Introduction to Artificial Intelligence

Summer 2025

Project: Predicting ECG Diagnosis

Version: 1
Teacher: Muhammad Farhan, muhammad_farhan.safdar.dokt@pw.edu.pl

1. Project details

Electrocardiogram (ECG) signals are non-invasive, preliminary and cheap diagnostic test that presents the functionality of heart muscle movements and can be used for various heart diagnostics. The aim of this project is to predict the heart diagnosis from ECG signals on CPSC-2018 dataset. The link to the dataset is given at here. Students may choose binary or multiclassification task, however, multiclassification is encouraging and will be marked with plus point.

The goals of this project are:

- To learn the steps, involve in pre- and post-processing of time series data
- To learn about various feature extraction techniques
- Learning the management of high dimensional data with array handling
- Getting comprehensive knowledge about time-frequency analysis
- To learn time series data and its classification using machine learning or deep learning

Project hints:

- Explore WFDB library [1] and convert data into NumPy arrays i.e., multi-dimension
- Consider denoising using libraries i.e., wavelet [2] family
- Explore the time-frequency spectral analysis using libraries including SciPy spectrogram [3], Mel-spectrogram [4]
- You can either use direct time-series data or it's conversion to 2-dimentional spectrogram images for model training
- In either case, hyper tuning parameters (at least 5-6 cases) per classifier should be maintain for the project efforts assessment
- Consider at least three or more classifiers in classification and compare the results. Student may use Scikit learn [5], PyTorch, Keras framework. However, compare the performance with Accuracy, F1 score, ROC curve, Sensitivity/Specificity

2. References

1. Waveform Database (WFDB), https://www.physionet.org/content/wfdb/10.7.0/
2. Wavelet Transformation, https://pywavelets.readthedocs.io/en/latest/regression/index.html
3. Scipy Spectrogram, https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.spectrogram.html
4. Mel-spectrogram, https://librosa.org/doc/main/generated/librosa.feature.melspectrogram.html
5. Documentation and user guide from Scikit-Learn.

3. Project Phases and Assessment

For each phase, there will be a report to submit followed by a discussion with the instructor. You may receive up to **25 points** for this project, assessed by the following criteria. Note the expected date (deadline) to complete each phase (earlier submissions are welcome).

Phase 1: Preliminary documentation (**0-5 points**)

Assessment	Deadline	Points
Preliminary report which contains: 1. Description of the dataset based on your observation. Provide descriptive statistics, plot some samples, report interesting patterns/findings. 2. Overview of your plan to tackle this problem. This includes: a. How to split the dataset into training and validation set b. What algorithms will be used, provide a brief description. c. What are the main tools/framework/libraries used for implementation. d. Proposed evaluation methods. How to measure and compare the performance of your algorithm. e. You can take inspiration from charts in this notebook	**09 May**	2 2
Preliminary discussion 3. Discuss with the instructor regarding your proposed solution. Obtain feedbacks.	**12 May**	1

Phase 2: Midterm solution (**0-5 points**)

Assessment	Deadline	Points
Submit the following files in GitHub: 1. Python code for the current state of implementation. 2. Updated report based on your findings. Including: a. Description of your implementation b. Preliminary results from the algorithm based on proposed performance metrics. c. Experiment analysis. Based on results from your experiments so far, explain what worked and what didn't (what tweaks you made and its effect).	**26 May**	 2 2
Mid discussion 3. Discuss regarding: a. Any changes from the initial plan b. Code demonstration c. Challenges or issues encountered so far	**27 May**	 1

Phase 3: Final solution (**0-15 points**)

Assessment	Deadline	Points
Project submission. Submit the following files in GitHub: 1. Final Python code for implementation.	**09 June**	 5
2. Final report. Please see final report guidelines below.		5
Final assessment 3. Code demonstration: a. Working training code (run for few epochs if it takes too long) b. Evaluation code for a trained model 4. Analysis of the results 5. Conclusions: what you've learned from this project	**13 June**	 2 2 1

4. Handing in guidelines

1. Use the GitHub platform to commit your progress and share the repository link instead of sending files directly (this includes the report). Don't forget to write clear instructions on how to use the code.
2. All communications (messages, oral discussion) will be on MS Teams.
3. Please be aware of the deadlines for each project phase.
4. Final project submission (committing the final code and report on GitHub) is on **June 15th**. Late project submissions for this final phase will result in **20% points decrease** per week overdue (up to 2 weeks).

5. Final report guidelines

The following is a suggested template for the final report. However, students may add chapters and sub chapters as needed.

1. Problem definition
2. Dataset
    a. Overview (describe the dataset)
    b. Pre-processing (what is done to the data before feeding them to the model)
    c. Post-processing (mention if used, any methods applied to output prediction)
3. Technical Approach:
    a. Architecture (what kind of model is used, describe briefly)
    b. Training details (describe the hardware and software used for this project. Describe the hyperparameters used for training, e.g.: loss function, learning rate, batch size, etc.)
    c. Evaluation details (what performance metrics are used, e.g.: accuracy, precision, recall)
4. Results
5. Conclusion
6. References
    a. If you rely on algorithms obtained from works of other people, please cite the author and their work (paper, git repo, blog).
    b. Please try your best to cite quotes, facts, images used in your report that you did not create yourself. This will make your work more credible.

6. Parting wisdom

As Francois Chollet (creator of Keras) says, machine learning is an iterative process. Results might not show immediately, so be patient and test out your ideas by making minor adjustments based on your results. Software bugs are real, so check your processing pipeline if there's any unintended variable changes or incorrect use of functions. Finally, and most importantly, don't forget to **have fun while learning**.