Deep Learning Assignment Spring 2025

Rastislav Hronsky and Dr. Gorkem Saygili

Deadline: at 23:59, 2nd of March 2025 (Sharp)

1 Practical Aspects

This is a **group** 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, 2025. The groups should be formed by a minimum of **four** or a maximum of **six** students. Students need to mention **their contribution** to the project in their reports explicitly.

The group formation deadline is **on Tuesday 18.2., 23:59**. Students who are unable to assign themselves to a group but are willing to complete the assignment should contact the course co-lecturer (R.Hronsky@tilburguniversity.edu) as soon as possible, but before the group formation deadline. They will be placed in groups with fewer than four members or with other students in a similar situation. Not assigning yourself to any group and not contacting the co-lecturer within the specified time will be interpreted as not willing to participate in the group assignment.

It is crucial (and your own responsibility) to establish communication within the group as soon as possible. If there are unresponsive members in your group and you report it to the co-lecturer before the group formation deadline (but the earlier the better), it may still be possible to change your group.

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, you can use the facilities provided by the GPU4EDU[1] clusters or other cloud services.

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.

Passing the assignment is not mandatory to pass the course but it is highly advisable.

[1]A tutorial on how to get started can be found here https://ilk.uvt.nl/~shterion/gpu4edu.html

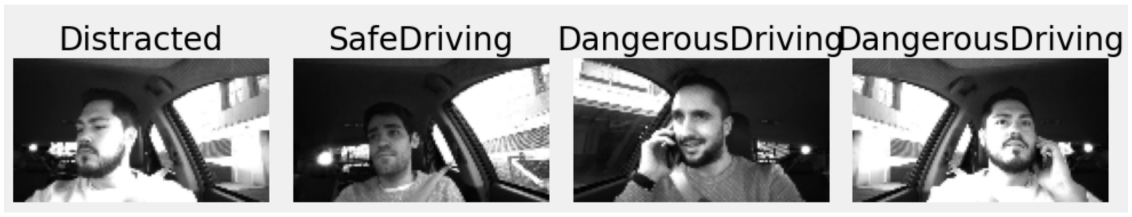


Figure 1: Example data points from the dataset.

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 3. The grades of each component are outlined as below:

- A baseline model along with the required plots and performance metrics (2 points).
- 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 (4 points).
- An extensive discussion about possible different networks or improvements to further enhance the performance of the algorithms relying on the existing literature (2 point).
- Using transfer learning (VGG16, ResNet50, or DenseNet) to improve performance (1 point).
- Discuss the potential risks, including ethical constraints, associated with deploying your algorithm in daily use. Consider aspects such as bias, privacy concerns, security vulnerabilities, and societal impact. (0.5 points)
- Good coding standards with explanatory comments (0.5 points)

3 Task Description

3.1 Driver Inattention Detection

For the assignment, you will be using Driver Inattention Dataset [1]. This dataset contains images of people while driving and is annotated according to the drivers' current state of attention. There are 6 possible states of attention: "dangerous driving", "distracted", "drinking", "safe driving", "sleepy driving", and "yawn" (see Figure 1 for examples).

After downloading and preparing the datasets using the provided notebook file, your assignment is to:

- 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)

- Use the provided validation set to validate your chosen model hyperparameters.

- Implement the baseline CNN model (exactly, **without any modification** for both model and dataset) that is shown in Fig. 2. It is a network consisting of:

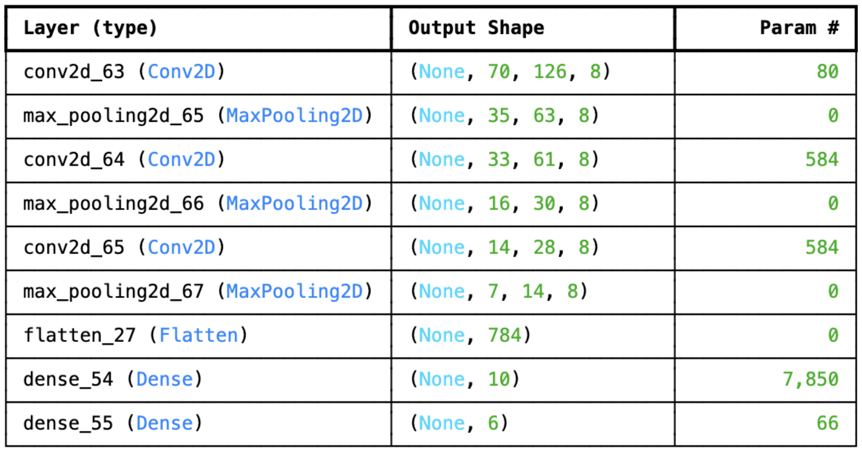
  1. 3 successive blocks, each containing a 2D convolutional (kernel size 3×3; 8 output channels) and a max-pooling layer (window size 2×2),
  2. a dense layer projecting a 784-dimensional space to 10 dimensions
  3. a dense layer projecting a 10 dimensions to 6 logits (1 for each class)

  Except for the last dense layer, all convolutional and dense layers should be followed by the ReLU activation function. The last layer activation is for you to determine.

- Train the model with 10 epochs (batch size is set to 32 as part of the provided preprocessing code). The optimizer should be 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 through epochs, (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. 3, and an example of a ROC curve and confusion matrix is shown in Fig. 4, 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.



Layer (type)	Output Shape	Param #
conv2d_63 (Conv2D)	(None, 70, 126, 8)	80
max_pooling2d_65 (MaxPooling2D)	(None, 35, 63, 8)	0
conv2d_64 (Conv2D)	(None, 33, 61, 8)	584
max_pooling2d_66 (MaxPooling2D)	(None, 16, 30, 8)	0
conv2d_65 (Conv2D)	(None, 14, 28, 8)	584
max_pooling2d_67 (MaxPooling2D)	(None, 7, 14, 8)	0
flatten_27 (Flatten)	(None, 784)	0
dense_54 (Dense)	(None, 10)	7,850
dense_55 (Dense)	(None, 6)	66

Figure 2: Baseline CNN Algorithm for image classification

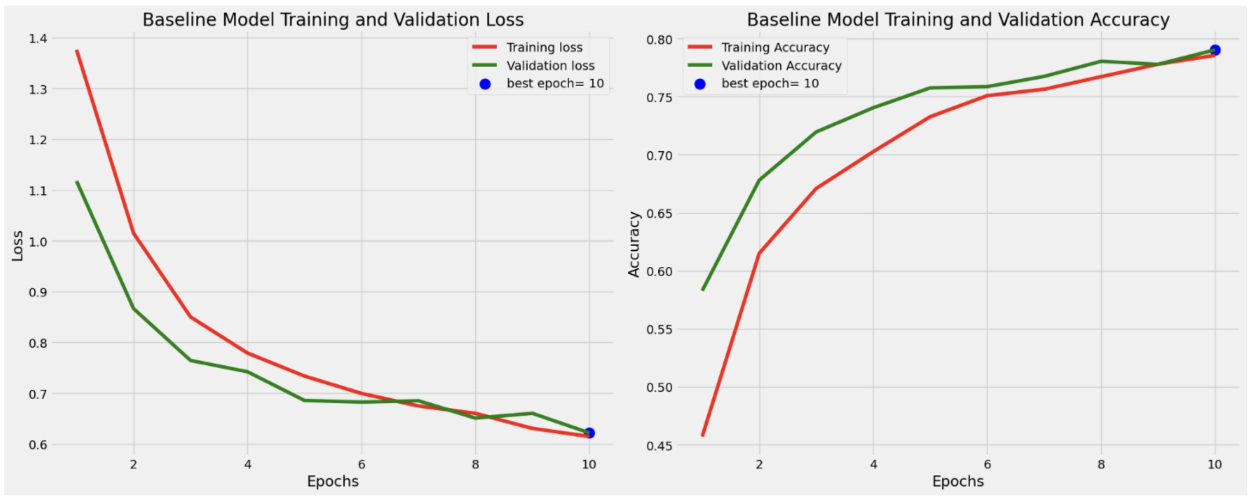


Figure 3: Examples of accuracy and loss plots. They shouldn't have to be in this format exactly.

- Add 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 and architectures that have been specifically used for similar problems before).
- 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.

3.2 Dataset

The dataset is available online and can be downloaded from:

https://www.kaggle.com/datasets/zeyad1mashhour/driver-inattention-detection-dataset

However, the starting code also contains a code cell to download the dataset into your working directory automatically – you may use it if you find it convenient.

For the sake of reducing computational workload and memory requirements, you will be working on a smaller dimension of 72×128 rather than the original resolution of the images.

Use the provided code to load the images. Your solution is required to use the data loading code provided in the starting code, thus you are not allowed to make changes to, for example, the image down-sampling or noise parameters to improve model performance. If you plan to use data augmentation techniques, you need to apply these without changing the original code. You are expected to use the same notebook(ipynb) file for the rest of the tasks.

4 Important Dates and Deliverables

4.1 Report

A **6-page (excluding references, title page)** group report should be submitted by **Sunday, March 2, 2025 until 23:59 (sharp)**. **Assignments exceeding the page limit will not be considered for evaluation.**

The report should include the following information:

- Title
- Student names and numbers
- **Explicitly mentioned contribution of each student.**
  For instance:
  student A: Hyperparameter optimization, Results Section, Discussion Section.

Student B: Baseline implementation, Results Section, Discussion Section.
Student C: Plotting test results, Introduction Section, Results Section.
Student D: Preprocessing of images, Methodology Section, Discussion Section, Conclusion.
Student E: Hyperparameter optimization, Results Section, Discussion Section, Future improvements, Limitations.

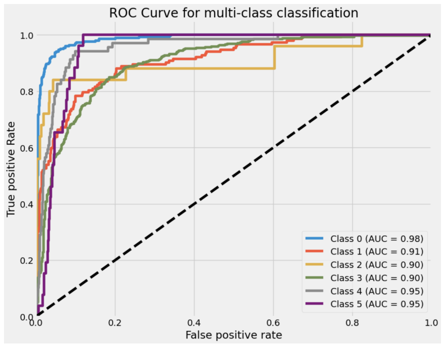
- A summary of your models (excluding the baseline model) - similar to the one provided in Fig. 4. To make optimum use of the page limit, you do not need to include the information about the Baseline model (model summary) in the report. You only need to include the results you use for model evaluation/comparison.
- 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. 3.1.
- 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

4.2 Code

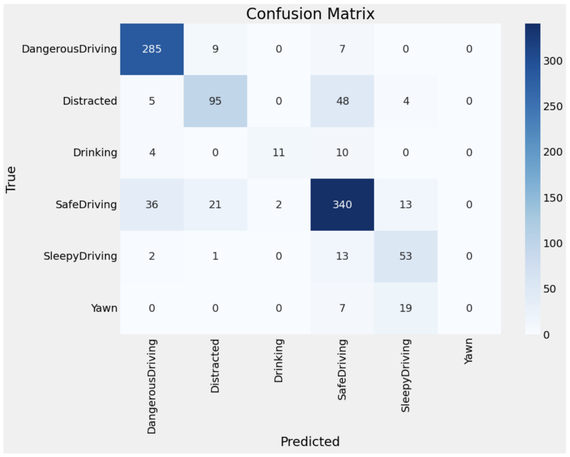
You can find an ipynb notebook including the codes which will help you to read and resize the data. You are expected to use the provided code to load and resize the dataset, and continue implementing your solution which generates the required figures. **Your plots from your run should be observable in 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. You may use comments or markdown cells to add explanatory remarks as you see fit.

4.3 Submission format

Each group is required to upload their Jupyter Notebook (.ipynb) file to the ' Assignment Code Submission' 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 group name in each file. Ensure that all outputs, including results, code



(a) ROC curve



(b) Confusion Matrix

Figure 4: Examples of ROC curve and confusion matrix.

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 result that you showed in your report should be also in the .ipynb notebook. Otherwise, you will not receive any credit for that result.

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

Very important remark: It is imperative for each group to produce their work independently from other groups. Deliverables will be run through plagiarism software to compare them to the literature, and deliverables of other students in this or other assignments. Any instances of academic fraud or plagiarism will be dealt with seriously. These changes are in effect to ensure a fair assessment for each group. Please adhere to these guidelines for a smooth and transparent evaluation process.

References

[1] Ortega, J., Kose, N., Cañas, P., Chao, M.a., Unnervik, A., Nieto, M., Otaegui, O., & Salgado, L. (2020). DMD: A Large-Scale Multi-Modal Driver Monitoring Dataset for Attention and Alertness Analysis. In: A. Bartoli & A. Fusiello (eds), Computer Vision – ECCV 2020 Workshops (pg. 387–405). Springer International Publishing.