Machine Learning in Physics — UCSD PHYS 139/239

Homework 3

Draft version due: Friday, February 17, 2023, 5:00pm

Final version due: Wednesday, February 22, 2023, 5:00pm

Policies

- Draft version due 5:00pm, Friday, February 17 on Gradescope (report and code).
- Final version due 5:00pm, Wednesday, February 22 on Gradescope (report and code).
- You are free to collaborate on all of the problems, subject to the collaboration policy stated in the syllabus.
- You should submit all code used in the homework. Please use Python 3 and sklearn version $\geq$0.18 for your code, and that you comment your code such that the TA can follow along and run it without any issues.

Submission Instructions

PLEASE NOTE that there are two steps to submitting your Homework. Both must be submitted by the deadline.

- Please submit your report as a single .pdf file to Gradescope under "Homework 3 Report Draft" or "Homework 3 Report Final". **In the report, include any images generated by your code along with your answers to the questions.** For instructions specifically pertaining to the Gradescope submission process, see `https://www.gradescope.com/get_started#student-submission`.
- Please submit your code as a .zip archive to Gradescope under "Homework 3 Code Draft" or "Homework 3 Code Final". The .zip file should contain your code files. Submit your code either as Jupyter notebook .ipynb files or .py files.

1 GitHub [20 Points]

Problem A [20 points]: Follow the instructions below. The purpose of this problem is to get you accustomed to Git commands and using GitHub.

- Create a GitHub account if you don't have one already: https://docs.github.com/en/get-started/signing-up-for-github/signing-up-for-a-new-github-account.
- Create an SSH key (either on your laptop or on DataHub, depending on where you will work with GitHub): https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent.
- Add the SSH key to your GitHub account: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account.
- Create a *public* GitHub repository called `<your_github_username>/hello-world`. See https://docs.github.com/en/get-started/quickstart/hello-world.
- Clone the GitHub repository locally on the command line:

```
git clone git@github.com:<your_github_username>/hello-world
```

- Create and check out a new "feature branch" called `readme_edits`

```
cd hello-world
git checkout -b readme_edits
```

- Make edits to the `README.md`, e.g. add a sentence like "Physics and machine learning are fun!"
- Stage and commit your changes with a helpful commit message

```
git add README.md
git commit -m "README update"
```

- Push your local changes to the remote repository

```
git push origin readme_edits
```

- Create a pull request by navigating to the webpage: https://github.com/<your_github_username>/hello-world/pull/new/readme_edits where you insert your GitHub username. The exact URL should be displayed on the command line.
- Check that the changes are what you expect them to be on the https://github.com/<your_github_username>/hello-world/pull/1/files tab and merge the pull request!
- Please provide your GitHub repository URL so that we can check you followed all the steps.

2 RNNs vs. CNNs for Time Series [40 points]

Relevant materials: lectures 9 and 10

This problem revisits the hands-on notebook `https://github.com/jmduarte/phys139_239/blob/main/notebooks/05_Time_Series_Data_RNN.ipynb`. For this problem, we will use the full 50k traces, by setting

```
n_train = 40000
n_test = 10000
```

We will also use a slightly modified learning rate schedule:

```
ExponentialDecay(initial_learning_rate=1e-3, decay_steps=100, decay_rate=0.8)
```

We *highly recommend* using the GPU-enabled DataHub with the latest `jmduarte/phys139_239:latest` image for this problem.

Problem A [15 points]: Replace the LSTM layers with bidirectional LSTM layers. Set `verbose=1` in the `model.fit()` command to be able to see the output during training, `batch_size=2048` to speed up the training, and `epochs=100` to train the model for (up to) 100 epochs. How many trainable parameters does the model have? How long does 1 epoch of training take (approximately)? What accuracy and AUC do you achieve for the classification task?

Problem B [15 points]: Now replace the LSTM layers with 1D convolutional layers with the hyperparameters indicated in the notebook. How many trainable parameters does the model have? How long does 1 epoch of training take (approximately)? What accuracy and AUC do you achieve for the classification task?

Problem C [10 points]: Purely from an accuracy/AUC perspective, which model performs better? Which model trains faster?