Deep Learning Assignment 2023

Dr. Gorkem Saygili

Deadline: October 6 2023 (Sharp)

1 Practical Aspects

The sections below described important practical aspects of the group assignment for the TiU Deep Learning Course, BLOCK 1, 2023.

1.1 Group Assignment

The assignment is to be made in (small) groups consisting of 3–4 students. Groups will **not** be pre-assigned by the lecturer. As such, you will be required to organise yourselves into groups. If you do not manage to join/form a group of **3–4** people within a week (i.e. by the **18th of September 2023**), you will be randomly assigned to an existing group of 3 (or merged with another group of less than 3 people). Groups formed with only 3 people should be aware of the fact that a 4th person might be added to their group.

1.2 Grading

The hands-on assignment will count for **30% of your total course grade**. The grades will be based on the quality of your work as judged by the instructor based on your report and code. There might be hardware limitations in order to train very complex neural networks. You are encouraged to mention improvements to your classifier that you could implement if you had more resources available.

The assignment itself does not necessarily require access to high-end GPUs or a very powerful machine. If you do want to experiment with GPUs or TPUs, you could use the facilities provided by Google Colab[1] as you have been using for the practical sessions.

Passing the assignment is not mandatory to pass the course but it is highly advisable. As it is not compulsory, **there will be no resit**. The exam may include questions that are easier to answer if you have worked actively on the assignment.

The assignment will be graded on 10 points (may be scaled to 100 by multiplying with 10 on Canvas), a detailed description of all the required components can be found in Section 2. The grades of each component are outlined as below:

- A baseline model along with the required plots and performance metrics (2 point).

[1] https://colab.research.google.com

- Exploratory data analysis steps (Visualization of sample images, and distribution of classes) (1 point).
- An improved model, the required plots and performance metrics, an explanation and discussion of the experiments, and the results obtained, including class-based performance variations and reasons behind them (5 points).
- An extensive discussion about possible different networks or improvements to further enhance the performance of the algorithms relying on the existing literature (such as hybrid architectures, adding special layers (i.e. drop-out)) (1 point).
- Using transfer learning (VGG16, ResNet50, or DenseNet) to improve performance (1 point).

2 Task Description

The assignment focuses on the task of detecting and diagnosing various skin lesions and abnormalities using computer vision techniques.

2.1 Skin Lesion Classification

Image classification is one of the most studied tasks in computer vision. The milestone paper [1], AlexNet, proposed a CNN architecture with ReLU activation functions and dropout layers to achieve accurate image classification results on the ImageNet classification challenge. For the assignment, you will be using the Skin Cancer dataset by the International Skin Imaging Collaboration (ISIC) [2].

After downloading and preparing the datasets, your assignment is to:

- Create a virtual environment and install tensorflow, matplotlib, pandas, keras, seaborn libraries. These are the libraries we recommend but you can also install the ones of your choice.
- Create train, validation, and test sets using stratified train-test splits with ratios of 0.2 (for both), and use a random seed of 42 for reproducibility.
- Normalize your image data to floating point numbers between 0 and 1. Convert also the target labels to floats.
- Randomly select 15 samples from the dataset. For each selected sample, display the image along with its corresponding label as text on top of the image. Arrange these images and labels in a single figure, ensuring that they are visually clear and labeled appropriately.
- Create a bar plot to visualize the class label distribution of the dataset. (Hint: this bar plot reveals how many samples the dataset has for each class)

- Implement the baseline CNN algorithm (exactly, without any modification) that is shown in Fig. 1. It is a network consisting of: two consecutive convolutional layers with 64 and 32 filters of size 3×3 with ReLU activations followed by a max pooling layer of size 2×2 and this block of convolutional layer followed by a max pooling layer is repeated two times. Finally, two dense layers of sizes 32 with Relu activation function and an output layer with the proper activation function (you are expected to find out) are added. The number of epochs should be set as 10 and batch size as 32. The optimizer is Adam, the metric should be accuracy and the loss function is expected from you :-).

- Analyze the performance of the baseline by plotting: (i) the training and validation losses and accuracies on the training and validation set (similar to Fig. 2a and Fig. 2b), (ii) the Receiver Operator Characteristic (ROC) curve with the Area under the Curve (AUC) score and a confusion matrix for the validation and test set. Examples of accuracy and loss plots are shown in Fig. 2, and an example of a ROC curve and confusion matrix is shown in Fig. 3, respectively. Report performance measures (accuracy, sensitivity, specificity, and F1-score). You can find the hint for plotting multi-class ROC curve here.

- Once you have a baseline model, adapt/fine-tune the network to improve its performance by: (i) changing the hyper-parameters (e.g. add more layers) and/or (ii) applying data augmentation techniques. Illustrate the improvements of your new network over the baseline by: (a) plotting the ROC curve with AUC score and (b) reporting performance measures. Compare and explain the differences between the two models as well as potential reasons behind the increase in performance.

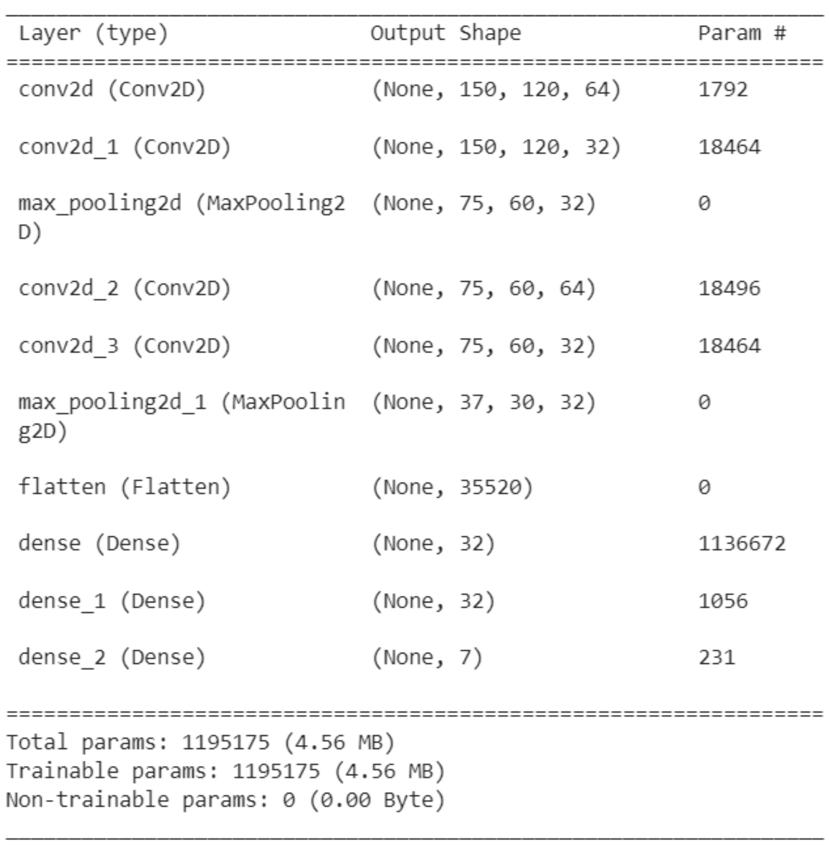
- Next, train a new model using transfer learning. Utilize VGG16, ResNet50 or DenseNet architecture for feature extraction. Freeze the layers until the fully connected layer such that these layers will not be updated through training. Add your fully connected layers (as many as you like) and present the results that you obtained on the test set (ROC curve with AUC score, performance measures, and confusion matrix). Comment on the performance with respect to the baseline and the network that you designed in the previous step.

2.2 Dataset

The dataset is available online and can be downloaded from:

`https://challenge.isic-archive.com/data/#2018`.
This dataset has many versions between 2016-2020. You are supposed to use only the training data from 2018. This dataset is entitled as 'Training data' under the section 2018 (10015 images (2.6 GB of size) and 1 ground truth response CSV file containing 1 header row and 10015 corresponding response rows). Additionally, you are required to download the 'Training Ground Truth', which provides the target values corresponding to the training data.

Figure 1: Baseline CNN Algorithm for image classification



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 150, 120, 64)      1792

 conv2d_1 (Conv2D)           (None, 150, 120, 32)      18464

 max_pooling2d (MaxPooling2  (None, 75, 60, 32)        0
 D)

 conv2d_2 (Conv2D)           (None, 75, 60, 64)        18496

 conv2d_3 (Conv2D)           (None, 75, 60, 32)        18464

 max_pooling2d_1 (MaxPoolin  (None, 37, 30, 32)        0
 g2D)

 flatten (Flatten)           (None, 35520)             0

 dense (Dense)               (None, 32)                1136672

 dense_1 (Dense)             (None, 32)                1056

 dense_2 (Dense)             (None, 7)                 231

=================================================================
Total params: 1195175 (4.56 MB)
Trainable params: 1195175 (4.56 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

The dataset consists of dermatoscopic images of pigmented skin lesions. It is designed for research and development in the field of skin cancer detection and classification. The dataset contains images of various types of skin lesions, including Melanoma, Melanocytic nevi, Basal cell carcinoma, Actinic keratoses and intraepithelial carcinoma, Benign keratosis-like lesions, Dermatofibroma, and Vascular lesions (in total seven class labels).

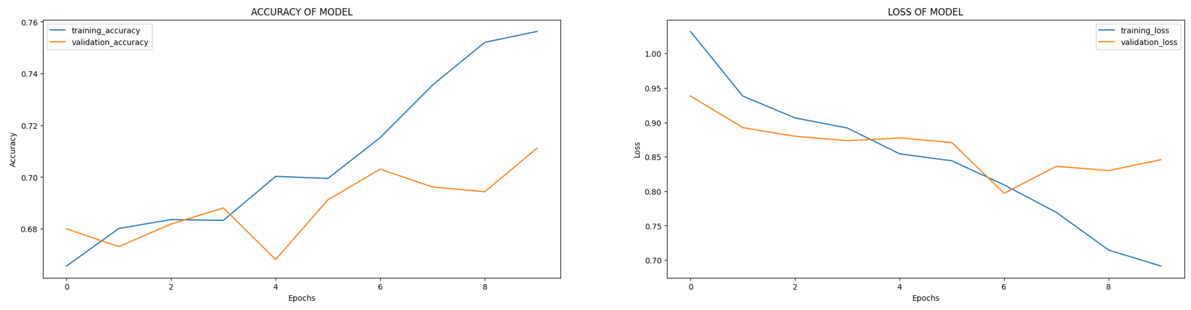
For the sake of reducing computational workload and memory requirements, you are required to resize all images during the preprocessing stage from their original size of 450×600 to a smaller dimension of 120×150.

You should use the provided code to load, resize, and save images. You are expected to use the shared ipynb file for all of the assignment.

3 Important Dates and Deliverables

3.1 Report

A 4 page (excluding references, title page) group report should be submitted by **Tuesday, October 6, 2023**. The report should include the following information:



(a) Training and validation accuracy

(b) Training and validation loss

Figure 2: Examples of accuracy and loss plots

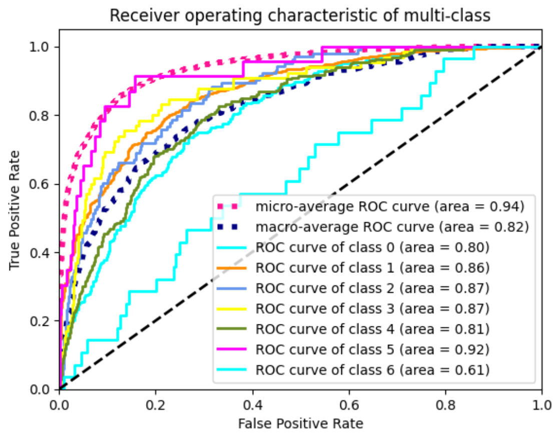
- Title
- Group Number
- Student names and Student numbers
- A summary of your models (excluding the baseline model) - similar to the one provided in Fig. 3.
- A brief description of your experiments, including possible pre-processing steps, training, hyperparameters, activation functions, optimization/regularization techniques so on or any other changes you made.
- The graphs/results requested in the task description, see Sec. 2.1.
- All the .ipynb code (that produces the results presented in your report) (ipython notebook file) which includes your results. Important: The results that you present in the report have to be in the ipynb file.

3.2 Code

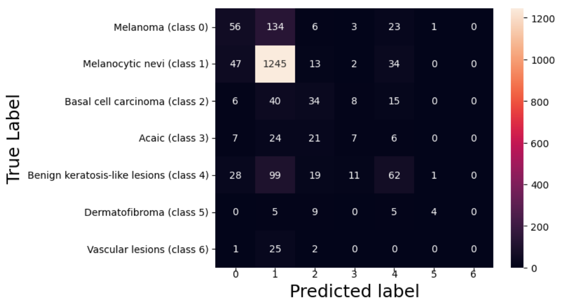
You can find an ipynb notebook including the codes which will help you to read and resize the data. You are expected to use the provided code to read and resize datasets, add the required normalization and train-test split steps that were mentioned above, and continue implementing your algorithms which can be run to generate your predictions. Your plots from your run should be seen in the .ipynb file that you submit together with your report (you should provide the results that are in your report). You do not need to submit your training data or the trained model. We strongly recommend you use Keras.

3.3 Submission format

Only one student from the group should upload the group's Jupyter Notebook (.ipynb) file with all visible outputs (all the results, including code execution outputs and any printed



(a) ROC curve



(b) Confusion Matrix

Figure 3: Examples of ROC curve and confusion matrix

or displayed information) to the 'Deep Learning Assignment' section on Canvas. Additionally, each student in the group is responsible for uploading their individual reports to the 'Assignment Individual Report Submission' section.

Very important remark: The reports have to be individual (each student has to write their own report). Only the quantitative results and visualizations can be similar between group members, but no text should be shared between students. Firm action will be taken in the event that academic fraud is discovered.

References

[1] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Communications of the ACM 60.6 (2017): 84-90.

[2] Skin Cancer Dataset: https://challenge.isic-archive.com/data/#2018, accessed on 10/09/23.