Title: Visualization and Prediction for a Limit Order Book using CNNs and RNNs
Author: Alan Dunetz
Category: Time Series, CNN, RNN
Code and Notebooks: www.github.com/dunetz/CS230Final

Abstract:
Four Deep Learning architectures were applied to a mid-price change prediction problem for a high frequency Limit Order Book (LOB) dataset: 1) A Wavenet-type architecture [12] employing dilated 1D CNNs with causal padding significantly outperformed other architectures; 2) The inclusion of hand-crafted features in the dataset resulted in an improvement over approaches that relied the LOB price/volume data alone; 3) Prediction performance for a single stock benefited from the inclusion of LOB data for other stocks; and 4) A variety of visualization techniques were employed to provide insight into what our networks learned.

1. Background
LOBs aggregate all orders on an exchange to buy and sell a security at different prices. The LOB therefore provides a snapshot of the market's cumulative demand to buy and sell securities. Imbalances between bid and offer order sizes, or sudden changes in order sizes or prices, for example, may be informative about future price direction. The prediction problem is difficult because stock price data is characterized by low signal to noise ratios. Traders constantly observe market patterns and seek to arbitrage and game any apparent signal. The problem is interesting to high-frequency traders seeking to avoid adverse selection. It is also interesting to regulators seeking to identify illegal trading activity, such as "spoofing", where orders are placed for the purpose of misleading investors.

2. Dataset
We used a publicly available high frequency dataset (FI2010) which is described in [5] below. The dataset consists of 10-days of limit order book data from June 2010 for five stocks that trade on the Helsinki exchange. Each record in the time series shows prices and aggregate order sizes for the first ten levels on each side (bid and offer) of the market (forty points of data in all). The total number of messages (i.e. arriving buy/sell orders or cancels) reflected in the time series is approximately four million. The dataset includes an orderbook snapshot after every 10 messages resulting in approximately 400,000 records in total for the five stocks. On average, there is one snapshot for approximately every one-half second, though timing varies based on the level of market activity.

A number of versions of the dataset are available using different normalization schemes. We used a normalization scheme where each series is normalized based on the mean and standard deviation of the the same series on the prior day.

In addition, each snapshot includes approximately 100 handcrafted signals based on the methodology described in [1]. Each record also includes five labels categorizing the direction of price changes over different horizons: 1, 2,3,5 and 10 lob snapshots. A smoothing process was applied to the labels to remove some of the noise from the prediction process.

3. Methodology
We used the first six days of data for training and days seven and eight for validation. Once the parameters of our models were finalized we refit them on days one through eight and

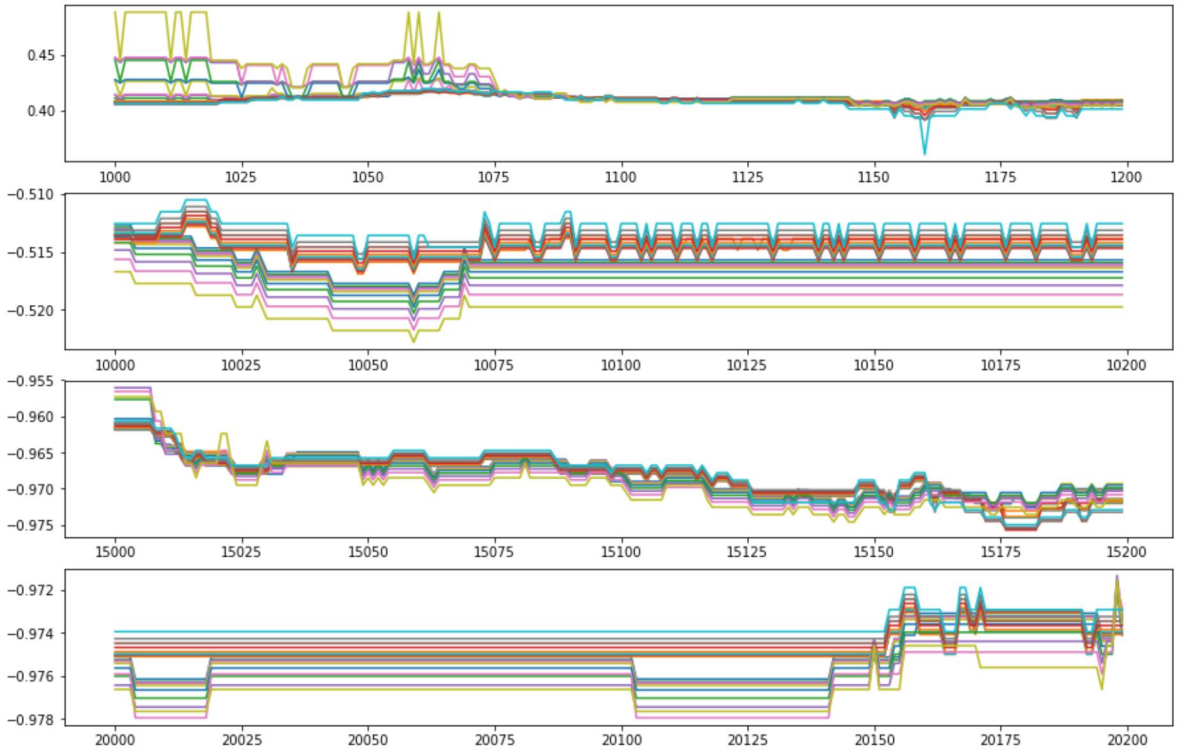

Figure 1: Sample LOB price movements

used days nine and ten for out-of-sample testing. Each window of data consisted of a sequence of LOB snapshots. The length of the window, itself, was a hyper-parameter. The target label was the ten period price change direction from the last snapshot in each window. Category labels were moderately unbalanced with 38.4% up; 24.7% unchanged; and 36.9% down. We used data for all stocks for training and validation except for the Transfer Learning experiment described in Section 8. Each window rolled forward one LOB snapshot past the prior window. However, we took care to ensure that windows and batches did not straddle multiple stocks or multiple days.

Our loss function was categorical cross-entropy. Our loss metrics were loss, accuracy, F1 and Cohen's Kappa. Kappa is a measure of agreement which takes into account the possibility of agreement occurring by chance.

4. Model Summary

LSTM	100 LSTM Units -> Dropout -> Softmax
CNN	3 Conv layers (16 filters) -> MP -> Conv Layers (32 filters) -> MP -> 100 unit FC->Softmax
CNN-LSTM	Same as CNN model with 100 unit LSTM layer replacing FC layer
Dilated-CNN	5 dilated Conv layers with causal padding-> 100 unit FC -> Dropout ->Softmax

Table 1 - See Github notebooks for additional detail.

5. Hyper-parameter Search

Key parameters tuned included learning rate, batch size, sequence length, regularization methods, layer size and number of layers.

We found that CNNs required low learning rates (<=.001) and small batch sizes (<=50). More than 4 conv layers or 2 max pool layers degraded performance. Optimal window length was approximately 100 LOB snapshots. Smaller filter sizes (2-4) worked better than larger ones. The first CNN layer spans all 40 features (4x40), essentially making the model equivalent to a 1D CNN, i.e, first layer filters were separate for each feature.

The LSTM model tended to overfit despite the use of L2 regularization and dropout. Only a single LSTM layer could be used without overfitting. Larger batch sizes degraded performance modestly but gave a major boost to training time. Increasing window size to 20 improved performance. Optimal layer size was probably smaller than the 100 units we actually used.

We used ReLu activation, the Adam optimization algorithm, and default Keras initializations.

6. Performance

Classification performance for the four models is summarized in the Table 3 below. All models

	Loss	Accuracy	F1	Kappa
LSTM	0.71|0.90	0.71|0.64	0.67|0.64	0.50|0.46
CNN	0.88|0.88	0.60|0.62	0.62|0.62	0.41|0.42
CNN-LSTM	0.72|0.81	0.69|0.64	0.67|0.64	0.50|0.46
Dilated-CNN	0.42|0.34	0.85|0.89	0.89|0.89	0.83|0.84

Table 3: Lefthand numbers are prediction performance on training data (Days 1-8); Right hand numbers are performance on test data (Days 9-10).

were trained using 100 epochs. The Dilated CNN model vastly outperformed the other models. However, these results, frankly, appear to good to be true, and merit further study/scrutiny. We were not, for example, able to recreate the model performance for the Dilated CNN model for the 5 period target horizon. Among the other models, the addition of an LSTM layer improved the CNN model. The CNN-LSTM models had less issues with overfitting than the LSTM model and would probably outperform the LSTM model given a longer training period.

7. Handcrafted Features

We replaced the outside LOB features (bid/ask price and volume for layers 6-10) with handcrafted expert features designed by [1]. We used random forests to help evaluate the relative importance of different features and selected short-term changes (derivatives) in the bid/ask price and volumes for layers 1-5. The addition of these handcrafted features resulted

in an increase in F1 by 14.5% and an increase in Kappa by by 21.4% compared to relying solely on the LOB price and volume data. We also noticed that the Loss in the Test set decreased more smoothly with the addition of the expert features.

8. Transfer Learning

We trained an LSTM model on all five stocks and then used the model to predict the out-of-sample category labels for each stock individually (5->1 model). We compared the performance of this model to the performance of models trained on each stock separately (1->1 models). We averaged the results on the five stocks across each of the two classes of models.

We found that the 5->1 model outperformed the 1->1 models on all metrics. In some cases the 1->1 models performed poorly out-of-sample due to the smaller amount of training data. The average Test F1 score of the 5->1 model was 7.65% greater than the 1->1 models. The average Kappa score of the 5->1 model was 49% higher than the 1->1 models.

9. Visualization

We We used gradient ascent to visualize what our CNN filters were learning. We found that filters learned to specialize in windows of different length analogous to the different lags used by traditional time series models such as ARMA.

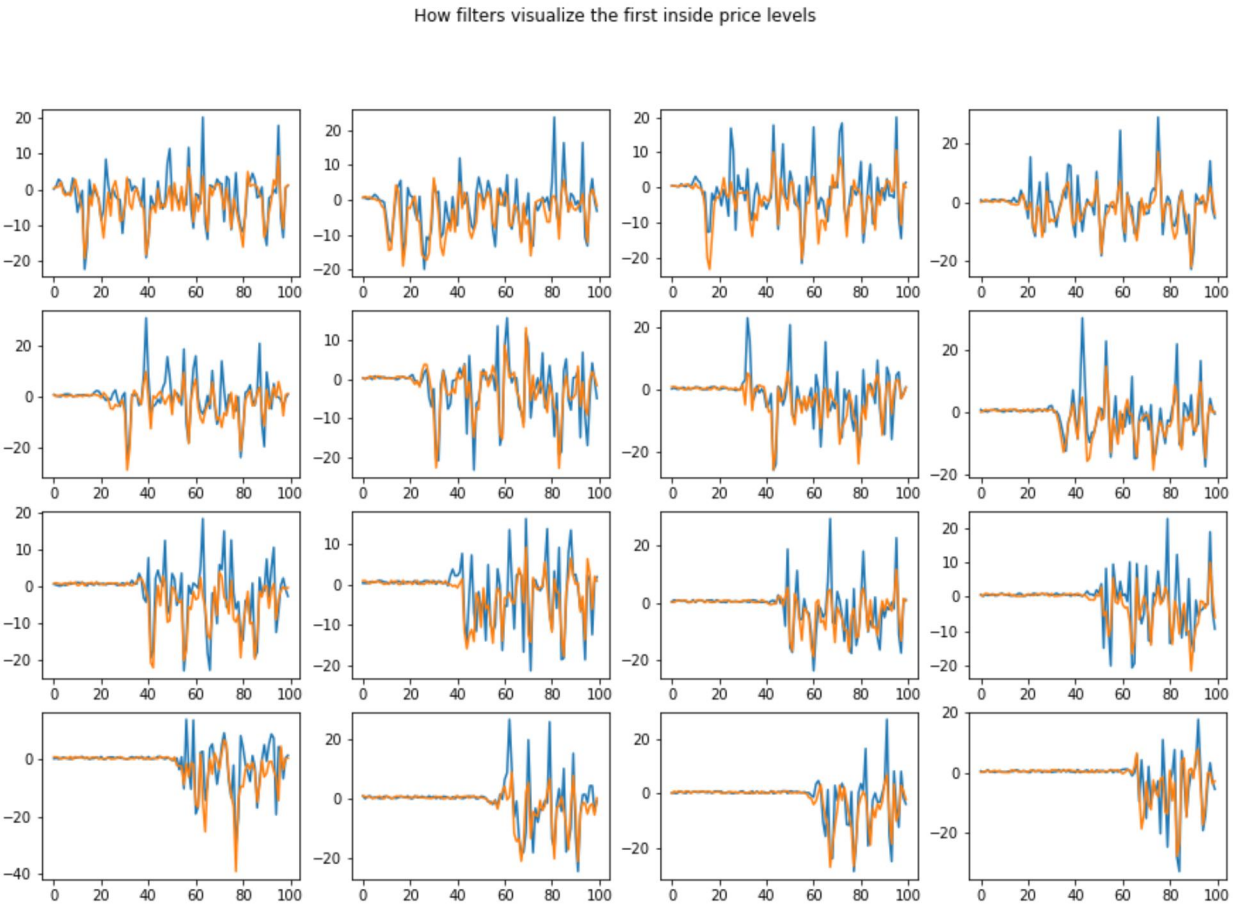

Figure 2: CNN filters learned to focus on feature windows of different length.

We examined the features of sequence windows which were assigned high probabilities for different category labels. When we we examined the windows individually we could not discern any pattern due to the noisiness of the data. However, when we averaged the values of the highest predictions for a given category label across all batches we could clearly identify certain model preferences.

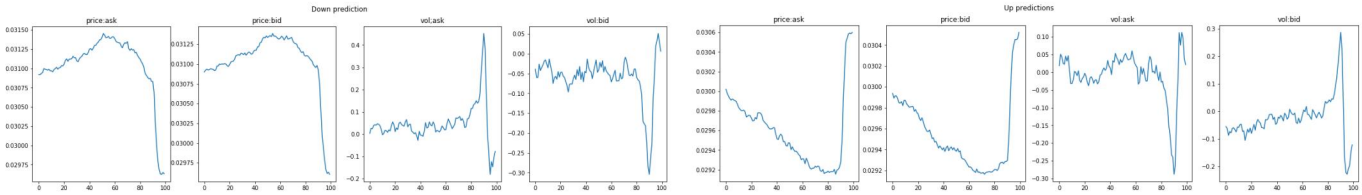

Figure 3: Average of actual input feature sequences associated with high predictions for a category (left side: down category for inside ask price, bid price, ask volume and bid volume; right side: up category for inside ask price, bid price, ask volume and bid volume)

We used gradient ascent to generate images of features associated with different categories. We found, for example, that the CNN-LSTM model generated inside bid/ask feature images with large spikes in opposite directions near the end of the window. The generated images in Figure 4 resemble the actual average bid/ask price features in Figure 3. We speculate that these images are not inconsistent with a microstructure phenomenon known as the bid-ask bounce where a stock price bounces rapidly back and forth between the bid price and ask price.

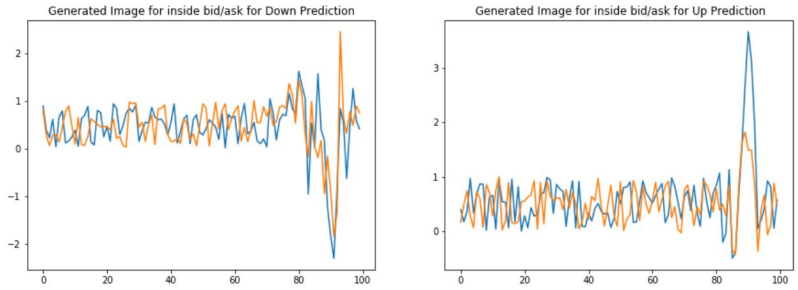

Figure 4: Input feature sequences generated using gradient ascent for different category labels: down(left) and up(right)

10.Future Directions

The performance of the dilated convents warrants further scrutiny and examination. We would like to apply these models to much larger and more recent LOB datasets. Finally we would like to explore higher dimensional visualization methods such as T-SNE.

References

1	2013	Kercheval and Zhang	Modeling high-frequency limit order book dynamics with support vector machines
2	2018	Ntakaris	Mid-price Prediction Based on Machine Learning Methods with Technical and Quantitative Indicators
3	2018	Tsantekidis	Forecasting Stock Prices from the Limit Order Book using Convolutional Neural Networks
4	2017	Tsantekidis	Using Deep Learning to Detect Price Change Indications in Financial Markets
5	2018	Ntakaris	Benchmark Dataset for Mid-Price Forecasting of Limit Order Book Data with Machine Learning Methods
6	2018	Zhang	DeepLOB: Deep Convolutional Neural Networks for Limit Order Books
7	2018	Nousi	Machine Learning for Forecasting Mid Price Movement using Limit Order Book Data
9	2016	Sirignano	Deep Learning for Limit Order Books
10	2018	Sirignano and Cont	Universal Features of Price Formation in Financial Markets
11	2018	Auguiers	Time Series Prediction Using LSTM Deep Neural Networks
12	2016	Oord	Wavenet: A Generative Model for Raw Audio
13	2017	Borovikh	Conditional Time Series Forecasting with Convolutional Neural Networks
14	2017	Jeddy	Time Series Forecasting with Convolutional Neural Networks - a Look at WaveNet

Contributions

There was originally a second person on my project. However, he made no significant contribution, apart from providing a link to reference [11], and I decided after the Milestone that I should proceed on my own. He informed me that he had downloaded the github repo (dunetz/cs230) which contained my work, and that he planned to use it as the basis for his own project. I have frozen this repo, as of the date of the Milestone, to help the teaching staff distinguish between his work and mine. My final project repo is dunetz/cs230final.