CSC 219-01 Artificial Intelligence (Spring 2023)

Project 2: Time Series Forecasting using LSTM and CNN

Due at 3:00 pm, Monday, October 16, 2023

Demo: class time, Monday, October 16, 2023

1. Problem Formulation

Time series forecasting is an important area of AI. It is important because there are so many prediction problems that involve a time component. This time component makes time series problems more difficult to handle. In this project, you practice with time series data to predict stock price.

This project is twofold. First, you apply LSTM and CNN models to your selected classification dataset. Second, you apply LSTM and CNN models to your selected regression dataset

Hint: for CNN models, treat each record of the time series dataset as a 1D or 2D image.

2. Dataset

Go to this link: **https://github.com/gzerveas/mvts_transformer**

The above link shows you one state-of-the-art technique for time series forecasting (based on the transformer architecture, which we will focus on later this semester). Skip the paper details for now. Simply go to the section about datasets used in the paper.

Get data from TS Archive

Download dataset files and place them in separate directories, one for regression and one for classification.

Classification: http://www.timeseriesclassification.com/Downloads/Archives/Multivariate2018_ts.zip

Regression: https://zenodo.org/record/3902651#.YB5P0OpOm3s

Choose one classification dataset and one regression dataset, respectively (you may choose any dataset you like from the repository). You can also find more details about each dataset here: http://tseregression.org/

3. Requirements

- Use training data to train your models and evaluate the model quality using test data
- Define window size and sequence the data appropriately.
- Do feature normalization. **Notice that you should NEVER normalize the output feature when training any regression models.** Otherwise, the RMSE of the regression model will be also normalized.
- Use EarlyStopping and ModelCheckpoint when training LSTM and CNN using Tensorflow.
- Tune the following hyperparameters manually to record how they affect performance in your report. Tabulate your findings. Show at least three different configurations for each model type. You may choose to do tuning manually or use KerasTuner
    - **Activation:** relu, tanh
    - **Number of layers**
    - **Neuron count in each layer**
    - **Optimizer:** adam and sgd
    - **Kernel number and kernel size** (for CNN only)
    - **Number of LSTM layers and neuron count in each layer** (for LSTM only)
- For the regression dataset, show RMSE and lift chart for your best LSTM and CNN models.
- For the classification dataset, show the precision, recall, F1-score of each label for your best LSTM and CNN models.

4. Grading Breakdown

You may feel this project is described with some certain degree of vagueness, which is left on purpose. In other words, **creativity is strongly encouraged**. Your grade for this project will be based on the soundness of your design, the novelty of your work, and the effort you put into the project.

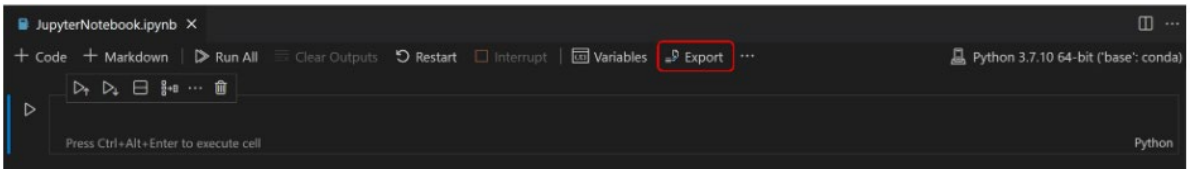
Use the evaluation form on Canvas as a checklist to make sure your work meets all the requirements.

5. Teaming

Students must work in teams of 2 people. Think clearly about who will do what on the project. Normally people in the same group will receive the same grade. However, the instructor reserves the right to assign different grades to team members depending on their contributions.

6. Deliverables

(1) The HTML version of your notebook that includes all your source code. In VS Code, you can export a Jupyter Notebook as an HTML file. To export, select the Export action on the main toolbar. You'll then be presented with a dropdown of file format options.



(2) Your report in PDF format, with your name, your id, course title, assignment id, and due date on the first page. As for length, I would expect a report with more than one page. Your report should include the following sections (but not limited to):

- Problem Statement
- Methodology
- Experimental Results and Analysis
- Task Division and Project Reflection
- Additional Features

In the section "Task Division and Project Reflection", describe the following:

- who is responsible for which part,
- challenges your group encountered and how you solved them
- and what you have learned from the project as a team.

In the section "Additional Features", you describe and claim credit for additional features.

To submit your notebook and report, go to Canvas "Assignments" and use "Project 2".

All the deliverables must be submitted **by team leader** on Canvas before

3:00 pm, Monday, October 16, 2023

NO late submissions will be accepted.

7. Possible Additional Features (5 pts per feature, 10 pts at most)

- For any classification/regression dataset, can you find the best window size (sequence length, i.e., how far we should look backward) that yields the best prediction results into future?

- Can you use LSTM to predict the values in a continuous future time period? Show the true values and predicted values in the same chart.

  Hint: train a regression model with multiple neurons in the output layer.

- Use WeightsandBiases for automatic hyperparameter tuning. WeightsandBiases provides free account for students. See examples here: https://docs.wandb.ai/guides/sweeps

- Can you build a mode using bidirectional LSTM, followed by Attention layers, to see if it can beat your baseline LSTM models?

  Here you can find the implementation for bidirectional LSTM layer:

  https://keras.io/api/layers/recurrent_layers/

  Read this paper for a comparison of RNN, LSTM and Attention:

  https://medium.com/swlh/a-simple-overview-of-rnn-lstm-and-attention-mechanism-9e844763d07b

  Here you can find two APIs to quickly add Attention mechanism on top of your LSTM model:

  https://github.com/philipperemy/keras-attention-mechanism

  https://github.com/CyberZHG/keras-self-attention

- Can you build a transformer model (an extension of RNN) for this problem? How it will compare with your baseline LSTM models?

  A nice introduction of Transformer: https://github.com/microsoft/AI-For-Beginners/blob/main/lessons/5-NLP/18-Transformers/TransformersTF.ipynb

  Code example: https://keras.io/examples/timeseries/timeseries_transformer_classification/

- Simply apply the MVTS model (PyTorch) and compare it with your baseline LSTM model.

8. In-class Presentation.

On the due day, each team has 5 minutes to present your work in the class. Explain your solutions by referring to your notebook. You do not have to prepare the PowerPoint slides for your presentation.