Alpha Generating DNN on Orderbook Microstructure

Name: Luhua Yang SID: 3036503321
Name: Zhiling Zhou SID: 3036433154
Name: Xinyu Li SID: 3036481567
Name: Yifei Tong SID: 3036414315
Name: Kefan Shi SID: 3036383817

10/02/2021

1 Introduction

Paper from 2021, PETTER N. KOLM, JEREMY TURIEL AND NICHOLAS WESTRAY. We are trying to predict stock price return in next moment using Orderbook states and Orderbook flow. We employ deep learning in forecasting high-frequency returns at multiple horizons for 6 stocks traded on Nasdaq using order book information at the most granular level. While raw order book states can be used as input to the forecasting models, we achieve better predictive accuracy by training simpler "off-the-shelf" artificial neural networks on stationary inputs derived from the order book. Specifically, models trained on order flow significantly outperform most models trained directly on order books.

2 Data Collection, Data Cleaning, and Data Preprocessing

2.1 Data Collection

The order book data is directly downloaded from the Onetick platform. The exchange we use is NASDAQ. We tried to query different times and find the valid date we have are 20170801, 20170824, 202170825, 202170828 - 20170831, 20170901, 20170905, 20170906, 20171003, 20171004, and 20171018-20171030. Since there are no continuous one month data, we choose one week data from 20171023 to 20171027. Because we use the tick by tick high frequency order book data, one week's data is already large enough to reflect the strategy's performance.

The universe is:

- Stock AAPL(Large Cap, 867.9B USD, Oct 30, 2017)
- Stock GOOGL(large Cap, 712.03B USD, Oct 30, 2017)
- Stock MSFT(Mid-large Cap, 641,7.03B USD, Oct 30, 2017)
- Stock FB(Mid-large Cap, 523.22B USD, Oct 30, 2017)
- Stock NFLX(Small Cap, 85B USD, Oct 30, 2017)
- Stock AAL(Small Cap, 22.4B USD, Oct 30, 2017)

These 6 equities represent three level of liquidity. AAPL possessed the best liquidity and AAL was the worst. Besides, the order book data is composed of the 5 level bid/ask price and size.

2.2 Data Cleaning

2.3 Order Flow

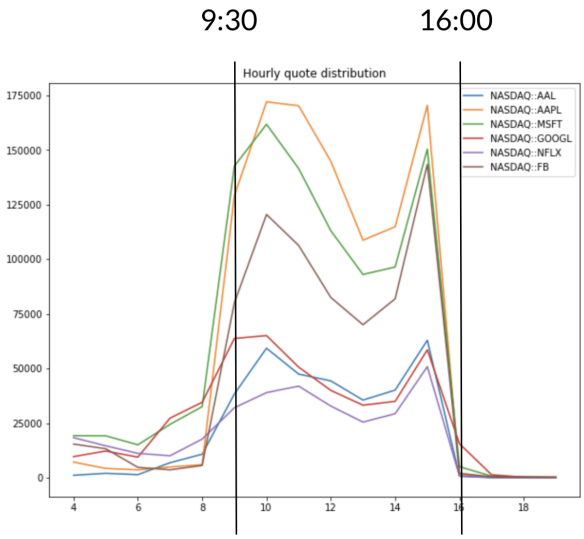

Figure 1: Quote distribution

We plot the distribution of quote data. We find that, there are many pre- and post- market data which lie before 9:30am and after 4pm EST. But they are relatively sparse compared with regular market session data.

Considering the model should run at regular market session, we filter data from 9:30am 4pm EST and drop the pre-market and post-market data.

3 Feature Engineering

We use two main features here in the analysis. The first one is order states, the second one is order flow.

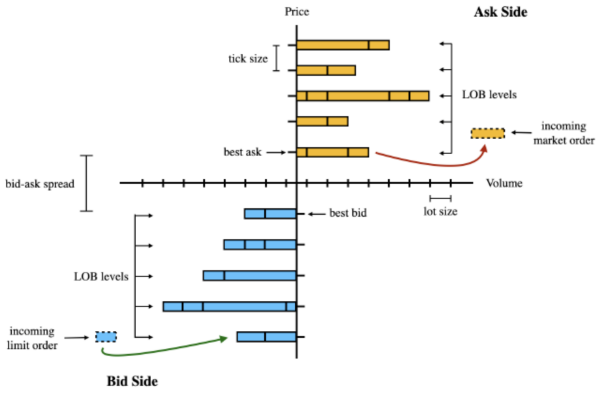

Figure 2: Limit Order Book Illustration

3.1 Order States

We use top 5 level of bid ask price and size as order states features.

3.2 Order Flow

Order flow is a measure of current bid and ask order updates. We calculate the top 5 level order flows using the following formulas.

$$\mathrm{bOF}_{t,i} := \begin{cases} v_t^{i,b}\,, & \text{if } b_t^i > b_{t-1}^i\,, \\ v_t^{i,b} - v_{t-1}^{i,b}\,, & \text{if } b_t^i = b_{t-1}^i\,, \\ -v_t^{i,b}\,, & \text{if } b_t^i < b_{t-1}^i\,, \end{cases}$$

$$\mathrm{aOF}_{t,i} := \begin{cases} -v_t^{i,a}\,, & \text{if } a_t^i > a_{t-1}^i\,, \\ v_t^{i,a} - v_{t-1}^{i,a}\,, & \text{if } a_t^i = a_{t-1}^i\,, \\ v_t^{i,a}\,, & \text{if } a_t^i < a_{t-1}^i\,, \end{cases}$$

Figure 3: Formula for order flow

3.3 Labeling

We use dollar return as labels. According to the paper, we calculate the dollar return as follows:

Let $\Delta t = 2.34 \times 10^7 / N$, where 2.34×10^7 is the number of milliseconds in a regular market session, and N is the number of valid samples.

Let $h_k = 1/5 \times k\Delta t$. And let $r_{t,k} = p_{t+h_k} - p_t$. Here, for simplicity, we let $k = 10$.

For regression model, such as ARX, we use continuous labels. And for classification models, such as LSTM and CNN-LSTM, we transform dollar returns to categorical labels. We try different methods to match the labels. And they have different advantages and disadvantages.

3.3.1 Naive method

It uses Pandas' resample with fixed time interval.

- Advantages: Time efficiency.
- Disadvantages: Many data fall in the same interval, after resampling, they are lost. Roughly, we lost about 2/3 rows of the data.

3.3.2 Improved method

We implement this method in practice: For every sample, find its corresponding label. E.g. for a sample at time t, find the mid-quote closest but less than time $t + \Delta t$

- Advantages: It keeps all the sample data. It is more reasonable labeling because every sample is mapped to a label that should take place Δt later.
- Disadvantages: Time-consuming

3.3.3 Transform continuous labels to categorical labels

For classification models, such as LSTM and CNN-LSTM, we transform dollar returns to categorical models according to the rule that: if $r_{t,k} > 0$, label as "up"; if $r_{t,k} = 0$, label as "unchanged"; if $r_{t,k} < 0$, label as "down".

And we calculate the distribution of categorical labels and find that they are imbalanced, where "unchanged" labels consist of 60% of all labels. We then use subsampling method against "unchanged" labels to mitigate the imbalance problem. Particularly, we subsample 25% for "unchanged" labels.

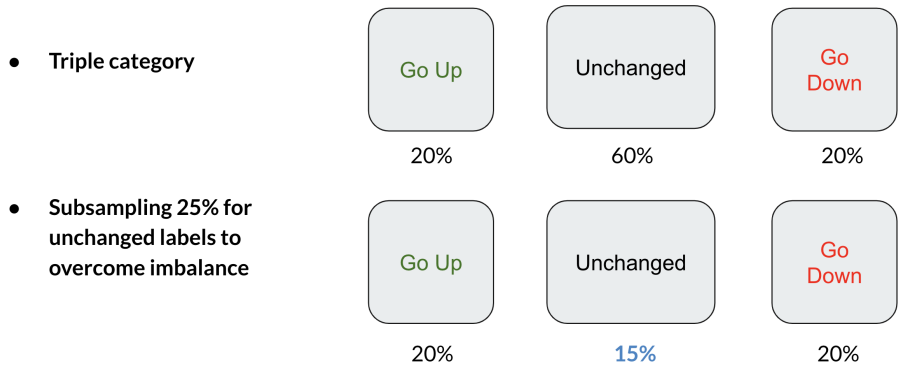

Figure 4: Subsampling

4 Benchmark Model: ARX

Autoregressive with exogenous inputs (ARX) takes LOBs or OFs features as exogenous inputs and predicts dollar mid-quote returns next time point.

$$y_t = w_0 + \sum_{i=1}^{n_y} w_i y_{t-i} + \sum_{i=1}^{n_x} \mathbf{v}_i^T \mathbf{x}_{t-i} + \epsilon_t \tag{1}$$

We can estimate ARX using OLS. We try different parameter for lags and we find that the results are not sensitive to the number of lags. And we choose ARX(5) according to BIC.

4.1 Out-of-Sample Performance

	GOOGL	AAPL	MSFT	NFLX	FB	AAL
Unconditional mean	0.040833	0.423963	0.270382	0.456964	0.374195	0.332392
LOB	0.028099	0.305723	0.167117	0.299816	0.259761	0.207467
OF	0.026668	0.156517	0.1072523	0.260680	0.158901	0.137286

Table 1: MAE

We use unconditional mean predictions as a benchmark. For unconditional mean's MAE of GOOGL 0.0408, it implies the predictions have approximately 0.0408 USD of average error. From the chart, we can conclude that, the performance of ARX with OFs features is better than ARX with LOBs features. And both perform better than the benchmark.

	GOOGL	AAPL	MSFT	NFLX	FB	AAL
Unconditional mean	716790353.44	136.13	478.11	425.73	122.43	228.19
LOB	6911365841.90	555.05	2241.08	1380.05	543.58	345.47
OF	5543515548.72	209.99	1284.98	1225.19	283.79	209.09

Table 2: MAPE

We also calculate MAPE. But because there are lots of 0 and nearly 0 values in true dollar returns, MAPE of GOOGL is too large to provide meaningful information. While from other MAPE, we can conclude that, the performance of models with OFs as features are better than their corresponding models with LOBs as features, which is consistent with the conclusion in the MAE part. However, ARX models tend to perform worse than the mean predictors.

4.2 ARX as Classifier, Out-of-Sample Performance (GOOGL)

To better comparing with latter NN models, we also transform the predictions of ARX to categorical predictions. To get a more balanced label, we choose 5×10^{-4} as threshold. If prediction $y > 5\times10^{-4}$, label as "up"; if $y < -5\times10^{-4}$, label as "down"; otherwise, label as "stationary". Then we obtain the confusion matrices and some statistics.

	up	stationary	down
up	6554	118	1312
stationary	9467	796	7302
down	2016	166	5732

Table 3: Confusion matrix: LOBs as features

Note: true labels by row and predictions by column.

	up	stationary	down
up	6506	86	1392
stationary	9106	895	7563
down	1562	139	6213

Table 4: Confusion matrix: OFs as features

Note: true labels by row and predictions by column.

	LOB	OF
Macro precision	0.49998	0.53016
Macro recall	0.52918	0.55030

From the charts, we can conclude that, the performance of ARX with OFs features is better than ARX with LOBs features becuse the former model has higher macro precision and recall.

5 Neural Network Models

It is hard to train regression models. Instead, we convert the problem into a classification problem. We use labeling method introduced in the labeling part.

5.1 LSTM

The first neural network model we are exploring here is LSTM. The sequence length is 20, parameter size is 10. We pick the optimal sequence length according to the paper's choice of 18, we then tried several other values and decided that 20 is the best fit for our data set. The structure is as follows:

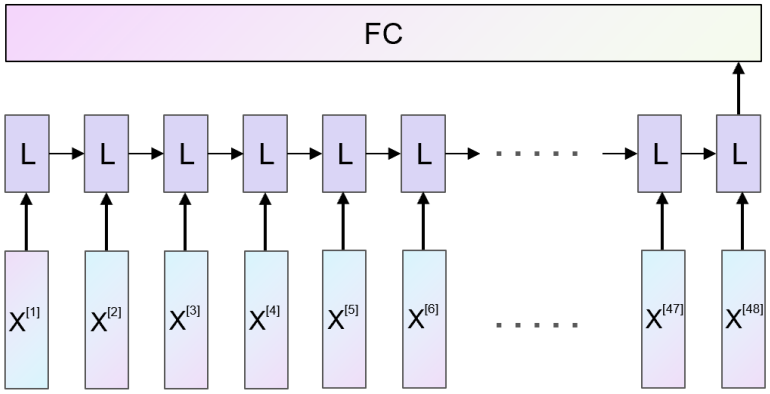

Figure 5: Architecture for LSTM Model

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
batch_normalization (BatchNo (None, 20, 10)            40
_________________________________________________________________
lstm (LSTM)                  (None, 20)                2480
_________________________________________________________________
dense (Dense)                (None, 3)                 63
=================================================================
Total params: 2,583
Trainable params: 2,563
Non-trainable params: 20
```

Figure 6: Parameters for LSTM Model

5.2 CNN-LSTM

The second model we have experimented here is the CNN-LSTM model. It is the advanced version of our previous LSTM model. Here we add another convolutional layer in the front of the LSTM layer, in order to extract more information from a longer sequence of data. There here the first input layer takes in a sequence of length 100. after a CNN layer and max pooling layer, the sequence length is recuded to 48. The the 48×20 matrix is passed to a double stacked LSTM layer. The structure is as follows:

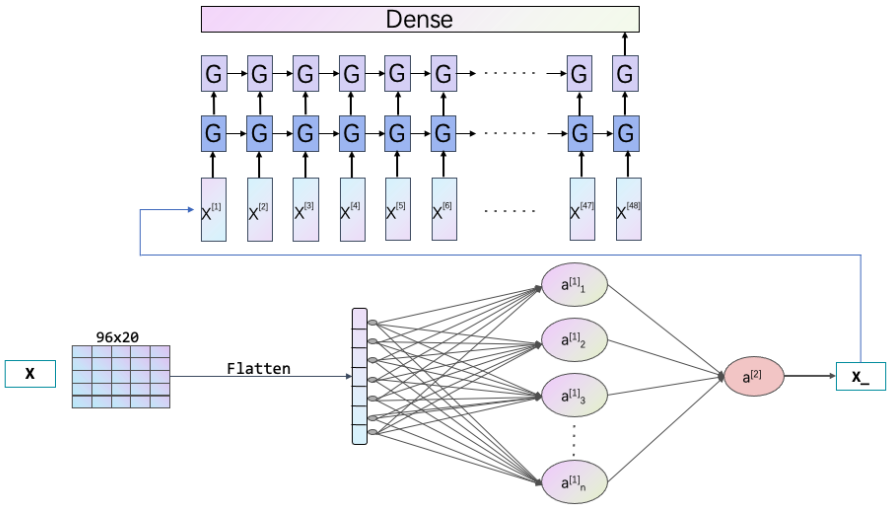

Figure 7: Architecture for CNN-LSTM Model

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
batch_normalization_2 (Batch (None, 100, 10)           40
_________________________________________________________________
conv1d (Conv1D)              (None, 96, 20)            1020
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 48, 20)            0
_________________________________________________________________
lstm_2 (LSTM)                (None, 48, 20)            3280
_________________________________________________________________
lstm_3 (LSTM)                (None, 20)                3280
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 63
=================================================================
Total params: 7,683
Trainable params: 7,663
Non-trainable params: 20
```

Figure 8: Parameters for CNN-LSTM Model

6 Evaluation and Discussion, Neural Networks

Here we showed several graphs of the training and validation loss of our neural network models. We use GOOGL and MSFT as examples here. As we can see, LSTM with subsampling has a less training and validation loss than the one without subsampling. We believe that the model has an improved prediction power after the data balancing. Another intersting point is that CNN-LSTM model tends to overfit. Since the training loss is decreasing while the validation loss is increasing after epoch 6. There we conclude that LSTM with subsampling is the optimal model in our data set.

6.1 Training and Validation Loss, GOOGL

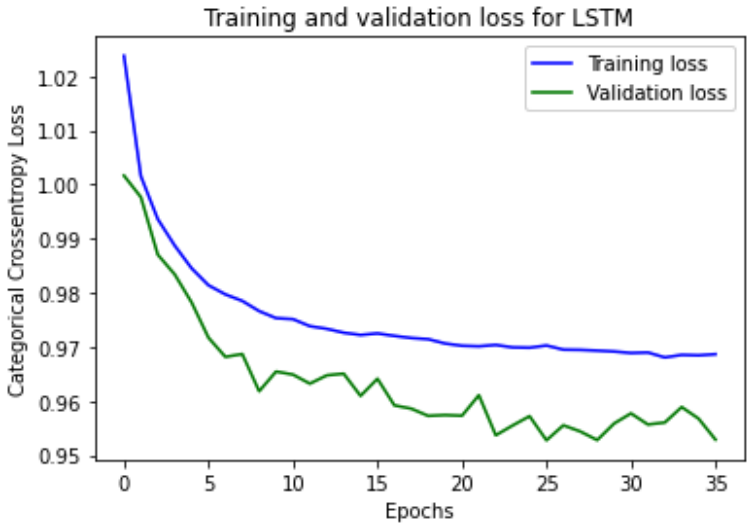

Figure 9: Train and Validation Loss for LSTM, Without Subsampling

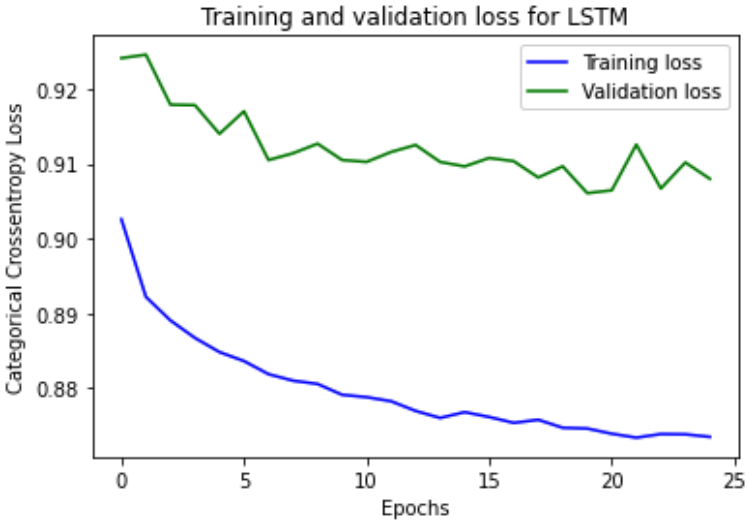

Figure 10: Train and Validation Loss for LSTM, With Subsampling

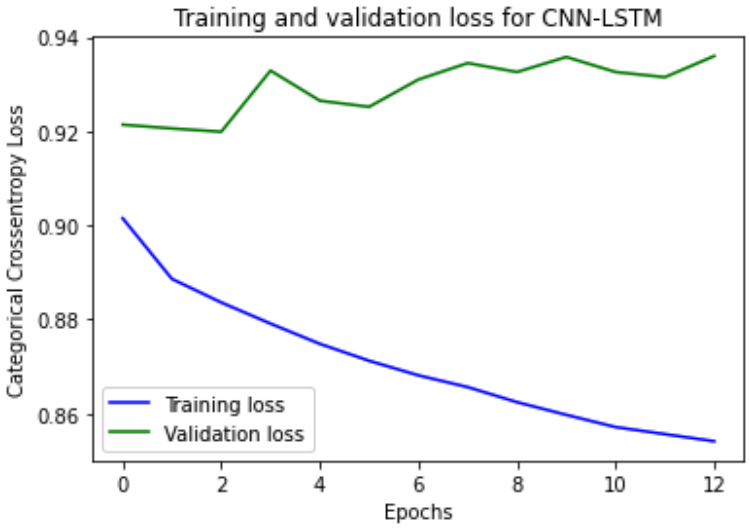

Figure 11: Train and Validation Loss for CNN-LSTM, With Subsampling

6.2 Training and Validation Loss, MSFT

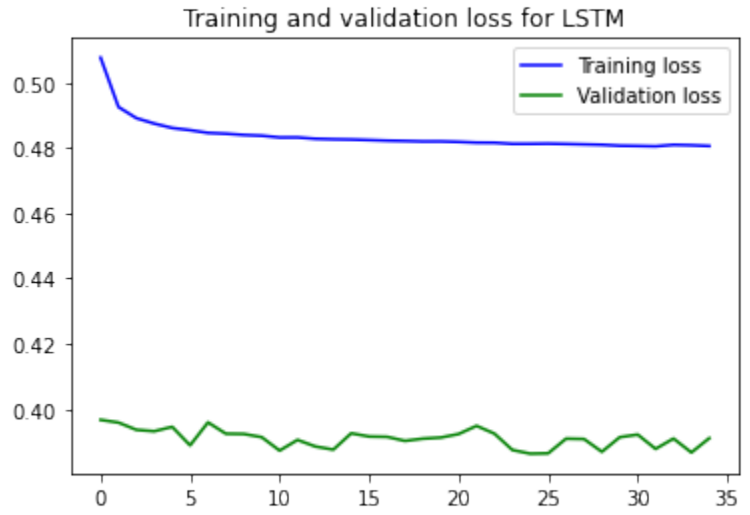

Figure 12: Train and Validation Loss for LSTM, Without Subsampling

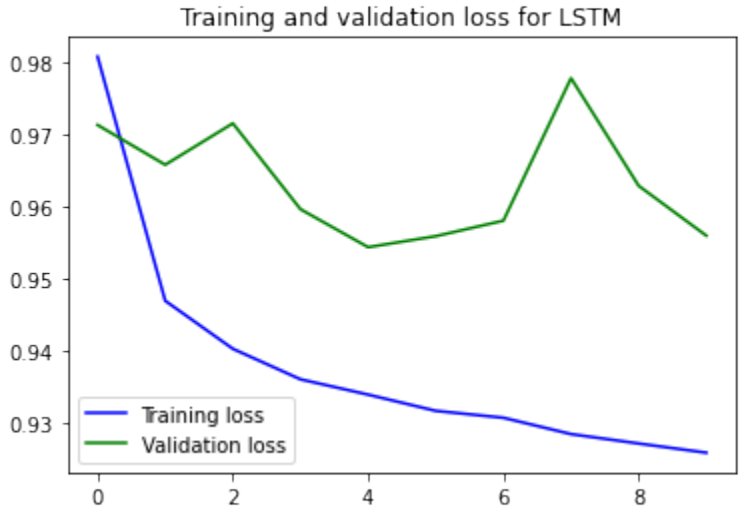

Figure 13: Train and Validation Loss for LSTM, With Subsampling

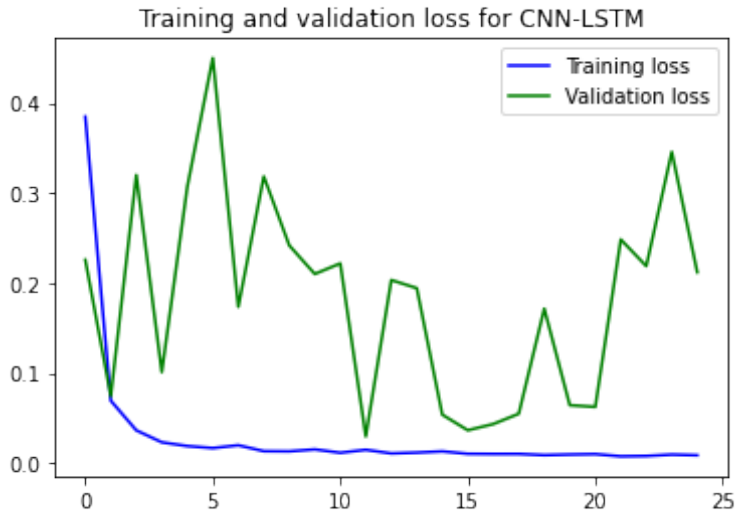

Figure 14: Train and Validation Loss for CNN-LSTM, With Subsampling

6.3 Out-of-Sample Performance (MAE)

The our of sample MAE shows that the LSTM with subsampling produces a consistent result across different stocks. However, it is still larger compared our ARX model. This indicates that the Neural networks may have a worse performance compared to regression model in this scenerio.

	GOOGL	AAPL	MSFT	FB
LSTM, No Subsampling	0.9627	0.6902	0.3911	0.7699
LSTM, With Subsampling	0.9080	0.9755	0.9559	0.9793
CNN-LSTM, With Subsampling	0.9360	1.0107	0.9461	0.9888

6.4 Confusion Matrix, GOOGL

The confusion matrix of our model for the GOOGL dataset. Here our LSTM with subsampling model is a classification model with predicts 3 classes including up, down and stationary. It is surprising to see that it does not have an stationary prediction, which means the model tends to have a favor of the direction rather than stay stationary.

In terms of macro precision and recall, the LSTM with subsampling model performs better than the ARX.

	Pred: up	stationary	down
True:up	2556	0	1772
stationary	1832	0	2632
down	393	0	477

Table 5: Confusion matrix: OFs as features

Note: true labels by row and predictions by column.

	OF
Accuracy	0.5369
Macro precision	0.59006
Macro recall	0.59008

Table 6: Confusion matrix: Precision and Recall

7 Conclusion

Our work shows that order flows data can be used as features in predicting price movements. Order book and order flow data required lots of data cleaning and feature engineering work, including matching labels, subsampling and converting numerical labels to categorical labels. We try LOBs and OFs features for ARX model and find that it performs better when using OFs as features. And we construct different NN models, including LSTM, LSTM with subsampling and CNN-LSTM with subsampling, using OFs features. The training loss graphs show in the examples of GOOGL and MSFT that the training performance of LSTM with subsampling is the best. Also, we use MAE to compare the performance of NN with the ARX benchmark. We find that NN may have worse performance. Finally, through the confusion matrix, we compare the classification performance for GOOGL data set. It shows that LSTM

with subsampling model has better performance than ARX according to macro precision and recall.