ECE1512 – Project A: Visual Interpretation of Convolutional Neural Networks

Assigned: Tuesday, February 1, 5:00 PM EST.
Due: Tuesday, March 1, 5:00 PM EST.

Background

Machine learning (ML) has made a great breakthrough in various computer vision tasks in recent years. However, the solutions based on machine learning and deep learning (DL) mostly act like "black-boxes." Though promising, there is little insight into how and why they make the decisions they make. The lack of interpretability limits the utility of the solutions, especially in highly regulated areas such as medicine, finance, and law. It is unclear why consumers/users of the technology should trust these complicated solutions and accept the results and recommendations. Explainable Artificial Intelligence (XAI) attempts to address such issues by generating human-understandable explanations of the causes laying behind the ML-based decision-making models and "open" the black box of these cumbersome models.

There are two types of XAI. The first type are explanations that help to understand the data better. The second type of explanations are the ones that help to understand the model better. We would like to work on the second type. Models can be inherently *explainable*, such as linear regression, naive Bayes models, or decision trees. However, more complicated models, such as the convolutional neural networks (CNNs) that are commonly used for image classification, tend to be too complex to be understood by humans and require an explanation.

Introduction

In this project, you will be exploring the problem of "Visual Explainable AI," which is a type of XAI for visual data such as images or videos. This field aims to investigate local explanations for the models trained for array/image classification purposes in the evaluation phase. Through visualizing the input features that are the most responsible in the model's decision making, Visual Explainable AI can help the end-users to verify the trustworthiness of the predictions given by these models, diagnose their cases of misclassification, and reach an understanding of the evidence leading the model to make its decision.

Systematically looking at the solutions in this field, they take a trained machine learning-based model and a test array/image as input. The output of these solutions is a heat-map named "explanation map" that highlights each given input region, based on their importance for the model's decision-making procedure. Mainly, visual explanation methods, a.k.a. "*attribution methods*," can be categorized into the following groups:

a) **Approximation-based methods:** The methods that approximate the importance of the input features in the target model's decision-making procedure (e.g., Local Interpretable Model-Agnostic Explanations (LIME) [1], SHAP [2], etc.).

b) **Backpropagation-based methods:** The methods that operate by backpropagating the signals from the output of the target model (e.g., Integrated Gradient [3], FullGrad [4], Layer-wise Relevance Propagation (LRP) [5], etc.).

c) **Perturbation-based methods:** The methods that probe the behavior of the target model by feeding it with perturbed (masked) copies of the input (e.g., Randomized Input Sampling for Explanation (RISE) [6], Semantic Input Sampling for Explanation (SISE) [7], Extremal Perturbation [8], etc.).

d) **CAM-based methods:** The frameworks that visualize the perspective of the target model by performing combination of the abstract features extracted by the model. (These methods are specialized for explaining CNNs, and based on Class Activation Mapping (CAM) method [9], (e.g., Grad-CAM [10], Grad-CAM++ [11], Ablation-CAM [12], etc.)

e) **Counterfactual-based methods:** The methods that produce a perturbation sample to change the model's original decision. The generated samples can act as a recommendation for end-users to achieve their desired outputs. Most of the current counterfactual explanation approaches (e.g., DeDUCE [13], StylEX [14], Counterfactual Visual Explanation [15], etc.) are the gradient-based method, which can only optimize the differentiable loss functions with continuous variables.

Objective

This project endeavors to generate explanation maps that interpret the behavior of two models trained on different datasets. The first model is a very shallow CNN trained on "**MNIST-1D**" [16], a small-scale dataset prepared for generic array classification. The second model is a VGG-7 network trained on the "**HMT**" [17] dataset utilized for histopathologic tissue classification. The detailed descriptions of the datasets are included in the next section. To begin with this project, you must create a GitHub repository for Project A as follows:

a. Create a group (individual) project personal GitHub repository. Use a proper naming convention **ECE1512_2022W_ProjectRepo_NameStudent(s)**.

b. Create a Project A sub-directory.

c. Populate your project A sub-directories with project related material.

Datasets

The following is a detailed description of two datasets that you will need for Project A. **Both datasets can be found in the Project_A_Supp.zip file which has been uploaded to Quercus.**

MNIST-1D:

- **Paper**: Scaling down deep learning [16]: https://arxiv.org/abs/2011.14439
- **GitHub link:** https://github.com/greydanus/mnist1d
- **Description**: This dataset is a 1-Dimensional and low-memory analogue of the popular digit classification dataset, MNIST. In the same way as the MNIST dataset, the MNIST-1D data are divided into **10 classes**, each of which represents a digit between 0-9**.** Unlike MNIST, each example in MNIST-1D train/test data is a one-dimensional sequence of points generated by augmenting a 1-D template representing each of the digits by random padding, random translation, adding Gaussian noise, adding a constant linear signal analogous to shear in 2D images, and lastly, downsampling to 40 data points.
- **Availability:** Publicly available (for academic purposes).
- **Resources needed:** CPU
- **Data size**: 4000 train data + 1000 test data (partitioned by the dataset promoters).

HMT:

- **Paper**: Multi-class texture analysis in colorectal cancer histology [17]: https://www.nature.com/articles/srep27988
- **Description**: This dataset was formed to elevate the performance of ML-based solutions in "histopathological tissue classification." HMT is an equally balanced dataset that contains images extracted from 10 independent samples of colorectal cancer (CRC) primary tumors and divided into one of the following 8 classes: (a) tumor epithelium, (b) simple stroma, (c) complex stroma, (d) immune cell conglomerates, (e) debris and mucus, (f) mucosal glands, (g) adipose tissue, (h) background.
- **Availability:** Publicly available (for academic purposes).
- **Resources needed:** CPU
- **Data size**: 4504 train images + 496 test images.

Experimental Setup – What you need

1. Prerequisites:
   1.1. Python 3
   1.2. TensorFlow framework
   1.3. Sci-kit-learn (suggested)
   1.4. NumPy (suggested)
   1.5. Matplotlib (suggested)

2. Download **Project_A_Supp.zip** from Quercus.
3. **MNIST1D.pkl** is a "pickle file" containing all data in the MNIST-1D dataset. The data in this file is save as a dictionary that can be also created using the function **make_dataset()** in the library **mnist1d_utils.py**. More information to read the dictionary is provided in the notebook MNIST1D.ipynb.
4. **hmt_dataset** is a folder containing two subfolders, including the train and test set for the HMT dataset.
5. **xai_utils.py** is a Python file including utility functions needed for three state-of-the-art solutions in the field of visual XAI, Grad-CAM [10], RISE (Randomized Input Sampling for Explanation) [6], and SISE (Semantic Input Sampling for Explanation) [7].
6. **HMT.ipynb** and **MNIST1D.ipynb** are two Python notebooks showing 1) how the models are trained with each of the two described datasets, and 2) how the explanation algorithms included in **xai_utils.py** are applied on each of the described models.

GPU Requirements

Note that you **do not need a GPU to successfully complete Project A. You should be able to train your models on your local (personal) machines**. However, if you so wish you can use a GPU by accessing Google Colab. Following are the steps to enable a GPU using Colab-

1. Upload the code base to Colab using your Google Drive.
2. Navigate to Runtime -> Change runtime type in the top bar.
3. Change runtime accelerator to GPU and click Save.
4. Use device = torch.device("cuda:0") after importing libraries and call .to(device) function to transfer your model and tensors to GPU. Make sure that model and tensor both are placed on GPU.

Part 1: 1-D Digit Classification

Task 1: 1-Dimensional digit classification [5 Marks]

1. Figure 1 depicts the backbone 1-D CNN model on the MNIST-1D dataset. Load the MNIST1D.ipynb file and use the TensorFlow framework to implement Figure 1 in the pre-assigned **"Model Creation"** section of MNIST1D.ipynb. The value of weight_decay in each layer should be 5e-4. [1 Marks]

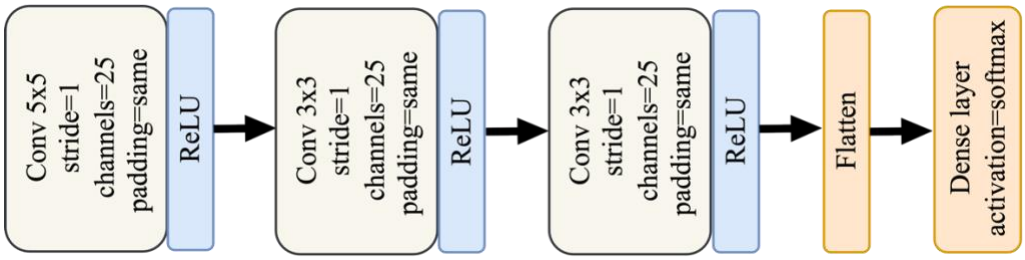

Figure 1: The proposed 1-D CNN model architecture on MNIST-1D

2. Train the model for 200 epochs with cross-entropy loss and SGD optimizer in the pre-assigned **"Training the Model"** section of MNIST1D.ipynb file. [0.5 Marks]

3. Upon completion of training, evaluate the performance of the trained model on the test data, assuming that the most-confident class is predicted for each image. Report the following: [2.25 Marks]
   a. Plot for loss curve and for accuracy curve. [0.5 Marks]
   b. Overall classification accuracy on the test set. [0.25 Marks]
   c. Class-wise classification accuracy for all classes. [0.25 Marks]
   d. Plot the classification ROC and AUC curves for each class. [0.5 Marks]
   e. Plot the normalized confusion matrix. [0.25 Marks]
   f. Precision, Recall, and F-1 score on the test set. [0.5 Marks]

   Note: You need to report code snippets, figures, and numerical values as part of your notebook. (More details regarding these metrics are available in the resources part)

4. Show some examples of the success/failure cases of the model. Among which two classes misclassification happens the most? Provide your insights and support your answers with analytic reasons. [1.25 Marks]

Task 2: CNN interpretation [10 Marks]

1. Select one of the attribution methods provided in xai_utils.py and one method other than those for which codes are provided (two methods, if you do the project in a group of two). Read the paper of the selecting methods and answer the following questions for each of them. [5 Marks]

    a. What knowledge gap did your two chosen attribution methods fill? [0.5 Marks] What novelty did they contribute compared to their prior methods? [0.5 Marks]

    b. Explain in full detail the methodologies of your selected methods. [2 Marks]

    c. Discuss the main advantages and disadvantages of your selected methods. Do you think these methods can concretely interpret the target model in difficult scenarios (e.g., when the target model is a deep CNN or the input contains a high amount of texture or noise)? Do you think your selected methods can analyze and inspect the cases of misclassification by the target model? Why? [2 Marks]

    Note: You can find a shortlist containing the references for some state-of-the-art attribution methods in pages 9, 10, and 11. Furthermore, considering the categorization provided in the introduction paragraph, you should not select two methods from same category.

2. Apply your selected method(s) on the CNN trained on the MNIST 1-D dataset, taking different inputs. [5 Marks]

    a. For each given input, your output should be a 1-dimensional explanation map that scores the input features based on their contribution to the model's prediction. [3 Marks]

    b. Qualitatively report the explanation maps you achieved and compare them with the templates presented for each of the digits. Discuss your results. Do you think the highlighted region is similar to the template corresponding to the digit predicted by your model? Do you think the explanation map shows the local behavior of the model well? [2 Marks]

Part 2: Histopathological Tissue Classification

Task 3: Biomedical image classification and interpretation [5 Marks]

1. Load the CNN trained on **models/HMT.h5** (VGG-7 architecture), and the test set **hmt_dataset/HMT_test**. Evaluate the performance of the trained model on the test data, using the same metrics that are mentioned in Task1 except part 3.a. [2.5 Marks]
2. Repeat the subsection 2.a of the Task 2 on the HMT dataset, using all the XAI methods you previously selected. [2.5 Marks]

3. (Optional) Use the commercial tool **"CAPTUM"** to explain the behaviour of the model which is trained on HMT dataset.

Task 4: Quantitative evaluation of the attribution methods [10 Mark]

Different from so-called "ground truth-based" evaluation metrics such as mean Intersection Over Union (mIoU) that compare the output of such algorithms with ground- truth masks, the concreteness and faithfulness of the attribution methods should be mainly assessed by another group of metrics named as "model truth-based," that verify the correctness of the explanations provided by an attribution method with the model's behavior. Model truth-based metrics (e.g., Insertion/Deletion, Drop/Increase) measure the relationship between the explanation maps generated by attribution methods and the target model's outputs.

In this task, you will evaluate the performance of your selected method, using a pair of model truth-based metrics, "Drop%" and "Increase%". This pair of metrics measure the decrease/incensement in the target model's confidence score when the target model is fed only with the most highlighted features by an explanation method, compared to when the whole input is given to the image. The intuition for these metrics is as follows:

- **Drop rate:** If we remove unimportant features from the input, the model's confidence score should not drop considerably.
- **Increase rate:** If we remove misleading features from the input, the model's confidence score may increase.

According to these metrics, low drop% and high increase% denote that the features assigned with a high score by the evaluated explanation method are highly considered in the model's decision-making procedure. More details regarding these metrics can be found in the resources, the last subsection in the notebooks **HMT.ipynb** and **MNIST1D.ipynb**, and Figure 2.

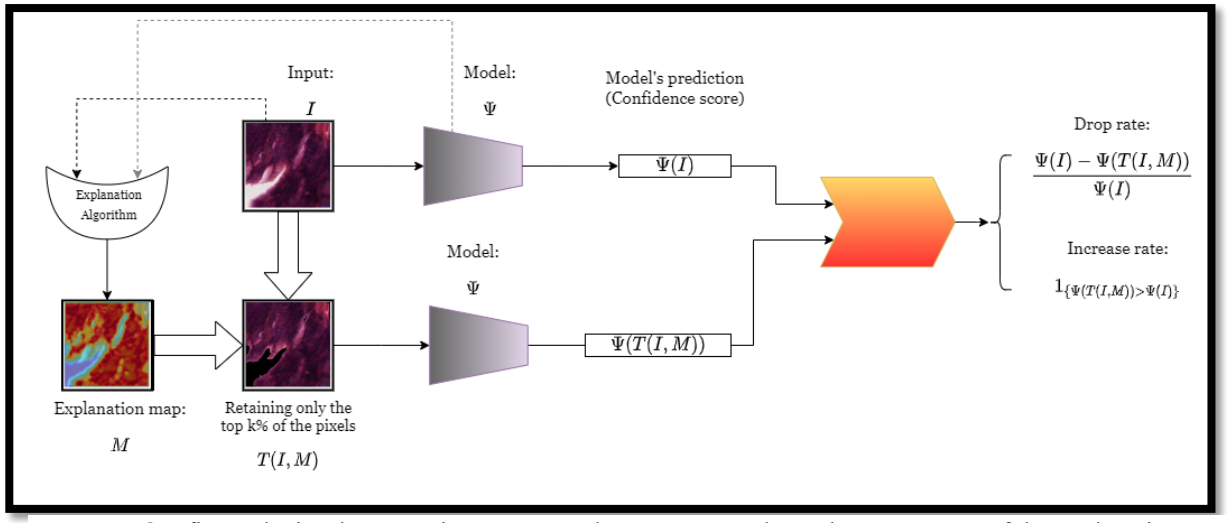

Figure 2: This figure depicts how metrics "Drop" and "Increase" evaluate the correctness of the explanations generated by an explanation algorithm.

1. Apply the "Drop%" and "Increase%" metrics to evaluate the performance of your selected attribution methods when applied to the CNN trained on both datasets. For the MNIST-1D dataset, take the parameter "k" shown in Figure 2 as 30%. This parameter can be tuned using the input argument **frac** in the function **calculate_drop_increase()**. Take this parameter for the HMT dataset as 90%. For both datasets, calculate the average drop% and average increase% on the whole test set. [5 Marks]

2. Discuss in full details the qualitative/quantitative results you have achieved. Were your selected methods successful in interpreting the target model trained on the HMT and MNIST-1D dataset correctly? In what cases they fail to explain the target model's predictions? If you select two attribution methods, in what cases each one of them works better than the other one? Support all your answers with detailed reasons. [5 Marks]

Notes

1. Coding is expected for this assignment, and your code must be included in your submitted report – use the Python programming language.
2. External code may be used only if properly cited and if it were initially introduced as part of non-XAI research.
3. Include as many data visualization results as possible – a good visual is worth more than a thousand words (as per your ECE1512 Lecture 1 handout – "One picture is worth more than ten thousand words" (anonymous))

Resources

Certain concepts and methods used in this assignment may be unfamiliar to you. Refer to these online resources for more details (cite if code is used):

- TensorFlow Installation: https://www.tensorflow.org/install/pip & https://keras.io/#installation
- Tutorial for TensorFlow V2: https://machinelearningmastery.com/tensorflow-tutorial-deep-learning-with-tf-keras/
- Training a CNN model in TensorFlow: https://www.tensorflow.org/tutorials/images/cnn & https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/
- Training and test sets: https://en.wikipedia.org/wiki/Training%2C_validation%2C_and_test_sets
- Evaluating a pre-trained CNN in TensorFlow: https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator & https://vijayabhaskar96.medium.com/tutorial-image-classification-with-keras-flow-from-directory-and-generators-95f75ebe5720

- ROC curves: https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/
- Confusion matrix: https://www.geeksforgeeks.org/confusion-matrix-machine-learning/
- Creation a GitHub repo: https://docs.github.com/en/get-started/quickstart/create-a-repo
- CAPTUM (Commercial tool): https://captum.ai/ & https://awesomeopensource.com/project/pytorch/captum

Background on Explainable AI:

- SHAP (Shapley Additive exPlanation): https://christophm.github.io/interpretable-ml-book/shap.html
- Explainer: https://explainer.ai/
- Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI https://www.sciencedirect.com/science/article/pii/S1566253519308103
- Explainable Artificial Intelligence: A Systematic Review https://arxiv.org/pdf/2006.00093.pdf
- The Mythos of Model Interpretability https://arxiv.org/pdf/1606.03490.pdf
- Trustworthy AI: A Computational Perspective: https://arxiv.org/abs/2107.06641
- EU High-Level Expert Group on Artificial Intelligence, Assessment List for Trustworthy Artificial Intelligence (ALTAI) for self-assessment: https://digital-strategy.ec.europa.eu/en/library/assessment-list-trustworthy-artificial-intelligence-altai-self-assessment
- Identifying Roles, Requirements and Responsibilities in Trustworthy AI Systems: https://arxiv.org/abs/2106.08258
- How to choose an Explainability Method? Towards a Methodical Implementation of XAI in Practice: https://arxiv.org/abs/2107.04427v1

Papers for visual explanation methods (you can browse for more papers):

[1] Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "" Why should i trust you?" Explaining the predictions of any classifier." In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135-1144. 2016. https://arxiv.org/abs/1602.04938

[2] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." In *Proceedings of the 31st international conference on neural information processing systems*, pp. 4768-4777. 2017. https://arxiv.org/abs/1705.07874

[3] Sundararajan, Mukund, Ankur Taly, and Qiqi Yan. "Axiomatic attribution for deep networks." In *International Conference on Machine Learning*, pp. 3319-3328. PMLR, 2017. https://arxiv.org/abs/1703.01365

[4] Srinivas, Suraj, and François Fleuret. "Full-gradient representation for neural network visualization." *arXiv preprint arXiv:1905.00780* (2019). https://arxiv.org/abs/1905.00780

[5] Binder, Alexander, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. "Layer-wise relevance propagation for neural networks with local renormalization layers." In *International Conference on Artificial Neural Networks*, pp. 63-71. Springer, Cham, 2016. https://arxiv.org/abs/1604.00825

[6] Petsiuk, Vitali, Abir Das, and Kate Saenko. "Rise: Randomized input sampling for explanation of black-box models." *arXiv preprint arXiv:1806.07421* (2018). https://arxiv.org/abs/1806.07421

[7] Sattarzadeh, Sam, Mahesh Sudhakar, Anthony Lem, Shervin Mehryar, Konstantinos N. Plataniotis, Jongseong Jang, Hyunwoo Kim, Yeonjeong Jeong, Sangmin Lee, and Kyunghoon Bae. "Explaining convolutional neural networks through attribution-based input sampling and block-wise feature aggregation." In *34th AAAI Conference on Artificial Intelligence*. 2021. https://arxiv.org/abs/2010.00672

[8] Fong, Ruth, Mandela Patrick, and Andrea Vedaldi. "Understanding deep networks via extremal perturbations and smooth masks." In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2950-2958. 2019. https://arxiv.org/abs/1910.08485

[9] Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. "Learning deep features for discriminative localization." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921-2929. 2016. https://arxiv.org/abs/1910.08485

[10] Selvaraju, Ramprasaath R., Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. "Grad-cam: Visual explanations from deep networks via gradient-based localization." In *Proceedings of the IEEE international conference on computer vision*, pp. 618-626. 2017. https://arxiv.org/abs/1610.02391

[11] Chattopadhay, Aditya, Anirban Sarkar, Prantik Howlader, and Vineeth N. Balasubramanian. "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks." In *2018 IEEE winter conference on applications of computer vision (WACV)*, pp. 839-847. IEEE, 2018. https://arxiv.org/abs/1710.11063

[12] Ramaswamy, Harish Guruprasad. "Ablation-cam: Visual explanations for deep convolutional network via gradient-free localization." In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 983-991. 2020. https://openaccess.thecvf.com/content_WACV_2020/html/Desai_Ablation-CAM_Visual_Explanations_for_Deep_Convolutional_Network_via_Gradient-free_Localization_WACV_2020_paper.html

[13] Höltgen, Benedikt, Lisa Schut, Jan M. Brauner, and Yarin Gal. "DeDUCE: Generating Counterfactual Explanations Efficiently." *arXiv preprint arXiv:2111.15639* (2021). https://arxiv.org/abs/2111.15639?context=cs.LG

[14] Lang, Oran, Yossi Gandelsman, Michal Yarom, Yoav Wald, Gal Elidan, Avinatan Hassidim, William T. Freeman et al. "Explaining in Style: Training a GAN to explain a classifier in StyleSpace." *arXiv preprint arXiv:2104.13369* (2021). https://arxiv.org/abs/2104.13369 & https://www.vedereai.com/introducing-stylex-a-new-approach-for-visual-explanation-of-classifiers/ & https://explaining-in-style.github.io/

[15] Goyal, Yash, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. "Counterfactual visual explanations." In *International Conference on Machine Learning*, pp. 2376-2384. PMLR, 2019. https://arxiv.org/abs/1904.07451

[16] Greydanus, Sam. "Scaling down deep learning." *arXiv preprint arXiv:2011.14439* (2020). https://arxiv.org/abs/2011.14439

[17] Kather, Jakob Nikolas, Cleo-Aron Weis, Francesco Bianconi, Susanne M. Melchers, Lothar R. Schad, Timo Gaiser, Alexander Marx, and Frank Gerrit Zöllner. "Multi-class texture analysis in colorectal cancer histology." *Scientific reports* 6, no. 1 (2016): 1-11. https://www.nature.com/articles/srep27988

What to submit & Evaluation [10 Marks]

1. **Project evaluation will be based on a submitted written report.** You need to submit a written report (on per team) **in PDF format using the ECE1512 Q page submission utility.** Your report should be approximately 20 pages in IEEE style. List any additional references you have used in your work. You need t provide the code, data, figures and any other auxiliary material) used in your work. To that end, your written report should include a link to your Project A GitHub repository.

2. **Project evaluation will include an examination of the code, notebooks, figures, and other auxiliary material posted on your GitHub repo.** It is highly recommended that you include a README document on your project's GitHub page.

3. **Ouriginal "plagiarism" scores** will be visible upon (report) submission, so make sure that you are not penalized for plagiarism in uncited text and code portions (if applicable).
4. **Late submissions will not be accepted.**

Submission guidelines - Front Page Matter:

Page 1. Cover Page. Typed:

- Project A title: **Visual Interpretation of Convolutional Neural Networks**
- Course number: **ECE1512 – 2022W**
- Student's name (or student names in a group of two students setting)
- Student ID (student IDs in a group of two students setting)
- Name/ID of submitting student (if applicable) – Note: Only one report per groups should be submitted
- Date due
- Date handed in