall accuracy score on the validation set (65.75% compared to the previous 72.25%). However, the use of the weighted sampler resulted in a much higher UAR score (46.69% compared to the previous 28.13%), showing that the model has improved at distinguishing between classes.

## Task 1e

### Vertical Flips

Three non-deterministic data augmentations techniques have been applied to the training data set. Firstly, random vertical flips have been used to augment this data. Vertical flips are not always appropriate for data augmentation as they often do not preserve the natural features of an image. However, in this dataset they are appropriate. This is because skin lesions have a cellular structure, and the photos of the skin lesions could have been captured from any orientation. Therefore, they can be viewed from any orientation.

Two other non-deterministic data augmentation techniques were applied. These included *random horizontal flips* and *random rotation (up to 30 degrees)*.

All three of the non-deterministic data augmentation techniques were passed through as a list of transforms to *random apply*, which applies the three transformations 50% of the time and doesn’t apply any transformations the other 50% of the time. This is because, in theory, it is good practice to retain some of the original images for training, as well as augment the images to create some noise so that the classifier can better learn the features that will enable it to classify the images with greater accuracy.

The *random rotation* technique was specifically chosen to help the classifier better learn patterns within the data, as it often rotates the corner pixels out of bounds and changes the corner pixels to 0. It is hoped that the classifier will then put further importance on the pixels in the centre, which are the most essential features.

### Effect of Augmentation

The effects of the augmentation increased overall accuracy to 71% (from 65% in the previous model) but UAR decreased from 47% to 44%. The confusion matrix as well as plots for accuracy and UAR for this run can be seen in Figure 5 below.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Figures 5, 6 & 7. Confusion matrix, accuracy plot and UAR plot for validation set with weighted sampler and augmented images

As evident from Figures 5, the model performance decreased. On additional runs, it often decreased even further. This may have eventuated as the classifier found it more difficult to learn the essential features. This could have also been further amplified by *random rotation*. It is possible that the disparity between the corner pixels (0s) and the other pixels during *random rotation* were learned as features. Therefore, centre cropping was also applied which removed the effect of 0s in the corners from *random rotation*. The centre is also where the most interesting information is contained, regardless of *random rotation*. The results after *centre cropping* to a size of 299can be seen in Figure 6 below.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Figure 8, 9 & 10. Confusion matrix, accuracy plot and UAR plot for validation set using weighted sampler, the data augmentation from the previous step and centre cropping

Accuracy reduced to 66.25% while UAR further reduced to 42.81% on the validation set. Still, to date, the best performance was found using the weighted sampler without any augmentation. However, the performance with augmentation looks like the model is improving from epoch to epoch (as shown in Figures 9 & 10), therefore it may be worth maintaining these augmentations and increasing the number of epochs in future trials. It may also be worth considering different augmentations as well as normalising the data before augmentation.

In a research paper by Perez, Vasconcelos, Avila & Valle (2018), it was found that the most effective augmentation techniques for skin lesions consisted of *random crops*, *affine*, *random flipping* and *changing the saturation, contrast, saturation and hue.* For this reason, the data augmentation techniques were changed to include *random crops* and *affine* before *random horizontal* and *random vertical flips*, as well as *ColorJitter* after these *flips*. These changes to augmentation on the training set were put into effect after all data was first normalised. Interestingly, this had a negative effect on performance as can be seen in Figures 10-12. These augmentations were used in conjunction with the ResNet18 model architecture (rather than the original baseline convolutional neural network). Although ResNet18 previously yielded the best results so far, these augmentations decreased accuracy and UAR.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Figure 10, 11 & 12. Confusion matrix, accuracy plot and UAR plot for validation set using weighted sampler, the Resnet18 model architecture and Perez, et al.’s (2018) augmentations

It was later found that removing the ColorJitter but maintaining Perez, et al.’s (2018) other augmentations led to the best result. The natural colours of the skin lesions appear to be important for model learning and classification.

### [Challenge] 5 crop augmentation

A challenge in this assignment was to *“Apply 5 crop augmentation with crop size 200x300. Make a distinct model which uses 5 crops at once to give a single answer. Include in your report how you did this and report the effect on performance”*. Theoretically, this should have increased model performance, particularly as Perez, et al. (2018) also found that 144 crops resulted in increased performance as their model benefitted from different representations of the same input image. However, the results from this challenge were underwhelming. Although this challenge was completed, it is not clear as to whether the implementation was satisfactory, particularly given the unimpressive results.

Firstly, a five cropping function was defined whereby the image was first resized to (400, 600) from (450,600). This is because (400,600) allows for symmetrical five crops of (200,300) as specified in the challenge, for example:

|  |  |
| --- | --- |
|  |  |
| *Figure 13. Code to transform the images for Five Crops. Figure 14. Example of evenly distributed 5 cropping* | |

Next, the datasets.py code was further edited to place each of the five crops in list of tensors. This list of five tensors were then stacked using torch.stack() to create one single tensor. The channels and crops were then concatenated, therefore effectively creating 15 channels (5 crops \* 3 channels). Having one single tensor then allowed the \_\_getitem\_ function in the LesionDataset to operate as usual.

|  |  |
| --- | --- |
|  |  |
| *Figures 15-16. Code to concatenate the tensors and to pass these through the model architecture.* | | | |

Following this, a new model was initiated in models.py which takes 15 channels as input. The tensors then go through the convolutional layers before arriving at 2 fully connected layers. The architecture for these layers is very similar to the SimpleBNConv network, other than the increase in number of input channels from 3 to 15 and one less convolutional layer (15 to 32 to 64 to 128) as opposed to (3 to 16 to 32 to 64 to 128). The results of this experiment led to the worst performance of any of the classifiers trialled in this project. This may be due to the model architecture as it is shallower with less convolutional layers. The results can be found below in Figures 15-17.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Figures 17-19. Confusion matrix, accuracy plot and UAR plot for validation set using weighted sampler and 5-crop data augmentations.

## Task 1f

### Experiments

Finally, the model from all experimentation that led to the best performance included:

* The ResNet18 pre-trained model architecture, with all weights finely tuned.
* The Adam optimiser with a learning rate of 0.0001.
* A weighted sampler to address the class imbalance
* Augmentations applied as per recommendations of Perez, et al. (2018) minus ColorJitter.

For a detailed list of all changes and their impacts from experiment-to-experiment, please refer to the attached Excel document and the [Weights and Biases report](https://api.wandb.ai/links/mfinster/8izviz8c) via this link.

In summary, however, the experimental results show that the most impactful changes were:

1. Addressing the class imbalance through weighted sampling.
2. Implementing and optimising data augmentation techniques based on prior research.
3. Utilising a pre-trained ResNet18 model with a greater number of convolutional layers.
4. Fine-tuning the learning rate.

Initially, addressing class imbalance through the implementation of a weighted sampler significantly enhanced the unweighted average recall (UAR), which was crucial given the initial imbalance observed towards the NV class.

Additionally, incorporating data augmentation techniques was another critical factor. Originally, the data augmentation techniques trialled in early runs were not successful and, in fact, led to worse performance. By adapting previous research findings from Perez et al. (2018), better augmentation strategies were implemented. Perez, et al. (2018) found that random cropping, affine transformations, colour changes and various flips helped to improve model performance on skin lesion image classification tasks. Although colour changes were eventually removed from this model, the other changes recommended by Perez, et al. (2018) were instrumental in the final model’s performance.

The transition to the pre-trained ResNet18 model also marked a significant improvement in our results. The ResNet18, with 20 convolutional layers, provided a more sophisticated and deeper architecture compared to the initial custom-built model with only 5 convolutional layers. This depth was important in achieving the higher accuracy and UAR, as the pre-trained model could likely capture more complex features from the data.

Lastly, fine-tuning hyperparameters, particularly the learning rate, resulted in further increases in performance. The optimal setting was found to be 0.0001 using the Adam optimizer. This fine-tuning allowed for more precise weight updates, thus likely enhancing the model's ability to learn more effectively.

Again, for a detailed list of all changes and their impacts, please refer to the attached Excel document and the Weights and Biases report.

### [Challenge] Batch size

This challenge question asks, *“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?”*

After each batch, the loss (difference between predictions and actual classes) is calculated and the weights (parameters within the neural network) are updated corresponding to the amount of loss. Halving the batch size will result in double the amount of times that the weights are updated per epoch. For example, if we have a sample size of 400, and the batch size is 64, then the weights will be updated using 64 samples 6 times, before a final batch of 16. However, if we have a same sample size of 400 with a batch size of 32, the weights will be updated 12 times using 32 samples each time, before a final batch of 16.

Weights are likely to receive significant updates after greater loss compared to smaller loss, as the model learns by minimising the loss function. This means that if a batch has a particularly difficult set of images that it misclassifies, the loss be a lot higher in a batch size of 32 compared to a batch size of 64. This is because the batch of 64 is more likely to have a mix of difficult and easy images to classify, therefore the loss will even out. The gradient of the loss function is therefore likely to look much smoother in larger batches. Given that the loss in a small batch could swing from small to large more often due to the smaller number of samples, it can appear erratic on a graph and it can appear like the model is not learning.

This could be a good thing under certain circumstances where perhaps there is a large number of easy-to-classify images that are similar to one another in the training split. By ‘penalising’ the model more harshly when difficult images are presented and misclassified within a batch, the model can potentially learn faster.

# References

Perez, F., Vasconcelos, C., Avila, S., & Valle, E. (2018). Data Augmentation for Skin Lesion Analysis. Dans *Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis* (pp. 303–311). Springer International Publishing. doi:https://arxiv.org/pdf/1809.01442