DO NOT INCLUDE THIS PAGE IN VERSION POSTED FOR STUDENTS

Facilitator notes and instructions:

- It is preferable to have two laptops per team – but only one is absolutely necessary
    - If there are two, at least one should have Matlab installed
    - The second laptop will need an internet connection (wifi is fine)

    - Frame the discussion of doing this PSS in terms of they are trying something new to help out the department. Data skills exercises are going to start appearing in other core BME courses.
    - The computational/AI questions of the PSS will not be graded.
    - Need to have backup outputs for each Matlab section in case groups have issues they can't work around
    -

- Students should be preconfigured with accounts to run software on the Partnership for Advanced Computing Environment (PACE) Instructional Cluster Environment (ICE). No action is required by students, the class itself is enrolled in ICE each semester via the instructors enrolling the class.

- Generally, students will do the following in order:
    - Log in to the PACE ICE cluster
    - Train a single neural network locally on a laptop during class (this will not complete)
    - As a team, manually diagnosis four ECGs as either normal or showing an arrythmia
    - Train a neural network on raw ECG signals using the ICE instructional cluster
    - Evaluate the accuracy of that neural network (it will be low)
    - Assess three options for using mathematics techniques to improve accuracy
    - Train a neural network on ECGs processed using one of those three options
    - Evaluate the accuracy of the second neural network
    - Compare and reflect on the different approaches

- In coordination with the PACE ICE leadership, the following should be in place
    - PACE ICE TA to help with code issues or access challenges
    - A reservation for an appropriate amount of hardware on the ICE cluster during the PSS
    - Students setup as users in ICE Users (file a ticket with PACEICE each semester)
    - Students need to have the GT VPN installed and configured
      you must log into the VPN to use ICE even if you are on campus

- For non-immediate assistance with ICE please contact
    - Todd Fernandez (BME)
    - ######### (ICE)

- Things to remind students of in advance:
    - Bring your laptop(s) – **with GT VPN installed**
    - Bring them fully charged if at all possible
    - They will be running Matlab code
    - They are not expected to WRITE any code
    - They MUST generally run the code in the order intended.
      The most common issue is failing to load the data causing model training to fail.

Problem-Solving Studio 3B – Cardiac Mechanisms of Action
BMED 3100
October 21, 2024

Group Number: _____
Names: _____

Objectives:

1. Use knowledge of ECG/EKG to identify events in Cardiac cycle.
2. Understand how clinicians perform disease diagnosis via ECG.
3. Evaluate accuracy of Machine learning models to diagnosis cardiac events via ECG
4. Understand how signals processing and concepts from calculus and mathematics modeling can improve the accuracy of algorithms for diagnosing ECG signals.
5. Understand pros and cons of large data and visual approaches to cardiac disease diagnosis

The following questions are based an open data challenge and example code from MathWorks

Data source: PhysioNet/Computing in Cardiology Challenge 2017
Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P. C., Mark, R., ... & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation [Online]. 101 (23), pp. e215–e220.
https://physionet.org/content/challenge-2017/1.0.0/

Code Basis: Classify ECG Signals Using Long Short-Term Memory Networks
Matlab Example Exercise (https://www.mathworks.com/help/signal/ug/classify-ecg-signals-using-long-short-term-memory-networks.html#ClassifyECGSignalsUsingLSTMNetworksExample-1)

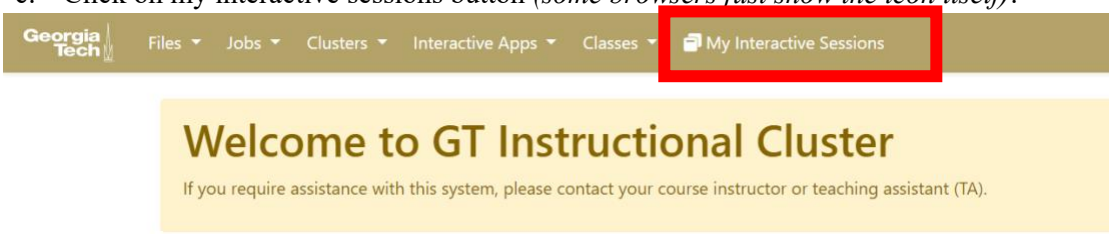
Data Explanation:
The data from the PhysioNet Challenge is single lead ElectroCardioGrams (ECGs). That means, a single electrical signal is recorded from each patient. The data consists of a set of ECG signals sampled at 300 Hz and are between 30 and 60 seconds in length. While the data contains many potential signals that can generate insights about ECG quality or cardiac behavior, each ECG was labeled by experts using on of four different classes: Normal (N), AFib (A), Other Rhythm (O), and Noisy Recording (~). The Matlab code simplifies the data by removing anything besides AFib (A) and Normal (N) signals for this exercise. The data set you will use contains 5788 of the 8528 ECGs in the larger challenge.

Atrial Fibrulation (AF) is a "tachyarrhythmia characterized by predominantly uncoordinated atrial activation with consequent deterioration of atrial mechanical function". AF is the most common cardiac arrhythmia, Despite the enormity of this problem, automated detection of AF remains problematic. Algorithms to detect AF generall take one of two approaches: atrial activity analysis-based or ventricular response analysis-based methods. Atrial activity analysis-based AF detectors analyze the absence of P waves or the presence of fibrillatory *f waves*. Atrial activity analysis-based AF detectors can achieve high accuracy if the recorded ECG signals are high quality, but are easily degraded by noisy data. In contrast, ventricular response analysis is based on the predictability of the inter-beat timing of the QRS complexes in the ECG. RR intervals are derived from the most obvious large amplitude feature in the ECG, the R-peak, the detection of which can be far more noise resistant. It is worth noting that AF detectors that combine both atrial activity and ventricular response could provide an enhanced performance by combining independent data from each part of the cardiac cycle.

Main instructions and Problems:

1) **Run the LOCAL Matlab code file:**
If you have two laptops – Use one of them to train a model without using the ICE cluster. Download the zip file from the link below. Then unzip the file and open *BMED3100_ECG_PSS_local.mlx*. Ensure it starts running. This will load the data and train a neural network to classify the ECG signals on your laptop. This is the same model you will train using the GT AI Makerspace
Local file link: http://tiny.cc/BMED3100LOCAL

2) **Log into ICE cluster (the AI Makerspace) on a separate laptop and start a session**
To do that you need to do the following:
a. Log into the GT VPN, see instructions here:
https://gatech.service-now.com/home?id=kb_article_view&sysparm_article=KB0042139
b. To log into ICE, go to this link:
https://ondemand-ice.pace.gatech.edu/pun/sys/dashboard
c. Click on my interactive sessions button *(some browsers just show the icon itself)*:



d. **Start a Matlab session on an H100 compute GPU**
- Click on **Matlab** on the left hand side **under interactive apps**
- **Change Quality of Service:** coe-ice
- **Select Node Type:** NVIDIA GPU H100 HGX
- **Change number of hours:** 3
- press **Launch** (leave everything else the same)

3) You will be taken to the *my interactive sessions* page while your session is setup. **When the launch button appears click it.**

4) **Load the Matlab code that you will run on the cluster** - Once you have Matlab running on the AI Makerspace, type the following two commands into the virtual session command window and press enter after each one:

First command: cd("/storage/ice-shared/bmed3100")
Second command: open BMED3100_ECG_PSS_ICE.mlx

5) From here on out, follow the instructions on the remaining pages - You will run specific sections of code and then answer questions on the worksheet or enter data into a shared google spreadsheet. It is broken into 4 parts – human diagnosis, train a model, improve the model, train a third model.

Links you will need:

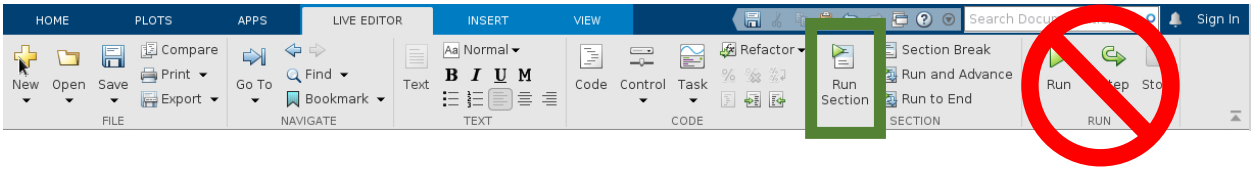
- The zip file containing the matlab code to run on your laptop: http://tiny.cc/BMED3100LOCAL
(full alternative: https://gtvault-my.sharepoint.com/:f:/g/personal/tfernandez32_gatech_edu/EuyQwGY3dhNGqf1p8OKvDOcBTZ6kgap-lMb7-LUnajl_mQ?e=oNksi9)

- The link to the GT Instructional Cluster Homepage: https://ondemand-ice.pace.gatech.edu/pun/sys/dashboard

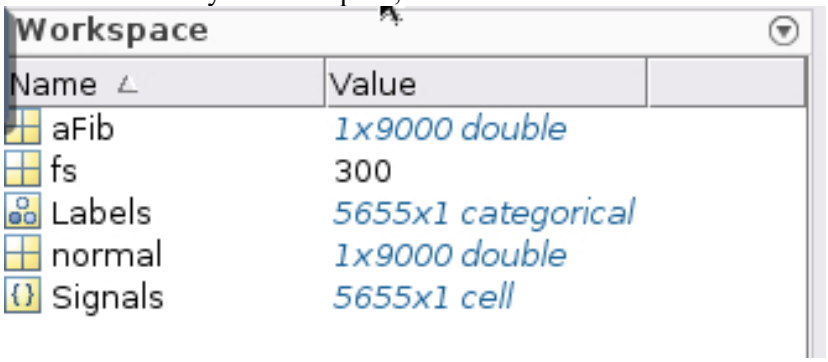
- The google spreadsheet to put your results in: http://tiny.cc/BMED31000SHEET
(full alternative: https://docs.google.com/spreadsheets/d/19mjW4iMQ2dn2-TBMAhg_6AZuTA1fgOkkk3QYA4oCEl0/edit?usp=sharing)

Part 1: Loading data and looking at ECGs yourself

Read the introduction to the code (briefly)…it includes the note to use the run section button, NOT the run button to run code. The code is divided into numbered sections (*Section #*).



- **Run Section 1 of the code** – this will load the raw ECG data into Matlab. You should end up with 5 total variables in your workspace, as shown below:



- **If your workspace DOES NOT show this** – please ensure you did step 5 on the previous page
  If you are still stuck, put your hand up for help.

- **Section 1 of the code outputs a summary of the data set telling you how many signals labeled as AFib and NOT AFib there are in the data. Writer those numbers down. We'll use it later**

- **Run Section 2 of the code -**this will output four (4) random ECGs as plots. As a team, inspect them and classify each one as either AFib or NOT AFib. Enter the signal number *(from the plot title, ECG#____)* and your diagnosis of each ECG into appropriate row (section and team number) in the google spreadsheet.

- **When you are done, as a team discuss the questions below.** As everyone finishes, you will discuss them as a class:
  - What in the ECG signals did you look for to make your diagnosis?
  - What made diagnosis challenging?
  - How confident are you in your diagnoses

Part 2: Training a neural network on the ECG Signals

In this part you will configure and train a relatively simple neural network on the raw ECG signal data. That involves running three sections (numbers 3, 4, and 5) of the code.

- **Run Section 3 of the code** – This section of code will address the 7:1 ratio of not AFib to AFib data and prepare it for use. It does so by replicating the AFib signals until there is an 1:1 ratio and randomly dividing them into three groups (Train, Test, Validate). When the code is done it will print the number of signals in each group. Look at those numbers – what do notice about test vs. training?

- **Run Section 4 of the code** – this section defines and then starts to train the first neural network. It should take between 10 and 15 minutes. When it is done it will print a summary of your newly trained neural network in the code window. While its running answer the three questions below:

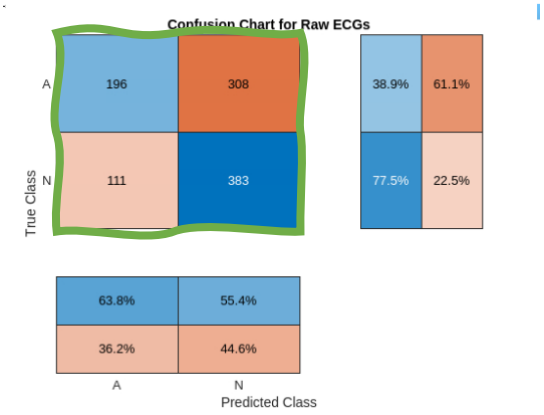
  1) Earlier you recorded the number of Not AFib and AFib signals in the original data set. Using those numbers, we want you to calculate a hypothetical accuracy if we just decided to be bad clinicians and classify all the ECGs as Not AFib. We could also do the opposite and classify every ECG as positive – calculate that hypothetical accuracy as well.
     Formula: *Accuracy = # correctly classified signals / # of total signals*

  2) How do those accuracies relate to the code in section 3 where we fixed the 7:1 ratio?

  3) In the graphic that pops up, called the training progress monitor, the validation accuracy (basically an internal test of the network's accuracy) and the training accuracy reflect the model's accuracy as it learns. What do you notice about those lines?

- **Report basic details the model** - When the model is done training done it will output some basic data. Please note the number of parameters and not how many iterations the neural network went through in training. Record those in the spreadsheet.

- **Run Section 5:** Section 5 uses the network to test the accuracy of your network. The results is an overall accuracy for this training network and a 'confusion chart'. This takes some time as it passes each ECG through the trained neural network. When done, record the accuracy and raw values (not percentages, see green box at right) from the confusion chart into the google spreadsheet.



- **When you are done, as a team discuss the questions below.** As everyone finishes, you will discuss them as a class:
  - What is the problem with having high test accuracy and low training accuracy?
  - Why might raw ECGs signal be challenging to fit a mathematical signal to?
  - What do you think would make this model better?

Part 3: Training an improved neural network on processed ECG signals

In this part you will improve the neural network that you trained above. Rather than changing the network we will use *feature extraction* to improve the data that we put into the model. That uses a set of signals processing techniques to increase the signal to noise ration. You will run three sections (numbers 6, 7, and 8) of the code in part 3 to train your second neural network.

- **Run Section 6 of the code** – This section of code will create three pairs of graphs showing the different signals processing techniques that will help to differentiate them better for the AI algorithm. For each graph, try and describe what makes the AFib and Not AFib signals different:
  Power Spectrum

  Instantaneous Frequency

  Spectral Entropy

- **Run Section 7 of the code** – This section performs the same feature extraction on the data, defines, and trains a second neural network. It takes 3-5 minutes, when it's done report the training time and the number of parameters in the spreadsheet. While its running answer the two questions below:
  1) **Do you think the feature extraction graphs would be easier/harder for a doctor to interpret?**

  2) **What about the electrical signals that the cardiac system creates might make it difficult to diagnose a condition by looking at the frequency domain rather than the time domain?**

- **Run section 8 and Report basic details the model** - When the model is done training done it will output some basic data. Please note the number of parameters and not how many iterations the neural network went through in training. Record those in the spreadsheet.

- **When you are done, as a team discuss the questions below.** As everyone finishes, you will discuss them as a class:
  - What about the signals processing do you think makes the accuracy of this model is higher?
  - Is this model sufficiently accurate for clinical applications?
  - Do you think you/a doctor should trust this model over than a model with 80% accuracy that is interpreting the raw ECG signals ?

Part 4: Picking how to improve your model

In the final section, your team has a choice. There are three options you can pursue to choose how to improve our model of ECG data. All the descriptions of the code are here. You should run ONLY ONE option – either section 9, 10, or 11 of the code, the descriptions are below. For sake of comparison, all three return to using the raw ECGs not the processed ones. Pick one then run that section of code.

- **Run section 9, 10, or 11 –** It will train one of three models described below

Name *(Option – Section)*	**Description**
Bigger network *Section 9 – Option 1*	This option goes back to the original raw data. This option trains a larger (i.e., 'more parameters') model on the same data. Research has found there can be major improvements from simply making the network large enough (e.g., ChatGPT and other LLMs). This option increases the size of the 'bidirectional LSTM' layer from 50 to 500. It also adds another 'hidden' layer (from 1 to 2) that can look at how those 500 parameters interact. In all, it creates a 2M parameters, a 100x increase.
Prioritize Afib data *Section 10 – Option 2*	Often, when doing diagnostics, we prioritize what is more important. In some cases (genetic screenings), false positives are worse than false negatives. In other cases (COVID), false negatives are much worse. Earlier we balanced the data so there was about equal Afib and Not Afib data for training. This model doubles the Afib data – and trains a model with 2 AFib signals, giving more weight in the model to AFib signals.
Look at more data at once *Section 11 – Option 3*	The first model used a 'mini batch' of 200 signals – meaning it only looked at 200 of about 8000 signals in each iteration. This batching helps manage memory space on the hardware doing the training. With each iteration, the model picks a new mini batch at random. However, this limits the data available to the model at any one time. This option doubles the amount of data that the model looks at in each iteration to 400.

- **Check on the 'local model'** – If you are running the local mode code on one of your laptops, check-in on it. How many iterations has it completed and how long has it run?

- **Diagnostic reasoning** is being able to identify key diagnostic indicators and connect them to the most appropriate diagnosis of a patient's signs (measurable) and symptoms (how patients experience symptoms-unmeasurable). What role do you think AI/ML has in diagnosing cardiac measures (ecg. ECGs, blood pressure, patient symptoms, medical records) now and in the future?

- **Overall reflection** – As models continue to train, as a group (then a class) answer these questions:
  - What happened with the improved models? Are they better or worse? Why?
  - What do you feel like you learned better about the cardiac system from this exercise?
  - What do you feel like you learned better about the Machine Learning from this exercise?
  - What is one thing your group is struggling with or confused by?

- **Enter results** – When, or if, your improved model finishes running, enter the results in the google spreadsheet. What do you notice?

Notes, questions and troubleshooting

- **A piece of the code (especially the scoring/test accuracy evaluation) didn't run.**
  Check your workspace and ensure all the input variables are loaded. The code is meant to be run in order. Meaning that it relies on some (most) of the prior sections. Especially the third group of models (where you have a choice) reuses significant portions of the setup (e.g., the model structure and data sets) from the first neural network model.

- **What's the difference between training accuracy, validation accuracy, and testing accuracy.**
  This article has a reasonably good explainer – we've modified it here to be clearer. In short, training data is used to train variables *in* the model itself (parameters). Validation data is used to assess the accuracy of the model while training, and make some adjustments to the *way* the model learns during training (hyperparameters). Test data is used AFTER the training is complete to evaluate whether the model is good at what it is supposed to do. A primary concern here is *overfitting* meaning that the model is very good at the training data, but it is fitted to that data in a way that makes it not useful for anything else.

Accuracy	Training	Validation	Test
Use	Training data is used by the model to adjust the parameters it uses to detect ECGs iteratively.	During training, validation data is used to evaluate model performance on and adjust the training process.	After training is complete, test data is used to assess the model's accuracy and on new to it data.
Dataset	Usually, the largest or second largest data set. More is typically better, bias in data here impacts model ability.	A subset of training data used to tune the training process itself to guard against overfitting.	Independent dataset not seen during training, used to evaluate final model performance.
About Overfitting	If accuracy on training data is higher than validation or test accuracy, it implies that the model is overfitting the data.	If validation accuracy lags significantly behind training accuracy, it suggests overfitting.	Similar to validation accuracy, a gap between test and training accuracy often indicates overfitting.

- **My code won't run after I started a new session, but the code is still in the same place.**
  Each time you start a new ICE or PACE cluster reservation, you get a new and blank workspace. It will leave the code in place but wipe away any saved variables or data. Therefore, you will need to retrain any models and reload any variables that you need.

- **Can I save this code to my desktop/personal computer and play with it more later**
  Of course – if you have questions Prof. Fernandez is probably your best bet. Please note Prof. Fernandez is not a professional software developer and no warranty on the quality of this code is implied or granted. In other words – its probably *really* bad code, use with caution..