# CI7520 –Assignment 2: Machine Learning with Deep Neural Networks

# MSc in Data Science and MSc in Games Development courses

Develop advanced machine learning and artificial intelligence solutions for applications such as video and image analysis, computer games, and data science.

Submission by **Friday 11th March 23:59** (70%, set and assessed by Vasileios Argyriou),

Team project: Max 4 Students in a Team

**Assignment 2: Machine Learning with Deep Neural Networks**

**Overview**

The subject of this assignment is the development of a Machine Learning Application with Deep Neural Networks using Python and Keras that were introduced in the lectures. This is a group assignment with maximum 4 students in a group.

**Essential Requirements**

You must meet all of these requirements. Exceptions with the individual agreement of your tutor may apply.

* The implementation language is Python and Keras
* User Interface will be provided (e.g. web based, mobile apps or other types may be considered).
* You may also freely use libraries, original sample code, as well as sample code provided by the Tutor (references should be provided)
* Usage of any other source code, not listed above, must be reported and referenced. Usage of any third party libraries must be agreed (by email) with the Tutor beforehand. Non-compliance may result in project being rejected.
* As a minimum solution you can consider using the examples discussed during the class.
* Any solutions that use classic ML *without* deep learning will be rejected.

**Essential Machine Learning Requirements**

You must meet all of these requirements.

* **Deploy existing DNN solutions related to your topic:** This part of the application requires to use at least one existing deep model with your input data.
* **Design, Train and Deploy your own DNN solution related to your topic:** This part expects to design, develop, train and evaluate at least one novel deep model.
* **Comparative Evaluation:** Use metrics to measure the performance of all the developed or used deep models. Metrics such as accuracy, F1, confusion matrix, etc. should be considered.
* **Interactive application with Graphical User Interface (GUI):** Design and build a GUI (e.g. web based, mobile app or an equivalent that fits with the project type) to allow the user to load and overview data, select and deploy a model among the existing and the developed ones, and finally to view the results.

**Topics**

Each team has to select one of the following topics and confirm their choice with the academics.

1. **Timeseries Anomaly Detection**

In this topic you need to build a deep learning based system that will allow real time anomaly detection for **timeseries**.

1. Load the data from the CSV files at the link below and verify no records are missing
2. We have a multivariant timeseries anomaly detection problem, so the data needs to be normalised and formatted considering that they are multivariant.
3. Visualise the data and the ground truth
4. The data needs to be further pre-processed. We have the option to consider each time record as an input which means we don’t consider that we have a timeseries or/and as a timeseries which expects to use a window of data samples (e.g. sample from *t-n* to *t+m*, where *t* is the current time instance and *n, m* the window length in the past and the future). In the second case the data needs to be reformatted using 2D representations.
5. Split the data into training and testing sets and consider that you can solve the problem as an anomaly detection problem and as a classification. Based on your approach the data needs to be separated to training and testing appropriately TIP: consider the issue that we have unbalanced data
6. Build and train at least few deep learning models with Keras. It is suggested to have one model (1) that considers the data individually at each time instance, one model (2) that considers them as a timeseries, one deep network (3) to perform as classification and one deep network (4) to handle the problem as anomaly detection. This will result in 4 separate models.
7. Evaluate the performance of the models on the test dataset and report the results in a comparative study (example metrics: F1, precision, recall, accuracy, confusion matrix, etc.)
8. Build a website or mobile app that will collect data in real time (e.g. every 5 seconds) using REST API, sends the data to a server (e.g. localhost or a free ones such as Heroku), execute the best deep model, and return to the app the outcome visualising if the incoming data are normal or not.
9. Link to the dataset <https://kingston.box.com/s/6x4jd5hiy59d720jreogvk1nbly1welu>
10. **Image Anomaly Detection**

In this topic you need to build a deep learning based system that will allow anomaly detection for **images**.

1. Load the data from the files at the link below and verify no records are missing
2. We have an image anomaly detection problem, so the data needs to be normalised and formatted considering that they are multidimensional.
3. Visualise a subset of the data and the ground truth
4. The data needs to be further pre-processed. We have the option to perform data augmentation aiming to improve the overall performance.
5. Split the data to training and testing sets and consider that you can solve the problem as an anomaly detection problem and as a classification. Based on your approach the data needs to be separated to training and testing appropriately TIP: consider the issue that we have unbalanced data
6. Build and train few deep learning models with Keras. It is suggested to have two models (2) that consider the problem as classification and other two (2) as anomaly detection. This will result in 4 separate models.
7. Evaluate the performance of the models on the test dataset and report the results in a comparative study (example metrics: F1, precision, recall, accuracy, confusion matrix, etc.)
8. Build a website or mobile app that will allow the user to upload or capture an image with the camera and using REST API to send the data to a server (e.g. localhost or a free ones such as Heroku), execute the best deep model, and return to the app the outcome visualising if the uploaded data/image is normal or not.
9. Link to the dataset <https://kingston.box.com/s/rlnzehay097dyhseof9ybcxm0p43wb5z>
10. **Medical Image Analysis and Classification**

In this topic you need to build a deep learning based system that will allow to classify medical images (DermaMNIST and RetinaMNIST data).

1. Load the data from the MedMNIST2D dataset for skin and retina only at the link below and verify no records are missing
2. We have a medical image analysis and classification problem, so the data needs to be normalised and formatted considering that they are multidimensional.
3. Visualise a subset of the data and the ground truth
4. The data needs to be further pre-processed. We have the option to perform data augmentation aiming to improve the overall performance.
5. Split the data to training and testing sets and consider that you can solve the problem as a classification. Consider the classes for each of the datasets and based on your approach the data needs to be separated to training and testing appropriately TIP: consider the issue that we may have unbalanced data
6. Build and train a few deep learning models with Keras. It is suggested to have two models (2) for DermaMNIST that consider the problem as classification and other two (2) for the RetinaMNIST dataset. This will result in 4 separate models.
7. Evaluate the performance of the models on the test dataset and report the results in a comparative study (example metrics: F1, precision, recall, accuracy, confusion matrix, etc.)
8. Build a website or mobile app that will allow the user to upload or capture an image with the camera and using REST API to send the data to a server (e.g. localhost or a free ones such as Heroku), execute the best deep model, and return to the app the outcome visualising the class of the uploaded data/image.
9. Link to the dataset <https://medmnist.com/>
10. **Game Level Generation**

In this topic you need to build a deep learning based system that will allow the user to generate dungeons and room decoration.

1. Load the data from the CSV files at the link below and verify no records are missing
2. We have a data/level generation problem, so the data needs to be normalised and formatted considering that they are multidimensional.
3. Visualise a subset of the data using image or heatmap representations
4. The data needs to be further pre-processed. We have the option to perform data augmentation aiming to improve the overall performance. It is welcomed if you want to extend the dataset manually and provide the outcomes for the new dataset.
5. Split the data to training and testing sets and consider that you can solve the problem as a data generation problem.
6. Build and train some deep learning models with Keras. It is suggested to have two models (2) for the dungeons and other two (2) for the rooms decoration dataset. This will result in 4 separate models.   
   *Another option here is to build (1) model for the dungeons and (1) for the rooms decoration, plus (1) algorithm to correct the model outcomes for the dungeons and (1) algorithm to correct the one for the rooms. This will result in 2 models and 2 correction algorithms.*
7. Evaluate the performance of the models on the test dataset and report the results in a comparative study (example metrics: FID, precision, recall, diversity, coverage, etc.)
8. Build a Unity, web or mobile app that will allow the user to run the models, generate a dungeon and decoration for the dungeon rooms. Then load the output in Unity and automatically build the 2D or 3D dungeon using predefined prefabs. The connection with the models and Unity can be implemented using REST API to call the server (e.g. localhost or a free ones such as Heroku), execute the best deep model, and return to the app/game the outcome visualising the dungeon.
9. Link to the dataset <https://kingston.box.com/s/oq4raj235ehcxfjoeozpxqny08wj9xpa>

**Group Report – [6-8 pages text and results (more pages if we include images/code parts/ references)]**

The Group Project Report should include:

* Aims and Objectives of your theme problem
* Analysis of the existing DNNs
* Analysis of the designed DNNs
* Analysis of the training process
* Analysis of the designed interactive application and the related GUI
* Comparative analysis and performance evaluation of all the DNNs used
* References (e.g. [IEEE](https://www.citethisforme.com/citation-generator/ieee), [Harvard](https://www.citethisforme.com/citation-generator/harvard), etc.).

**Individual Report – Conclusions and Self-evaluation [max 1 page]**

* Conclusions should include a short analysis of your own contribution (a paragraph 10 lines) and a reflection on your work and the expected outcomes (a paragraph 10 lines).
* Use the template in APPENDIX I to assess both the group submission and your own individual submission, as well as to rate the contributions of the group members towards the group submission.

**Learning outcomes being assessed**

* Select and specify suitable methods and algorithms relevant for a particular data analysis process;
* Build machine learning and artificial intelligence systems using software packages and/or specialised libraries;
* Articulate and demonstrate the specific problems associated with different phases or tasks of a machine learning or artificial intelligence pipeline;
* Assess and evaluate machine learning methods using datasets and appropriate criteria;
* Develop advanced machine learning and artificial intelligence solutions for applications such as video and image analysis, computer games, information security, data science and mechatronics.

**Marking Scheme (70%) [60% software and 10% the report + video]**

Following items will be concerned in the process of marking:

* Completed parts (30%)
* Code quality (10%)
* Effort/Complexity –what technical skills have been demonstrated (10%)
* Application quality – Highlights (10%)
* Quality of documentation + video (10%)

**Marking Scheme (100%)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Criterion** | **9-10** | **6-8** | **3-5** | **1-2** | **0** |
| **Completed parts (30)** | All the parts are completed and novel solutions and methods have been implemented, state of the art results on the selected datasets | All or most of the parts have been completed, the solution is working without errors | Half of the parts not completed, the solution is partially running or with errors | No method has been properly implemented, only few of the parts completed, or partially running | None |
| **Code quality (10)** | The code structure is excellent, the scientific code is well optimised and new deep learning structures (e.g. layers) were implemented, state of the art tools and components that are publishable | The code is working without problems or bugs, advanced programming structures are used (proper documentation, classes, inheritance, etc.) | Only few parts of the code are running and the quality is very low (no documentation, no proper structure, etc.) | No running code,  Tools and programming languages not specified were used | None |
| **Effort –what technical skills have been demonstrated (10)** | Implementation of state of the art novel ML solutions, comparative study, support of different datasets. | Advanced technical and scientific skills, advanced analysis and overview of the methodologies, diverse implementation to support any dataset | Few average skills and additional effort is present (more datasets, more advanced ML methods than the ones covered in the class, etc.) | Basic skills both for machine learning and programming, no extra effort noted | None |
| **Application quality – Highlights (10)** | A complete standalone solution, advanced visual analytics, dynamic interactions with models, support of big data or online platforms etc. | Advanced dynamic interface, improved backend, support of multiple modalities of data, etc. | Few highlights are available, such as dynamic data visualisation, more than the basic models trained and deployed | Basic contribution with the minimum required functionality | None |
| **Quality of documentation + video (10)** | Sharp report that clearly describes both the basic principles and the details of the work, including proper referencing.  Excellent video analysing all the parts of the solution, including info of the end users, tutorial style analysis, including the related maths, results and training approaches. | Clear description of basic principles of the work and satisfactory presentation of most of the details, including proper referencing.  Very good quality of the video, covering all the related parts, both scientific and implementation | Adequate description of the work clarifies the basic aspects of work, although many details may have been missed. Adequate Referencing.  The video quality is low, does not cover all the parts of the implementation | Poor description of the work that fails to clarify even basic aspects of the work. Inadequate Referencing.  Video not informative, not covering main contributions | None |

**Deliverables**

**[A] Canvas:** Online submission of the **individual** **report** using **Canvas. This is an individual submission with the Individual Report including the Conclusions and the Self-Evaluation (see APPENDIX I).**

**This report is not marked, but it helps to understand the contributions of each student in the group report and application. If it is not submitted then we will assume that the student didn’t contribute to the project.**

**[B] Box.com:** The project should be sent using the shared **BOX.COM** folder.

For each team a **Box.com** folder will be shared giving access only to upload the **project models, code and data**.

The project folder should contain:

* Runnable program of your solution accompanied with all the files required to run.
* All the datasets that are required to train and deploy the DNNs
* All the pre-trained models (both the ones found online and the ones the students trained)
* A copy of the group report
* A video demonstrating the application (no more than 250MB)

**If there are any technical issues and you cannot submit the assignment online, please email the BOX.COM link and your report to** Vasileios.Argyriou@kingston.ac.uk **before the deadline.**

**Mode of submission:** On line submission

**Late submissions:** for details please refer to the Module Guide.

Plagiarism is a serious academic offence and will always be challenged where it is suspected. When submitting work for assessment ensure it is your own and not someone else’s code, ideas, words and images. You must declare that the task submitted for assessment is your own independent work and that all sources used in the submission are referenced.

Please ensure that your name is on all components of the work presented.

If there are any ambiguities or requirements that you do not understand then tell me immediately either in person, or email:  [Vasileios.Argyriou@kingston.ac.uk](mailto:Vasileios.Argyriou@kingston.ac.uk)

**APPENDIX I – Self-Evaluation Template**

Include the following template at the end of your individual submission.

For each marking criterion:

* + - * Assess the work submitted, by providing a mark from 0-10 for each task of selected topic
      * Rate the contribution of each group member (including yourself) towards the group submissions (Tasks (a)-(h), group report and video). Use the following scale:

1. Did not contribute to this task
2. Willing but not very successful
3. Average
4. Above Average
5. Outstanding

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Contribution (1-5 for each group member, including yourself)** | | | |
| **Criterion** | **Mark (0-10)** | ***{member 1}*** | ***{member 2}*** | ***{member 3}*** | ***{member 4}*** |
| **Task (a)** |  |  |  |  |  |
| **Task (b)** |  |  |  |  |  |
| **Task (c)** |  |  |  |  |  |
| **Task (d)** |  |  |  |  |  |
| **Task (e)** |  |  |  |  |  |
| **Task (f)** |  |  |  |  |  |
| **Task (g)** |  |  |  |  |  |
| **Task (h)** |  |  |  |  |  |
| **Group Report** |  |  |  |  |  |
| **Video** |  |  |  |  |  |
| **Individual Conclusions** |  | X | X | X | X |