Stock Market Trading with LSTM Analysis

Franklin Doane
dept. of Computer Science
Tennessee Technological University
Cookeville, United States
fgdoane42@tntech.edu

I. Dataset Creation

A. Find stocks by market cap

I downloaded a list of all stocks ranked by market cap off the internet

B. Select largest stocks with sufficient data

Next, I used the list along with the yahoo finance api to select the 1500 most valuable companies that satisfied the following conditions: data for training dating back to the beginning of 2011, and additional data before that sufficient for indicator calculation and for creating the first time-series input with Jan 01 2011 as it's last date.

C. Download train and test data

I used 10 training datasets, the first one containing data for all market days in 2011-2012, with each following adding 1 sequential year of data. There are 10 evaluation sets, each contain data for a single year, from 2014 - 2023. Each training set and testing set downloaded data for every stock in the list using the yahoo finance api. All data is made up of 1 day candlesticks. Each download with the size of the dataset, with additional older data sufficient for calculating indicators and for supplying the 50 day history for the first day of the year.

II. Training Setup

A. Load dataset

Dataset is loaded from csv files.

B. Calculate additional features

The only additional feature calculated on the data is the day of the week 1-hot-encoded into 5 separate features.

C. Slice into training examples

The data then gets cut into time series, each with a length of 50. The data from the day following each time series is used to create the label, which consist of the relative increase in high price and low price compared to the close price of the previous day. The training examples from each stock get concatenated together into 1 dataset.

D. Seperate validation set

Once all the training examples have been created, 15% of the training examples are sliced off of the back of the dataset to become the validation set. The rest of the examples is the training set.

E. Shuffle training data

The ordering of the training set gets shuffled.

F. Split batches

The training data gets split into whole batches of size 200.

G. Initialize model

The model is initialized with random parameters. The model is an LSTM with 1 hidden layer of size 64. The output of the LSTM is fed into the following sequence:

$$Fully\_Connected(64 \rightarrow 32) \tag{1}$$

$$Leaky\_ReLU(leak = 1e^{-5}) \tag{2}$$

$$Fully\_Connected(32 \rightarrow 2) \tag{3}$$

III. Training Loop (per epoch)

A. Calculate loss

The loss for each batch is calculated with the following formula. The formula utilizes Mean Squared Error loss and a magnitude penalty for the logits.

$$Loss = mse(Y, \hat{Y}) + \lambda \sum_{i=1}^{B} (\hat{Y}_{i,1}) - \lambda \sum_{i=1}^{B} (\hat{Y}_{i,2}) \tag{4}$$

Here the first logit is the high price that the model is predicting, and the second is the low price. B represents the batch size, and λ is the magnitude penalty coefficient.

B. Optimize

The loss is backpropagated for each batch, and the parameters are updated with the Adam optimizer. The learning rate for the optimizer starts at 1e-3, and is set by the PyTorch LowerOnPlateau scheduler. The metric tracked by the scheduler is validation loss.

C. Validation

After all the batches are used for training, validation occurs. The model predicts all of the examples in the validation set and loss is calculated. Unlike during training, the loss in validation is calculated as a pure Mean Squared Error loss with no magnitude penalties. The loss is then used to update the learning rate scheduler.