Deep Learning Assignment Spring 2024

Dr. Gorkem Saygili and Busra Ozgode Yigin

Deadline: March 10, 2024

1 Practical Aspects

This is an **individual** assignment. Please consider the rubric on the canvas page for this assignment before the submission. The sections below describe important practical aspects of the assignment for the TiU Deep Learning Course, BLOCK 3, 2024.

1.1 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. 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 points).
- Preprocessing (for one-hot-encoded target labels) and exploratory data analysis steps (Visualization of sample images, and distribution of classes) (0.5 point).
- An improved model, with required plots and performance metrics, an explanation/justification, and discussion of the experiments, implementation choices, and the results obtained, including class-based performance variations and underlying reasons (5 points).
- An extensive discussion about possible different networks or improvements to further enhance the performance of the algorithms relying on the existing literature (1 point).

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

- Using transfer learning (VGG16, ResNet50, or DenseNet) to improve performance (1 point).
- Good coding standards with documentation (Comments) (0.5 points)

2 Task Description

The assignment focuses on the task of detecting and diagnosing various lung and colon cancers using computer vision techniques.

2.1 Lung and Colon Cancer 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 Lung and Colon Cancer Histopathological Image Dataset (LC25000) [2].

After downloading and preparing the datasets using the provided notebook file, 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.
- Convert the target values using the proper function for one hot encoding.
- 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 properly.
- 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)
- Create train, validation, and test sets by performing **stratified train-test splits** with **proportions of 60% for training, 20% for validation, and 20% for the test set**, using a random seed of 42 for reproducibility.
- Implement the baseline CNN algorithm (exactly, **without any modification** for both model and dataset) that is shown in Fig. 1. It is a network consisting of: two consecutive Convolutional layers with 128 and 64 filters of size 3×3 with ReLU activations. Each convolutional layer is followed by a MaxPooling layer with a size of 2×2. Finally, two dense layers of sizes 128 and 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 should be Adam, the metric should be accuracy and the loss function is expected from you :-).

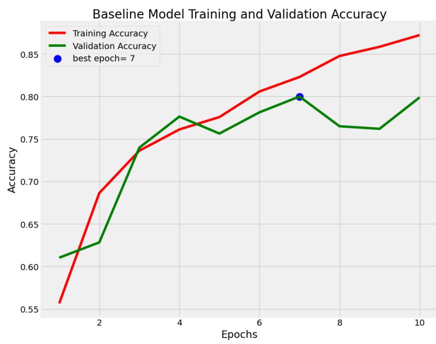
Figure 1: Baseline CNN Algorithm for image classification

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 120, 120, 128)     3584

 max_pooling2d_2 (MaxPoolin  (None, 60, 60, 128)       0
 g2D)

 conv2d_3 (Conv2D)           (None, 60, 60, 64)        73792

 max_pooling2d_3 (MaxPoolin  (None, 30, 30, 64)        0
 g2D)

 flatten_1 (Flatten)         (None, 57600)             0

 dense_3 (Dense)             (None, 128)               7372928

 dense_4 (Dense)             (None, 32)                4128

 dense_5 (Dense)             (None, 5)                 165

=================================================================
Total params: 7454597 (28.44 MB)
Trainable params: 7454597 (28.44 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

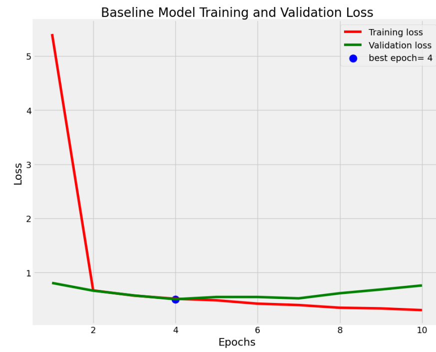
- Analyze the performance of the baseline by plotting: (i) the training and validation losses and accuracies on the training and validation set through epochs (similar to Fig. 2a and Fig. 2b), (ii) the Receiver Operator Characteristic (ROC) curve with the Area under the Curve (AUC) score and a confusion matrix **for the validation and test sets**. 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, precision, recall, and F1-score) **for both validation and test sets**. 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. It is critical to intentionally tweak at least six different hyperparameters, ensuring that these adjustments are not randomly chosen. You are expected to make purposeful selections based on the intricacies of the problem at hand, and conduct optimization within a logical framework. Consider changes such as the addition of more layers, alterations in filter sizes and numbers, adjustments to activation functions, fine-tuning the learning rate together with different optimizers, and experimenting with the number of neurons in Dense layers. Document and analyze the rationale/reasoning behind each modification in your report. Illustrate the improvements of your new network over the baseline by: (a) plotting the ROC curve with the AUC score; and (b) reporting performance measures. Compare and explain the differences between the two models as well as potential reasons behind the increase/decrease in performance.

- Add an extensive discussion about possible different networks or improvements to



(a) Training and validation accuracy



(b) Training and validation loss

Figure 2: Examples of accuracy and loss plots. They shouldn't have to be in this format exactly.

further enhance the performance of the algorithms relying on the existing literature (such as hybrid architectures and architectures that have been specifically used for similar problems before) (1 point).

- Next, train a new model using transfer learning. Utilize either VGG16, ResNet50, or DenseNet121 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.

Tip: As you work on your assignments, please bear memory constraints in mind, especially when using platforms like Google Colab. While normalization is a common preprocessing step, it may not always be necessary and can impact memory usage. So you can skip the normalization process in this assignment, to prevent potential memory issues.

2.2 Dataset

The dataset is available online and can be downloaded from:

https://academictorrents.com/details/7a638ed187a6180fd6e464b3666a6ea0499af4af.

This dataset contains 25,000 histopathological images with 5 classes. All images are 768 x 768 pixels in size and are in jpeg file format. The images were generated from an original sample of HIPAA compliant and validated sources, consisting of 750 total images of lung tissue (250 benign lung tissue, 250 lung adenocarcinomas, and 250 lung squamous cell carcinomas) and 500 total images of colon tissue (250 benign colon tissue and 250 colon adenocarcinomas) and augmented to 25,000 using the Augmentor package.

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 768×768

to a smaller dimension of 120×120.

You should use the provided code to load, resize, and save images. You are expected to use the same notebook(ipynb) file for the rest of the tasks.

3 Important Dates and Deliverables

3.1 Report

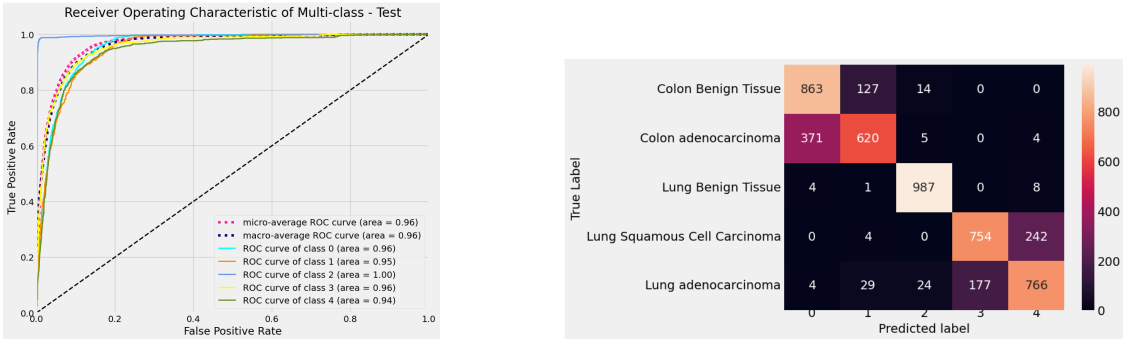
A **6-page (excluding references, title page)** individual report should be submitted by **Sunday, March 10, 2024 until 23:59 (sharp)**. **Assignments exceeding the page limit will not be considered for evaluation.**

The report should include the following information:

- Title
- Student name and number
- A summary of your models (excluding the baseline model) - similar to the one provided in Fig. 3. To make optimum use of the page limit, you do not need to include the information about the Baseline model (model summary) in the report. You only need to include the results you use for model evaluation/comparison.
- 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 such as **justification about your choices for further improvements**. Since specific values for all hyperparameters of the baseline model are explicitly requested in the assignment, there is no need to provide explanations and justifications for baseline hyperparameters. **Please reserve explanations and justifications only for your own choices.**
- 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. Otherwise, **results not included in the ipynb file will not be considered for evaluation**.
- Bibliography

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 exploratory data analysis and train-test split steps that were mentioned above, and continue implementing your algorithms which can be run to generate your results. **Your plots from your run should be observable in the .ipynb file** that you submit together



(a) ROC curve

(b) Confusion Matrix

Figure 3: Examples of ROC curve and confusion matrix. They shouldn't have to be in this format exactly.

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.

3.3 Submission format

Each student is required to upload their Jupyter Notebook (.ipynb) file to the 'Deep Learning Assignment' section on Canvas. If you plan to upload more than one notebook file (it is better to have it in a single file) please merge them into a zip file before uploading and make sure to put your name in each file. Ensure that all outputs, including results, code execution outputs, and any printed or displayed information, are visible in the notebook. The code parts that give an error will not be considered for evaluation. At the top of your file, please reference any code, methods, or ideas that are not your own or not provided in this course. Remember that these codes must be publicly available and free to use. You do not have to reference the course worksheets, the textbook nor the lecture slides. Please also include any special instructions that are required to run your code.

Every student must submit their reports as a .pdf file to the 'Assignment Report Submission' section on Canvas.

Very important remark: It is imperative for each student to produce their work independently and individually. Any instances of academic fraud or plagiarism will be dealt with seriously. These changes are in effect to ensure a fair and individual assessment for each student. Please adhere to these guidelines for a smooth and transparent evaluation process.

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://academictorrents.com/details/7a638ed187a6180fd6e464b3666a6ea0499af4af, accessed on 15/12/23.