Neural Networks for Limit Order Books

Justin A. Sirignano
Department of Mathematics, Imperial College London
Mathematical Finance Section

December 31, 2015 *

Abstract

This paper develops a new neural network architecture for modeling spatial distributions (i.e., distributions on $\mathbb{R}^d$) which is computationally efficient and takes advantage of local spatial structure. We find statistical evidence for local spatial structure in limit order books, motivating the new neural network's application to limit order books. The neural network is trained and tested on nearly 500 stocks. The neural network uses information from deep into the limit order book (i.e., many levels beyond the best bid and best ask). Techniques from deep learning such as dropout are employed to improve performance. Due to the significant computational challenges associated with the large amount of data, training is parallelized across 20 GPUs. The "spatial neural network" is shown to outperform other models such as the naive empirical model, logistic regression (with nonlinear features), and a standard neural network architecture.

*The author thanks the Mathematical Finance Section of the Department of Mathematics at Imperial College London for generously providing funds for computations. The author also thanks Apaar Sadhwani (Stanford University), Kay Giesecke (Stanford), Rama Cont (Imperial College London), and David Harding (Winton Capital Management) for insightful comments.

1 Introduction

The "limit order book" is a system used by financial exchanges such as NASDAQ and NYSE to match buyers and sellers of stocks. It governs how buy and sell orders can be submitted, when and how orders are executed, and ultimately the stock's price evolution. The complexity and high-dimensionality of limit order books make modeling challenging. A limit order book consists of hundreds of price levels where orders may be submitted and its dynamics are nonlinear. Modeling requires the analysis of large amounts of data, which can be both statistically and computationally challenging.

In spite of the wealth of research on limit order books, there is very little literature adapting machine learning methods to the limit order book setting. In this paper, we design and test neural networks for modeling limit order book dynamics. Neural networks are particularly well-suited for limit order books due to their ability to perform well with high-dimensional data and capture nonlinear relationships. Neural networks also scale favorably with large amounts of data. In addition to investigating the performance of standard neural networks (originally designed for classification), we develop a new neural network architecture for modeling spatial distributions (i.e., distributions on $\mathbb{R}^d$). This new architecture has several advantages over the standard neural network architecture for modeling distributions on $\mathbb{R}^d$, including better generalization over space, lower computational expense, and the ability to take advantage of any "local spatial structure". For the dataset considered in this paper, this "spatial neural network" has lower out-of-sample error, much faster training times, and greater interpretability than the standard neural network.

Models are trained and tested using limit order book data for 489 S&P 500 and NASDAQ-100 stocks over the time period January 1, 2014 to August 31, 2015. In total, there is roughly 50 terrabytes of raw data, which is filtered to create training, validation, and test sets for the limit order book at discrete time intervals. There are substantial technical challenges to analyzing the large amounts of data and model training is computationally expensive. Distributed storage and parallel computing are used accelerate data processing. 20 GPUs are used to train and test deep neural networks.

We compare several approaches for modeling the joint distribution of the best ask and best bid prices at a future time conditional on the current state of the limit order book. In out-of-sample tests, neural networks strongly outperform simpler approaches such as assuming the naive empirical distribution or logistic regression (even with nonlinear features). Failure to outperform the naive empirical model would imply that the state of the limit order book contains no information on future price movements. The spatial neural network outperforms the standard neural network. Both neural networks have several hidden layers and are trained using methods from deep learning such as dropout and inter-layer batch normalization.

The spatial neural network's outperformance of the standard neural network can be partly attributed to its taking advantage of the limit order book's local spatial structure. We find statistical evidence for a particular form of local behavior in limit order books. The spatial neural network's architecture mimics this local behavior, yielding a low-dimensional model of movements deep into the limit order book. This allows the spatial neural network to effectively use information from deep in the limit order book (beyond the best bid and best ask). The spatial neural network provides a significant increase in performance over

the standard neural network for more volatile stocks which have a stronger dependance on liquidity deeper into the limit order book.

It is to be emphasized that this paper models the *joint distribution* of the best bid and best ask prices. This is essential for risk management applications (e.g., computing value-at-risk). This paper's approach could also be easily used to model the joint distribution of the best bid, best ask, and other quantities. For example, it might be of interest to model the joint distribution of the best bid price, best ask price, the queue size at the best bid, and the queue size at the best ask. Finally, although this paper focuses on limit order books, the spatial neural network provides benefits for any setting which requires modeling a distribution on $\mathbb{R}^d$.

1.1 How does a Limit Order Book Work?

Stocks are traded via matching buy and sell orders according to an order-driven system. Orders may only be submitted at discrete price levels (determined by the "tick-size", which is \$.01 in the USA). A *limit order* is a buy or sell order for a stock at a certain price. The limit order will appear in the limit order book at that price and remain there until executed or cancelled. The "limit order book" consists of all limit orders at all prices. The "bids" are the buy limit orders and the "asks" are the sell limit orders. The best ask price is the lowest sell limit order and the best bid price is the highest buy limit order. Oftentimes there can be a spread between these prices (i.e., empty price levels with no orders between the best bid and best ask prices). A *market order* is an order to immediately buy or sell the stock. A market buy order is executed at the best ask price while a market sell order is executed at the best bid price; a market order consumes some (or all) of the supply/demand at the best ask/best bid prices.

Figure 1 shows an example of a limit order book. The limit order volume j discrete price levels from the best ask price is referred to as the *volume at level* j. In Figure 1, the spread is one tick (i.e., a single level).

The limit order book represents the supply and demand for the stock at different price levels. Over time, the limit order book (and with it the best ask and best bid prices) will evolve due to new limit orders, cancellations, and market orders. For practical purposes, it is of greatest interest to model the future distribution of the best ask and best bid prices given the current state of the limit order book. The best ask and best bid prices at time t are the prices at which a market participant can immediately buy or sell the stock at time t.

The mid-price is the average of the best bid and best ask prices, and is taken as the "price" of the stock. However, it is an artificial quantity since one cannot buy or sell at the mid-price. Consequently, it is important to model both the best ask and best bid prices.

1.2 Related Literature

Significant research has been conducted with regards to limit order book dynamics and several modeling approaches have been developed. Cont, Stoikov & Talreja (2010), Cont & Larrard (2013), Avellaneda & Stoikov (2008), and Avellaneda, Reed & Stoikov (2011) develop stochastic models for limit order book

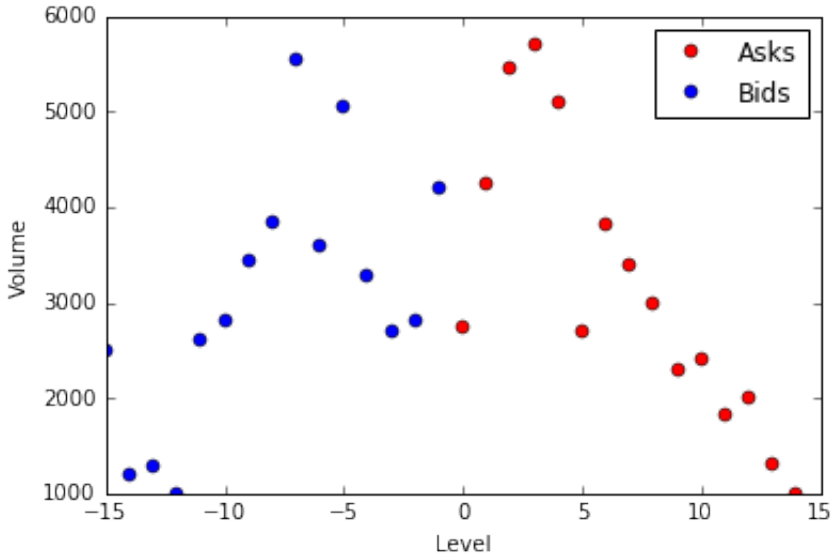

Figure 1: Limit order volumes for first 15 ask and bid prices for Microsoft. Level 0 is the best ask price.

dynamics. Another related vein of research considers the optimal execution of a buy or sell order. Optimal execution requires understanding the price impact of an executed order given the current state of the limit order book. Cont, Kukanov & Stoikov (2014), Eisler, Bouchaud & Kockelkoren (2012), Maglaras, Moallemi & Zheng (2015), Moallemi, Park & Roy (2012), Alfonsi, Fruth & Schied (2010), Alfonsi & Schied (2010), Predoiu, Shaikhet & Shreve (2011), Schied & Schoneborn (2009), Bayraktar & Ludkovski (2014), and Donier, Bonart, Mastromatteo & Bouchaud (2014) study this problem and develop approaches for optimal execution. Stochastic models have also been developed for limit order books using asymptotic approximations; this is quite natural given the nature of limit order books. Limit order books have large amounts of orders as well as a large number of ticks (i.e., price levels). Cont & Larrard (2012), Blanchet & Chen (2013), and Gao, Dai, Dieker & Deng (2014) develop limiting laws in this spirit. A number of other papers such as Cont (2011), Bouchaud, Mezard & Potters (2002), Potters & Bouchaud (2003), Biais, Hillion & Spatt (1995), Gould & Bonart (2015), and Gould, Porter, Williams, McDonald, Fenn & Howison (2013) empirically study the statistical characteristics of limit order books.

There is relatively little literature on machine learning approaches to limit order books (or financial applications in general). Kearns & Nevmyvaka (2006) use reinforcement learning for optimal order execution. Kercheval & Zhang (2015) use support vector machines to model limit order books. Kempf & Korn (1999) study the relationship between price changes and net order flow for German index futures with neural networks. Fletcher & Shawe-Taylor (2013) use multiple kernel learning for currency exchange rate prediction. Outside of limit order books, Khandani, Kim & Lo (2010), Butaru, Chen, Clark, Das & Lo (2015), and Sirignano, Sadhwani & Giesecke (2016) use machine learning approaches for modeling consumer and commercial loan risk. Mamaysky & Glasserman (2015) develop sentiment analysis methods for the prediction of market stress based upon news articles. Chinco, Clark-Joseph & Ye (2015) use LASSO to study cross-stock

information diffusion.

1.3 Advantages of Neural Networks

Neural networks are particularly well-suited to limit order book modeling due to their highly flexible and nonlinear characteristics, successful performance for high-dimensional inputs, and favorable scaling with large amounts of data. There are certainly other methods which could be applied. Decision trees, boosted trees, and random forests are also able to learn nonlinear functions. However, their disadvantage is that they divide the input space into rectangular cells while neural networks can learn arbitrary functions of the input space. The ability to learn arbitrary functions (i.e., to *generalize*) is essential for high-dimensional inputs; dividing a high-dimensional space into rectangular cells quickly suffers from the curse of dimensionality. Decision trees are also not optimized for online learning; upon the arrival of new data, typically the complete structure of the decision tree will change and it must be re-formed from scratch. In contrast, neural networks are easily trained online: the parameters can be simply be updated with minibatch gradient descent. Gaussian process regression is another method which rivals neural networks in accuracy for small data sets. Unfortunately, Gaussian process regression does not scale well and quickly becomes intractable for larger data sets.

Neural networks have achieved major success on many classification tasks, such as image classification. Neural networks have 99% accuracy on the MNIST dataset and 95% accuracy on the CIFAR-10 dataset.[1]

1.4 Organization of this Paper

Section 2 describes the dataset. Section 3 presents evidence for local spatial structure in limit order books. In Section 4, different neural network architectures are analyzed for modeling spatial distributions. In particular, Section 4.3 develops the new architecture for modeling spatial distributions. The deep learning approaches and the GPU computational framework used to train neural networks are explained in Section 5. Out-of-sample results for predicting the distribution of the best bid and best ask are reported in Section 6. Section 6 also includes analysis of the numerical results.

2 The Data

We use Level III limit order book data from the NASDAQ stock exchange. For each stock, there is event-by-event data recording the current state of the limit order book. Every event (order submission, cancellation, and transactions) in the limit order book is recorded as well as the state of the limit order book at the time of each event. The times of events are reported with millisecond to nanosecond decimal precision, depending upon the time period. Between events, the limit order book state does not change. The limit order book data

[1] The MNIST dataset contains images of handwritten digits. The goal is to classify correctly the handwritten digits. The CIFAR-10 dataset is composed of images from ten different classes (e.g., automobiles, birds, dogs). The goal is to classify correctly an image as one of the ten classes.

includes the first 100 *nonzero* levels in the limit order book (50 on the ask side and 50 on the bid side). The nonzero levels are levels at which there is a nonzero volume. Thus, the dataset includes at least the first 100 levels and typically many more due to a large fraction of levels having zero volumes. At each nonzero level, the volume is reported.

The data includes trading halts. During the trading halts, the limit order book is reported as unchanging. These samples are removed from the dataset for model training and testing. Trading halts can occur for various reasons, including extraordinary volatility, regulatory concerns, SEC trading suspensions, or unusual market activity indicating a technical issue or manipulation. Trading halts occur infrequently for the stocks in this paper's dataset.

The data used in this paper comes from the time period January 1, 2014 until August 31, 2015 and includes 489 stocks primarily drawn from the S&P 500 and NASDAQ-100. The large number of stocks and long time period increase the robustness of the results in this paper. Notable stocks in the dataset include Facebook, Apple, Netflix, Amazon, Amgen, Bank of America, Microsoft, Boeing, Berkshire Hathaway (Class B shares), Broadcom, and Caterpillar. A full list of stocks is provided in Appendix A. In total, the raw data is roughly 50 terrabytes, which is filtered to create training, validation, and test sets for the limit order book.

We train and test models for two prediction cases:

1. Modeling the joint distribution of the best ask and best bid prices at time $t + \Delta t$ given the current state of the limit order book at time t.

2. Modeling the joint distribution of the best ask and best bid prices upon the *next price move*. The next price move is defined as the first time at which the best bid price or best ask price changes.

For Case [1], models are trained and tested specifically for $\Delta t = 1$ second, although the methodologies are of course applicable to any time horizon Δt. For Case [2], the time horizon is random. Specifically, if $\tau_1, \tau_2, \ldots$ are the times at which either the best ask price changes, the best bid price changes, or both change, we model the joint distribution of the best ask and best bid prices at time τ_{k+1} given the current state of the limit order book at time τ_k. Thus, $\Delta\tau_k = \tau_{k+1} - \tau_k$ can vary widely from a fraction of a second to many seconds. More volatile stocks will have more data samples while less volatile stocks will have less data samples. For instance, AAPL has 27 million samples while FOX has 2 million samples. In Case [1], all stocks will have approximately 10 million data samples. Case [1] considers a much less volatile quantity since frequently the best bid and best ask prices do not change[2], while Case [2] conditions on a change occuring.

Processing and storage of the raw dataset is challenging due its large size. We use distributed storage and parallel computing to store and process this data. Training complex models with many parameters (such as neural networks with multiple layers) on such a large data is computationally expensive. Models are trained and tested using GPU clusters; see Section 5 for more details.

[2]Typically, the best ask price changes only 10-15 % of the time.

2.1 Nonlinearity of Limit Order Books

The goal is to use neural networks to capture nonlinear relationships between the state of the limit order book and the distribution of future best bid and best ask prices. It is well-known that the limit order book has a nonlinear relationship with future price movements.[3] An example of nonlinear behavior for the stock Boeing is shown below in Figure 2, where one can see that the probability the best ask price decreases has a strong nonlinear dependence on the volumes at the best ask and best bid. The probability shown in the figure is the output of a neural network fitted to historical data for Boeing. The displayed relationship shows a strong dependance on supply and demand. As the best ask volume increases (more selling pressure), the probability of a decrease in the best ask increases. As the best bid volume decreases (less buying demand), the probability of a decrease in the best ask increases.

We show later that neural networks have consistently lower error than logistic regression across many different stocks. A logistic regression is the softmax of a linear function of the input while a neural network is the softmax of a nonlinear function of the input. As an input to the logistic function, we include both the original data and nonlinear features of the original data. Specifically, we include the order imbalances at each level, which is a nonlinear function of the volumes at each level.[4] Order imbalance (sometimes referred to as "queue imbalance") has been identified as a key driver of best bid and best ask price dynamics; see Gould & Bonart (2015), Cao, Hansch & Wang (2009), Yang & Zhu (2015), Cartea, Donnelly & Jaimungal (2015), Stoikov & Waeber (2015), and Lipton, Pesavento & Sotiropoulos (2013). The neural networks' outperformance indicates the existence of significant nonlinearity in limit order book dynamics, beyond that of the well-known order imbalance feature.

2.2 Importance of modeling the *joint* distribution of best ask and best bid prices

The spread is the difference between the best bid price and best ask price. The spread is sometimes modeled as a constant. This reduces modeling the joint movements of the best bid price and best ask price to simply modeling the best ask price. Such an approach is equivalent to modeling the best bid price and best ask price as moving in lockstep: if one moves by k levels, the other also moves by k levels. However, statistics shown in Table 1 demonstrate that the spread frequently varies. Table 1 is based upon the data case [1] at a fixed 1 second time horizon.

For each stock j of the 489 stocks in the dataset, we calculate the fraction of the time where the best bid price and best ask price change by the same amount conditional on a change in either the best bid price or best ask price occuring (denoted Z^j in Table 1). That is, how frequently do the best bid and best ask prices move in lockstep? We then compute the quantiles of $Z^1, \ldots Z^{489}$. For the majority of stocks, the best bid

[3] For instance, see Figure 4 in Cont & Larrard (2013).

[4] Let V_k^a be the volume at the level $(k + \text{best ask price})$ and let V_k^b be the volume at the level $(\text{best bid price} - k)$. The order imbalance at the k-th level is $\frac{V_k^b - V_k^a}{V_k^a + V_k^b}$. The order imbalance ranges from -1 to 1 and measures the imbalance between supply and demand for the stock.

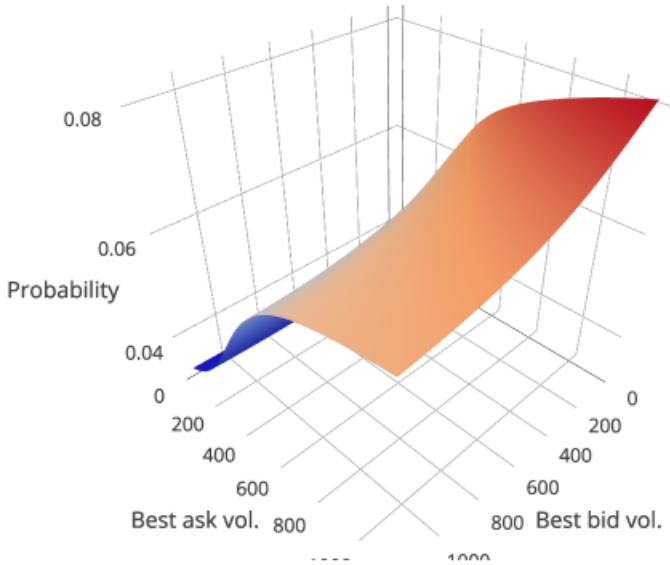

Figure 2: Probability that the best ask price for Boeing decreases for a 1-second horizon. Neural network fitted using best ask volume, best bid volume, and spread.

price and best ask price will more than 50% of the time not move in lockstep. For half of stocks, the best bid price and best ask price only move in lockstep 17% of the time. Table 1 also reports the fraction of time where the best bid price changes but the best ask price does not (or vice versa) conditional on a change in either the best bid price or best ask price occuring (denoted V^j). For the vast majority of stocks, a change in one of the two will occur without a change in the other at least 25% of the time. For half of stocks, a change in one of the two will occur without a change in the other nearly 70% of the time. In total, the first two rows of Table 1 provide strong evidence that the movements of the best ask and best bid must both be modeled. One cannot simply model the mid-price and assume that the spread is a constant number of levels. This highlights the importance of a model for the *joint* distribution of the best ask and best bid prices.

In addition, for each stock j of the 489 stocks, we calculate the 10 %, 50 %, and 90 % quantiles of the spread for stock j. We then compute the quantiles of $Q_p^1, \ldots Q_p^{489}$ where Q_p^j is the p-percentile quantile for stock j. This yields insight into the size of the spreads across the stocks in the dataset.

Feature / Quantile (%)	5	10	20	50	80	90	95
Z^j	.03	.05	.08	.17	.39	.58	.70
V^j	.26	.36	.52	.67	.74	.76	77
Q_{10}^j	1	1	1	1	2	3	6
Q_{50}^j	1	1	1	2	4	8	13
Q_{90}^j	1	1	2	5	10	18	28

Table 1: Summary statistics for the spread and the co-movement of the best bid and best ask prices.

3 Local spatial structure in limit order books

Limit order books exhibit some degree of local spatial structure. In Section 4.3, a new architecture for neural networks is designed which can take advantage of such local spatial structure. In this section, we provide statistical evidence for local spatial structure in limit order books.

Without loss of generality, let the current best ask price at time t be level 0 and let the best ask price at time t be the frame of reference for the entire limit order book. Let Y be the future best ask price at time $t + \Delta t$. Conditional on $Y \geq y$ where $y > 0$, the probability that $Y > y$ strongly depends upon the volume directly at the level y of the limit order book at time t. The dependance on volumes at other levels is small relative to the dependance on the volume at level y. Figure 3 demonstrates this phenomenon where the conditional movement of Y depends only locally on the current limit order book state. The conditional probability that $Y > y$ given $Y \geq y$ decreases with the volume at y.

There is some intuition regarding why the relationship in Figure 3 may hold. To reach a level $y' > y$, the sell limit orders at level y must first be consumed by buy orders. The larger the sell limit order volume at level y, the less likely the future best ask price will reach a level $y' > y$. Since we have already conditioned on $Y \geq y$, the limit orders at levels $y' < y$ are less relevant. Similarly, the event $Y > y$ requires only that the buy orders consume the sell limit orders at y, so the limit order volumes at levels $y' > y$ are less important.

The behavior of the best ask price when increasing is analogous to a "geometric random variable" whose probability of increasing from y to $y+1$ depends upon the volume at level y. The neural network architecture in Section 4.3 mimics this local behavior.

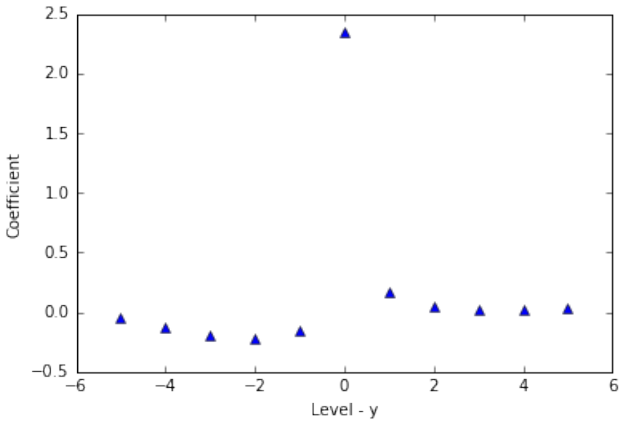

Figure 3: Coefficients from logistic regression for the probability that the future best ask price Y is greater than y conditional on $Y \geq y$ where $y > 0$; i.e., $\mathbb{P}[Y > y | Y \geq y] = \left(1 + \exp(b + \theta \cdot \text{factors})\right)^{-1}$. The current best ask price has been centered at 0 and the time horizon is 1 second. The plotted coefficients are coefficients for the limit order volumes at price levels minus y. The conditional probability that $Y > y$ given $Y \geq y$ decreases with the volume at y. The reported coefficients were fitted on the stock Amazon.

Figure (4) shows similar local structure for the stock Apple. The probability that the future best ask price $Y = y$ given that $Y \geq y > 0$ strongly depends upon the volume at level y. The larger the volume at level y,

the less likely that the entire volume at that level can be consumed by the current buying demand and thus the more likely Y equals y (and the less likely Y will move to higher levels in the limit order book). We also note that there is very little dependence on the best ask volume in Figure (4); conditional on $Y \geq y$, the volumes at previous levels $y' < y$ in the limit order book become less influential.

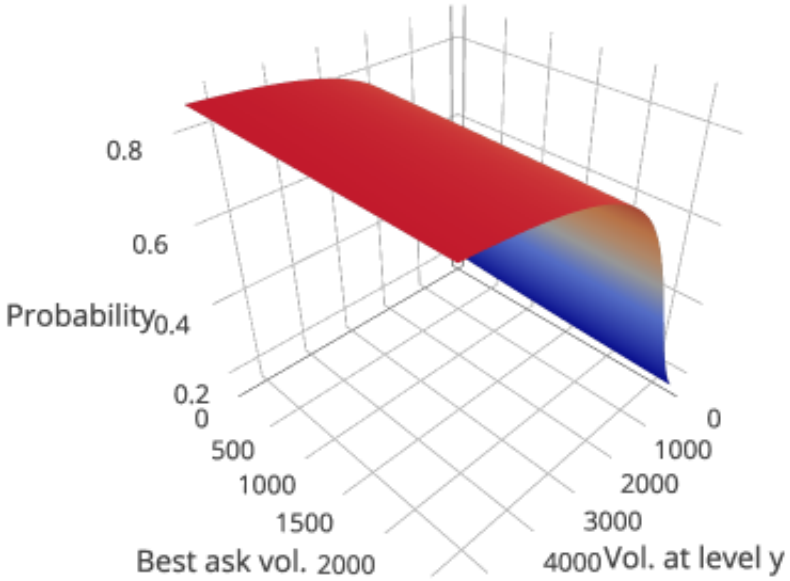

Figure 4: Probability that the future best ask price $Y = y$ given that $Y \geq y > 0$ for the stock Apple. Probability is the output of a neural network fitted to historical data for Apple and the time horizon is 1 second.

3.1 Statistical evidence across many stocks

Although Figure 3 is compelling, it is only one stock. A detailed analysis is now conducted across the entire dataset of 489 stocks. The results provide strong evidence for local spatial structure. For each stock, we perform a logistic regression similar to Figure 3. Specifically, let Y be the best ask price at $t + \Delta t$. Without loss of generality, let 0 be the current best ask price at time t and let the best ask price at time t be the frame of reference for the entire limit order book. We fit a logistic regression for:

$$\mathbb{P}[Y > y | Y \geq y] \quad = \quad \Big(1 + \exp(b + \theta \cdot (\text{volume at level } y - K, \ldots, \text{volume at level } y + K\Big)^{-1}, y > 0, \tag{1}$$

where $b \in \mathbb{R}$, $\theta = (\theta_{y-K}, \ldots, \theta_{y+K}) \in \mathbb{R}^{2K+1}$ and we choose $K = 10$. The volumes are from the current limit order book state at time t. Ask volumes are given a positive sign, while bid volumes are given a negative sign.[5] Ignoring bid volumes and performing the statistical analysis solely for ask volumes yields

[5]The probability of an increase in the best ask price Y decreases with more sell liquidity and increases with more buy liquidity, hence the opposing signs.

similar results. Volumes are from the state of the limit order book at time t. The time horizon Δt is 1 second.

We fit the logistic regression (1) for each stock in the dataset, resulting in 489 different parameter fits $\theta^1, \ldots, \theta^{489}$. Fitting is performed on the time period January 1, 2014 until May 31, 2015 (which will also be the training set used later in this paper for fitting models). For each stock j, the following "coefficient ratio" is calculated:

$$\text{Coefficient ratio for stock } j = \frac{\max_{y-p,\ldots,y+p} \theta^j_y}{\max_{y'=y-K,\ldots,y-p-1,y+p+1,\ldots,y+K} |\theta^j_{y'}|}. \tag{2}$$

The coefficient ratio (2) compares the local influence of levels close to y versus the influence of levels farther away. $\theta^j_{y'}$ is the coefficient for the volume at level y' for stock j. The larger the magnitude of the coefficient $\theta^j_{y'}$, the greater the dependance on the volume at level y'. If $p = 0$:

$$\text{Coefficient ratio for stock } j = \frac{\theta^j_y}{\max_{y'=y} |\theta^j_{y'}|}, \tag{3}$$

and the coefficient ratio measures the influence of the volume locally at level y versus the volumes at levels $y - K, \ldots, y - 1, y + 1, \ldots, y + K$. It also gives the direction of the dependance on the volume at level y. If (3) is positive, then $\mathbb{P}[Y > y | Y \geq y]$ decreases as the volume at level y increases. Table 2 gives summary statistics for the coefficient ratio (2) across all the stocks in the dataset. There is a strong dependance on the local volume at level y for the majority of stocks. The sign is also positive.

Coefficient Ratio / Quantile (%)	5	10	20	50	80	90	95
p = 0	0.84	1.47	3.38	6.43	9.89	13.20	17.69
p = 1	1.02	2.31	5.83	12.83	19.93	24.71	30.92

Table 2: Summary statistics for the coefficient ratios across all stocks.

The local dependance is strongest for more volatile stocks. For each stock, Figure 5 plots the coefficient ratio (3) for $p = 0$ versus the standard deviation of the change in the best ask price. More volatile stocks show a stronger local dependance on the volume at y. We observe the local dependance in the upper tail of the best ask price's distribution and the lower tail of the best bid price's distribution (i.e., when the best ask or best bid price moves *into the limit orders on their respective side of the book*).

3.2 A Local Model for the Upper Tail of the Best Ask

The limit order book's local structure motivates a simple local model for the upper tail of the best ask price, which will later in fact form the core of the new neural network architecture proposed in Section 4.3. Let Y be the future best ask price and let x_y be the volume at level y. Then, conditional on an increase in the best

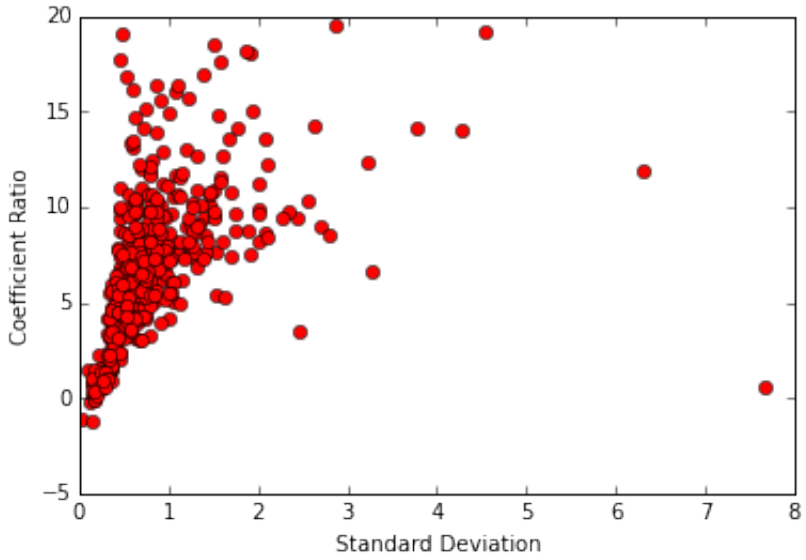

Figure 5: Plot of the coefficient ratio with $p = 0$ versus the standard deviation of changes in the best ask price for each stock. The plot has been cropped to $[0, 20] \times [0, 8]$ (some outlier data points with coefficient ratios greater than 20 are not visible in the displayed plot).

ask price (i.e., $Y > 0$), we can model the magnitude of the increase as follows:

$$\mathbb{P}[Y = y | Y \geq y] = f(x_y), \quad y > 0. \tag{4}$$

(4) completely describes the distribution of the random variable Y conditional on it increasing. It is analogous to modeling Y as a geometric random variable, but with a non-constant probability of increasing at each step. Note that $\mathbb{P}[Y > y | Y \geq y]$ is simply $1 - f(x_y)$.

Although (4) has a local dependence on the state of the limit order book (only taking as an input the volume at level y), globally the distribution of Y depends upon *all of the ask volumes*. That is,

$$\mathbb{P}[Y = y | Y > 0] = f(x_y) \prod_{y'=1}^{y-1} \Big(1 - f(x_{y'})\Big), \quad y > 0. \tag{5}$$

An alternative to (4) would be to model $\mathbb{P}[Y = y | Y > 0]$ as a function $g(y, x_1, x_2, \ldots, x_L)$ where L is the total number of levels in the limit order book. The function g is far more complex than the function f due to the high-dimensionality of the former's input. Furthermore, if the local behavior in (4) holds, (5) shows that g will depend in a nontrivial way upon all the ask volumes as it will be the composition of many f functions. Even if f takes a simple form (such as a logistic regression), the global distribution (5) will be highly nonlinear, requiring g to also be highly nonlinear.

The standard modeling approach would be to statistically estimate g. Due to the high-dimensionality of the input and the complexity of g, this can be challenging and may be prone to overfitting. The approach proposed in this paper is to directly estimate the local model f. This reduces the statistical estimation

problem to estimating a low-dimensional model with a simpler functional form, potentially leading to a more accurate estimated model. Training a model to learn f will also be faster than training a model to learn g due to the simpler nature of f in terms of both its dimension and functional form.

We will use an architecture similar to (4) for the neural network proposed in Section 4.3. It turns out such an architecture has additional important advantages such as computational efficiency, generalization over the output space, and the ability to model distributions on the entire positive real line.

4 Neural Network Architectures for Modeling Distributions on $\mathbb{R}^d$

The goal of this paper is to model a distribution on $\mathbb{R}^d$ via neural networks. Although there are many applications, we are particularly motivated by modeling the joint distribution of the best ask and best bid prices at a future time conditional on the current state of the limit order book.

In order to model a distribution on $\mathbb{R}^d$, we first discretize $\mathbb{R}^d$ into the grid $\mathcal{R}^d$ where $\mathcal{R} = \ldots, r_{-2}, r_{-1}, 0, r_1, r_2, \ldots$ and then model a distribution on the discrete space $\mathcal{R}^d$. In the context of limit order books, this discretization is exact since price levels are discrete multiples of the tick-size.

Section 4.1 reviews the standard neural network architecture for classification, which has no generalization over the space $\mathcal{R}^d$. Section 4.2 discusses a straightforward modification which allows the neural network to generalize over $\mathcal{R}^d$. However, training this neural network architecture is computationally expensive. We develop a new architecture for modeling distributions on $\mathcal{R}^d$ in Section 4.3. The proposed neural network architecture is computationally efficient, can take advantage of local spatial structure, and can be more interpretable than the aforementioned architectures.

4.1 Standard Neural Network Architecture for Classification

The basic neural network for classification is a highly nonlinear parameterized function that takes an input $x \in \mathcal{X}$ and produces a probability distribution on the finite discrete space $\mathcal{Y}$.

Given an input x, the output $f_{\theta,l}(x) \in \mathbb{R}^{d_l}$ of the l-th layer of a neural network is

$$f_{\theta,l}(x) = g^l(W_l f_{\theta,l-1}(x) + b_l), \quad l = 1, \ldots, L, \tag{6}$$

where $W_l \in \mathbb{R}^{d_l} \times \mathbb{R}^{d_{l-1}}$, $b_l \in \mathbb{R}^{d_l}$, $f_{\theta,0}(x) = x$, and $d_L = |\mathcal{Y}|$. For $l = 1, \ldots, L-1$, the nonlinear transformation $g^l(z) = \big(\sigma(z_1), \ldots, \sigma(z_{d_l})\big)$ for $z \in \mathbb{R}^{d_l}$ and $z_1, \ldots, z_{d_l} \in \mathbb{R}$. The function σ is nonlinear; typical choices are sigmoidal functions, tanh, rectified linear units (ReLU), and clipped rectified linear units. The function g^L for the final layer L is the softmax function:

$$g(z) = \Big(\frac{e^{z_1}}{\sum_{i=1}^{d_L} e^{z_i}}, \ldots, \frac{e^{z_{d_L}}}{\sum_{i=1}^{d_L} e^{z_i}}\Big), \quad z \in \mathbb{R}^{d_L}. \tag{7}$$

The final output of the neural network $f_{\theta,L}(x)$ is a probability distribution on $\mathcal{Y}$ conditional on the features x. The parameters collectively are $\theta = (W_1, \ldots, W_L, b_1, \ldots, b_L)$, where L is the number of layers

in the neural network. The objective is to choose the parameters θ such that the log-likelihood $\mathcal{L}$ of the neural network's output $f_{\theta,L}$ is maximized for the data.

Let the data be $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ where $(x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$. Then, the normalized log-likelihood of the data for the neural network model is

$$\mathcal{L}(\mathcal{D}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{y \in \mathcal{Y}} \mathbf{1}_{y=y_n} \log f_{\theta,L}^{y}(x_n), \tag{8}$$

where $f_{\theta,L}^{y}$ is the y-th element of the vector $f_{\theta,L}$.

The complexity of the model is determined by both the number of layers ("depth") and the number of neurons $(d_1, \ldots, d_L)$ in each layer. Although the dividing line is somewhat arbitrary, neural networks are typically considered deep if there are three or more hidden layers ($L \geq 4$). Equation (6) describes the basic neural network achitecture, and there are several popular modifications to the architecture of the layers $1, \ldots, L-1$ (e.g., convolution neural networks). The discussion below is also applicable to these other architectures.

A potential drawback of the standard neural network for classification is that, although it generalizes over the input space $\mathcal{X}$, it does not allow for generalization over the output space $\mathcal{Y}$. As mentioned earlier, one approach to modeling a distribution on $\mathbb{R}^d$ is to discretize $\mathbb{R}^d$ into the grid $\mathcal{R}^d$ where $\mathcal{R} = \ldots, r_{-2}, r_{-1}, 0, r_1, r_2, \ldots$ and then model a distribution on the discrete space $\mathcal{R}^d$. Many problems may have some spatial structure where the event $r \in \mathcal{R}^d$ will be strongly related to the event $r' \in \mathcal{R}^d$ if r and r' are close in distance. A training sample at r should then allow one to learn about both r and r'. However, the standard neural network architecture for classification would regard r and r' as completely separate events, failing to take advantage of any available spatial structure since it has no generalization over space.

One glaring case where the standard neural network for classification fails due to a lack of generalization is when $r_k = k\Delta r$ and Δr is small. For a dataset with N samples, the fraction of grid points with at least one data sample tends to zero as $\Delta r \to 0$. Consequently, the trained neural network will predict that events at the vast majority of the grid points in $\mathcal{R}$ will occur with probability zero. A model which generalizes over the output space can avoid this pitfall. As a simple example, consider fitting a density to i.i.d. samples from a continuous random variable. In this case, there are no features (i.e., explanatory variables); formally, one can just replace the feature vector with a vector of zeros for every sample. The neural network will then trivially give the empirical measure of the observed samples, which will be zero at many grid points if Δr is small. This is a very bad statistical estimate since the samples come from a random variable with a continuous density. Instead, one should estimate a smoothed density from the samples; the new neural network architecture developed in Section 4.3 is able to do this.

Generalizing over space is especially useful in the tails of the distribution where less data is available. The tails of the distribution, although associated with less frequent events, are important for risk analysis since they represent extreme events which can have a disproportionate impact. Generalization over space

also helps to combat the curse of dimensionality. The number of grid points grows exponentially with the dimension d, meaning there is less data per grid point (and less data per state $y \in \mathcal{Y}$). This can be a source of overfitting.

There are other disadvantages to applying the standard neural network to modeling spatial distributions. Since $\mathcal{Y}$ is a finite discrete space, $\mathcal{R}$ must be truncated to cover only a finite region of space, which may not be desirable. Secondly, even if the bulk of the events occur in a small region of space, probabilities may still be needed at a large number of spatial points, incurring significant computational cost. For instance, even if 99% of events occur in $[0, 1]$, and the rest are uniformly spread across $[-1000, 1000]$, probabilities at all (discretized) spatial points must be calculated for each data sample during training and prediction. This incurs significant computational cost and thus slower training rates, especially in higher dimensions $d > 1$ where the number of grid points grows rapidly with d.

4.2 Straightforward modification to allow generalization

There is a straightforward modification to create a neural network which generalizes over space. This modification has been studied before; for instance, see Likas (2001). Let $f_\theta(x, y) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be the unnormalized log-probability of the event y conditional on the feature x, where $f_\theta(x, y)$ is a neural network with inputs (x, y). The probability of y conditional on the feature x is

$$\frac{e^{f_\theta(x,y)}}{\sum_{y' \in \mathcal{Y}} e^{f_\theta(x,y')}}. \tag{9}$$

Due to the continuity of f_θ, the probabilities (conditional on the feature x) of y_1 and y_2 will be close if the distance between y_1 and y_2 is small.

(9) can be computationally expensive. For each training sample, $f_\theta(x, y)$ and its gradient must be evaluated at every $y \in \mathcal{Y}$. If the number of training samples is N, this is comparable to training a standard neural network for binary classification on $N \times |\mathcal{Y}|$ training samples. For instance, if $\mathcal{Y}$ is a Cartesian grid covering $\mathbb{R}^3$ with $1,000$ grid points in each direction, $|\mathcal{Y}| = 1$ billion.

A second disadvantage is that (9) cannot model distributions on $\mathcal{R}^d$ but instead must truncate the space in order to form a finite grid.

4.3 A computationally efficient architecture for modeling spatial distributions

First consider a distribution on $\mathbb{R}_+ = (0, \infty)$, which is discretized into $\mathcal{R}_+ = r_1, r_2, \ldots$. Later this is extended to the more general case of $\mathcal{R}^d$. Let $f_\theta(x, y) : \mathcal{X} \times \mathbb{R} \to \mathbb{R}$ be a neural network. Provided certain technical conditions,[6] the distribution of a random variable $Y \in \mathcal{R}_+$ conditional on the random variable

[6]It is required that $\sum_{y \in \mathcal{R}_+} \mathbb{P}[Y = y | X = x] = 1$ for any x. A sufficient condition is that the hidden units for the neural network $f_\theta(x, y)$ are bounded (e.g., sigmoidal, tanh, or clipped ReLU). Proving that this technical condition holds for ReLU (where $f_\theta(x, y)$ is unbounded in y) requires further analysis.

$X \in \mathcal{X}$ is completely specified by the following model:

$$\mathbb{P}[Y = y \big| Y \geq y, X = x] = \frac{e^{f_\theta(x,y)}}{1 + e^{f_\theta(x,y)}}. \tag{10}$$

(10) is analogous to the first arrival time of a nonhomogenous Poisson process in continuous time. The log-likelihood for a training sample (x, y) is

$$\mathcal{L}(\{(x,y)\}) = \log\left(\frac{e^{f_\theta(x,y)}}{1 + e^{f_\theta(x,y)}}\right) + \sum_{y' \in \mathcal{R}_+ : y' < y} \log\left(\frac{1}{1 + e^{f_\theta(x,y')}}\right) \tag{11}$$

The architecture (10) has two advantages over (9). The first is that the neural network $f_\theta(x, y)$ and its gradient need to be evaluated at far fewer grid points. For each sample (x, y), (10) only needs to be evaluated up until y whileas (9) needs to be evaluated on the entire grid. Secondly, (10) can model the entire space $\mathcal{R}_+$; there is no need to form a truncated grid as in (9).

4.3.1 Extension to $\mathcal{R}^d$

(10) can be extended to model distributions on $\mathcal{R}^d$. Let $Y = (Y_1, \ldots, Y_d) \in \mathcal{R}^d$ and have the conditional distribution:

$$\begin{aligned}
\mathbb{P}[Y = (y_1, \ldots, y_d) \big| X = x] &= \mathbb{P}[Y_1 = y_1 \big| X = x] \prod_{i=2}^{d} \mathbb{P}[Y_i = y_i \big| Y_{0:i-1} = y_{0:i-1}, X = x], \\
\mathbb{P}[Y_1 = y_1 \big| X = x] &= g_\theta^1(x, y_1), \\
\mathbb{P}[Y_i = y_i \big| Y_{0:i-1} = y_{0:i-1}, X = x] &= g_\theta^i(x, y_{0:i-1}, y_i),
\end{aligned} \tag{12}$$

The conditional distributions $g^1, \ldots, g^d$ will be functions of neural networks, which will be specified shortly. Note that the framework (12) avoids the curse of dimensionality for large d since the computational expense of the log-likelihood grows linearly with d:

$$\mathcal{L}(\{(x,y)\}) = \log\left(g_\theta^1(x, y_1)\right) + \sum_{i=2}^{d} \log\left(g_\theta^i(x, y_{0:i-1}, y_i)\right). \tag{13}$$

The conditional distribution of Y_1 conditional on X is completely specified by:

$$\begin{cases}
\mathbb{P}[Y_1 = y_1 \big| Y_1 \geq y_1, X = x] = \frac{e^{f_\theta^{1,+}(x,y_1)}}{1 + e^{f_\theta^{1,+}(x,y_1)}} & y_1 \geq 1 \\
\mathbb{P}[Y_1 = z | X = x] = h_\theta^{1,z}(x) & z \in \{y_1 > 0\}, \{y_1 = 0\}, \{y_1 < 0\} \\
\mathbb{P}[Y_1 = y_1 \big| Y_1 \leq y_1, X = x] = \frac{e^{f_\theta^{1,-}(x,y_1)}}{1 + e^{f_\theta^{1,-}(x,y_1)}} & y_1 \leq -1
\end{cases}$$

$f_\theta^{1,-} : \mathcal{X} \times \mathbb{R} \to \mathbb{R}$ and $f_\theta^{1,+} : \mathcal{X} \times \mathbb{R} \to \mathbb{R}$ are neural networks. The neural network $h_\theta^1(x)$ is a standard neural network for classification (as described in Section 4) which produces a vector of three probabilities for the events $\{y_1 > 0\}, \{y_1 = 0\}, \{y_1 < 0\}$, and $h_\theta^{1,z}(x)$ is the z-th vector element. The standard neural network for classification h_θ^1 is required to "stitch" together $\mathcal{R}_+$ and $\mathcal{R}_- = \ldots, r_{-2}, r_{-1}$.

Similarly, the conditional distribution of Y_i conditional on $(Y_{0:i-1}, X)$ for $i \geq 2$ is completely specified by:

$$\begin{cases} \mathbb{P}[Y_i = y_i | Y_i \geq y_i, Y_{0:i-1} = y_{0:i-1}, X = x] = \frac{e^{f_\theta^{i,+}(x, y_{0:i-1}, y_i)}}{1 + e^{f_\theta^{i,+}(x, y_{0:i-1}, y_i)}} & y_i \geq 1 \\ \mathbb{P}[Y_i = z | Y_{0:i-1} = y_{0:i-1}, X = x] = h_\theta^{i,z}(x, y_{0:i-1}) & z \in \{y_i > 0\}, \{y_i = 0\}, \{y_i < 0\} \\ \mathbb{P}[Y_i = y_i | Y_i \leq y_i, Y_{0:i-1} = y_{0:i-1}, X = x] = \frac{e^{f_\theta^{i,-}(x, y_{0:i-1}, y_i)}}{1 + e^{f_\theta^{i,-}(x, y_{0:i-1}, y_i)}} & y_i \leq -1 \end{cases}$$

Example 4.1 (Limit Order Book)**.** *Modeling the best ask and best bid prices at a future time conditional on the current state of the limit order book is equivalent to modeling the change in the best ask and best bid prices. We measure the change by the number of levels that the best ask and best bid prices move.*

$$(Y_1, Y_2) = (\textit{change in best ask price}, \textit{change in best bid price}) \in (\ldots, -2, -1, 0, 1, 2, \ldots)^2.$$

The neural network h^1 predicts whether the best ask price will increase, decrease, or stay the same. If h^1 predicts that the best ask increases, $f^{1,+}$ predicts how many levels it will increase. If h^1 predicts that the best ask decreases, $f^{1,-}$ predicts how many levels it will decrease. h^2, $f^{2,+}$, and $f^{2,-}$ play similar roles for the best bid price.

4.3.2 Advantages of the spatial neural network

There are several potential advantages to this proposed architecture for the spatial neural network. The model and its gradient can be evaluated at far fewer grid points in the computationally efficient architecture. Secondly, the proposed architecture can model the entire space $\mathcal{R}^d$; there is no need to form a truncated grid as in standard architectures. One disadvantage is that the architecture is composed of several neural networks instead of a single neural network. The number of neural networks grows linearly with d.

The proposed architecture can also take advantage of "local spatial structure", if it exists in the application setting. The spatial neural network (10) is local in nature; it models the local dynamics within a small region in space. The spatial neural network (10) can leverage a priori knowledge that conditional on Y being in some region of space, the local dynamics of Y in that region only depend a particular subset of the values in the vector X. This can improve performance and increase interpretability since it reduces the dimension of the input space. For example, if X is a vector containing information at locations in $\mathbb{R}^d$ and $Y \in \mathbb{R}^d$, Y's local dynamics in some small region of $\mathbb{R}^d$ may only depend upon information at locations close to that small region. This is arguably the case for limit order books; see Section 3. Such local behavior can be naturally modeled by $f_\theta(x, y)$ in (10). Let $f_\theta(x, y) = g_\theta(m(x, y), y)$ where g_θ is a neural network and $m(x, y)$

is a map taking the vector of limit order volumes at all levels and outputting a smaller vector of limit order volumes at only levels close to y. Although the local distribution of Y conditional on Y being in a particular region depends only on a subset of X, the global distribution of Y still depends upon the entire variable X. In the limit order book setting where $y = (y_1, y_2) = (\text{change in best ask}, \text{change in best bid})$, the map $m(x, y)$ would output a vector of limit order volumes at levels close to $(y_1 + \text{current best ask}, y_2 + \text{current best bid})$.

The spatial neural network's compoutational cost only grows linearly with the dimension d due to the dimension splitting in (12). However, the trick (12) is not unique to the spatial neural network and can also be used with the standard neural network.

4.4 Other approaches to modeling spatial distributions

Another approach to modeling spatial distributions is to use Gaussian mixtures and model the parameters (means, covariances, and mixture weights) with neural networks. This produces a continuous distribution on $\mathbb{R}^d$. Various frameworks combining Gaussian mixtures with neural networks have been proposed by Variani, McDermott & Helgold (2015), Demuynck & Triefenbach (2013), Paulik (2013), van den Oord & Schrauwen (2014), Yu & Seltzer (2011), Sainath, Kingsbury & Ramabhadran (2012), and Deng & Chen (2014).

Gaussian mixture models are not suitable for the limiting order book setting. The distribution of the best ask and best bid prices does not have a density since the best ask and best bid prices take values at discrete levels. A Gaussian mixture model would converge during training to a mixture of Gaussians with zero variances, meaning there's no advantage over the standard neural network architecture which models distributions on a discrete space. Numerical difficulties may also emerge as the variances become small. In other applications outside of limit order books, Gaussian mixture models may have an advantage over the "spatial neural network" developed in this paper when the disribution is smooth and its tails are close to Gaussian. It should also be emphasized that Gaussian mixture models produce an actual density while the architectures in this paper require first discretizing space.

The spatial neural network proposed in this paper has some other advantages over Gaussian mixture models. Gaussian mixture models may require a large number of Gaussians to model sharp (or discontinuous) changes. A large number of Gaussians may also be needed if the tail is not Gaussian. Gaussian mixtures cannot model local spatial structure. Finally, as mentioned in the previous paragraph, Gaussian mixtures are not suitable for distributions with delta functions.

5 Model Training

The neural networks are trained using approaches from deep learning, which we describe in Section 5.1. The computational implementation using GPU clusters is outlined in Section 5.2. The division of the dataset into training, validation, and test sets is specified in Section 5.3.

5.1 Deep Learning

We use 4 layers for the neural networks. Neural networks with 3 or more hidden layers are referred to as "deep neural networks". A neural network with more layers has lower bias, but is also more difficult to train and more prone to overfitting (i.e., high variance). Recent research has developed many new methods for training deep neural networks, and we employ several of these techniques. We use dropout to prevent overfitting (Srivastava, Hinton, Krizhevsky, Sutskever & Salakhutdinov 2014). Batch normalization is used between each hidden layer to prevent internal covariate shift (Ioffe & Szegedy 2015). Stochastic gradient descent with momentum is used for training (Sutskever, Martens, Dahl & Hinton 2013). The learning rate is adaptive, decreasing by a constant factor whenever the training error increases over a training epoch. Early stopping via a validation set is imposed to reduce overfitting (Bengio 2012). Although ReLU units have often produced the best performance for deep neural networks, it may be preferable in the limit order book setting to use hidden units which are bounded (e.g., clipped ReLU, sigmoidal, or tanh). The limit order volumes are unbounded, and a small fraction have very large values. These outlier values can cause undesirably large gradient steps.

In order to make the comparison between the standard neural network architecture and the spatial neural network as fair as possible, we apply the methods above in exactly the same manner when training both of the neural networks. More discussion is provided in Section 6.

5.2 Computational Approach

Due to the size of the dataset and the large number of parameters in the neural networks, model training is computationally expensive. To address this, we use 20 GPUs to train the models. Accessing and processing data is accelerated via distributed storage. Pre-processing of data is performed using parallelization over 150 vCPUs on multiple high-performance multi-core processors.

Model training is parallelized across 20 GPUs. Each GPU itself has 1,500 CUDA cores. GPUs allow massive parallelization via the large number of cores and have become the preferred approach for neural network training. We also use NVIDIA's cuDNN library, which is a highly optimized library of primitives for training deep neural networks on GPUs.

Filtering the original raw data to create datasets for model training is itself very challenging. The original dataset contains roughly 50 terrabytes of raw data. Distributed storage across multiple devices allows us to access data at a higher aggregate throughput. Data processing is parallelized across 5 compute-optimized Intel Xeon E5-2666 v3 Haswell processors. Each processor has 36 vCPUs, for a total of 180 vCPUs. Each processor has 4,000 Mbps throughput.

In data case [1] (fixed time horizon of 1 s), each stock has roughly 10 million samples. Over the entire dataset of 489 stocks, this makes for 5 billion data samples in total. In data case [2] (random time horizon at which the next change in the best ask or best bid prices occurs), each stock has on average 5 million samples. Over the entire dataset, this amounts to 2.5 billion data samples. Each sample contains a vector of length 200, recording the state of the limit order book across the first 50 bid and ask levels.

5.3 Division of Data into Training, Validation, and Test Sets

The data is divided into three sets. The test set is all data from June 1, 2015 to August 31, 2015. The training data is composed of 95% of the data from January 1, 2014 to May 31, 2015 (drawn at random). The validation set is the remaining 5% of the data from January 1, 2014 to May 31, 2015. Models are trained and tested separately on each stock; i.e., a new randomly initialized model is trained for each stock.

6 Out-of-sample Results

Out-of-sample results for Case [1] and Case [2] are reported in Sections 6.1 and 6.2. Case [1] is a fixed time horizon of 1 second and Case [2] is a random time horizon at the next price move. (See Section 2 for a detailed description.) For both Cases [1] and [2], out-of-sample performance is reported for the distribution of the best ask price as well as the joint distribution of the best ask and best bid prices. The spatial neural network outperforms the standard neural network for classification with lower error, higher accuracy, and faster training times.

Neural network results are also compared against baseline models. The first baseline model is the naive empirical model, which is simply the naive empirical distribution from the training set. If models do not have lower errors than the naive empirical model, then the limit order book contains no information on future movements of the best ask and best bid prices. The second baseline model is a logistic regression whose input includes nonlinear features (the order imbalances). The input also includes the volumes at the different levels. If the neural network has lower error than the logistic regression, this indicates that limit order book dynamics have significant nonlinearity beyond the nonlinearity of the order imbalance feature. Both of the neural networks outperform these baseline models.

In Sections 6.1 and 6.2, both the standard neural network and spatial neural network have 4 layers with 50 neurons in each hidden layer. The deep learning techniques discussed in Section 5.1 are applied in the same manner to the standard neural network and spatial neural network. In addition, all models use the same batchsize, initial learning rate, momentum, and dropout rate. The initial parameters are randomly initialized for all models. All models are trained and tested separately on each stock; i.e., models $\mathcal{M}_i^1, \mathcal{M}_i^2, \ldots, \mathcal{M}_i^{489}$ are trained where $i \in \{\text{naive empirical model}, \text{logistic regression}, \text{standard neural network}, \text{spatial neural network}\}$. The model $\mathcal{M}_i^j$ is trained only on the training set for stock j and $\mathcal{E}_i^j$ is the out-of-sample error of the model $\mathcal{M}_i^j$ on the test set for stock j. A new random initialization of the model parameters is used for each stock. Although the results reported in Sections 6.1 and 6.2 are for a single set of hyperparameter choices for the batchsize, initial learning rate, momentum, and dropout rate, we did test a wide range of values for these hyperparameters on a small subset of stocks and found that results are robust to the choice of these hyperparameters.

Modeling the best ask and best bid prices at a future time is equivalent to modeling the change in the best ask and best bid prices; see Example 4.1. We measure the change by the number of levels that the best ask and best bid prices move. Since the standard neural network cannot model the entire real line, $\mathcal{R}$ is

truncated to $-50, -49, \ldots, 49, 50$ for model comparison purposes. The "error" reported is the cross-entropy error, which is equivalent to the negative log-likelihood.

The dimension splitting trick (12) is applied to the standard neural network and logistic regression when modeling the joint distribution of the best ask and best bid prices. This is done for two reasons. First, this makes the other models consistent with the spatial neural network. Secondly, without using (12), the number of output states becomes $|\mathcal{R}|^2 = 10,201$ and the standard neural network's convergence during training is so slow that this approach becomes infeasible even with the large amount of computational resources. Similarly, the logistic regression training also becomes very slow.

Figure 6 shows that the spatial neural network performs better in comparison with the standard neural network for more volatile stocks. This observation matches the initial intuition motivating the spatial neural network: distributions spread over wider ranges of $\mathbb{R}^d$ will be more challenging for the traditional neural network designed for classification.

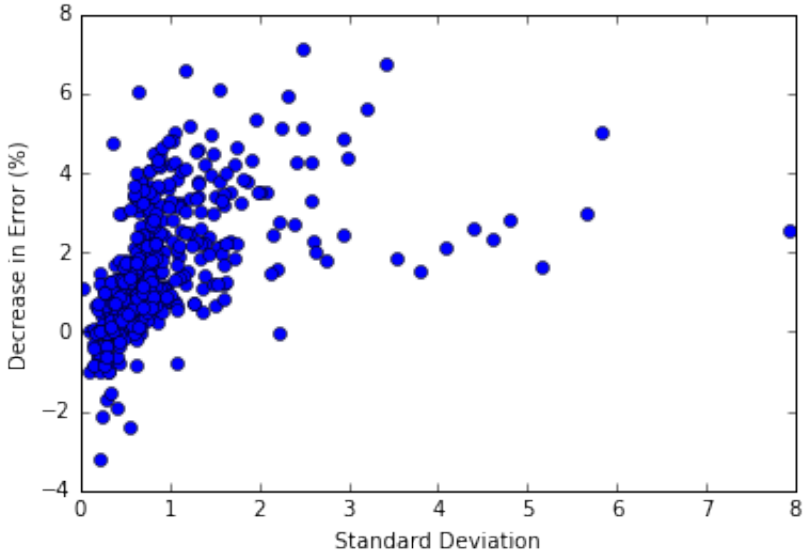

Figure 6: Percent decrease in error for spatial neural network versus standard neural network plotted against the standard deviation of the change in the best ask price (for Case [1]). The change in the best ask price is the number of levels which the best ask price has moved over the 1-second time interval. The errors compared here are from the case for the joint prediction of the best ask and best bid prices.

The logistic regression fairs very poorly when used to model the joint distribution. This is because the conditional distribution of the best bid price given the change in the best ask price is a nonlinear function. For instance, consider the probability that the best bid price does not change conditional on a change in the best ask price. If the best ask price decreases, the best bid price is more likely to decrease. If the best ask price increases, the best bid price is more likely to increase. Thus, the probability that the best bid price remains the same is highest when the best ask price does not change and decreases when the best ask price moves up or down. The conditional probability of the best bid price is then a nonlinear function of the

change in the best ask price.

6.1 Case [1]: Fixed Time Horizon of 1 second

Out-of-sample performance of the models is reported for the prediction case of a fixed time horizon of 1 second. The models seek to predict the joint distribution of the best ask and best bid prices at time $t + \Delta t$ given the current state of the limit order book at time t.

Tables 3 and 4 compare the out-of-sample performance of the different models for the marginal distribution of the best ask price. Tables 5 and 6 compare the out-of-sample performance of the different models for the *joint distribution* of the best ask and best bid prices. The neural networks consistently have lower error than the naive empirical model and the logistic regression. The spatial neural network consistently has lower error than the standard neural network.

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	1/489	1/489	1/489
Logistic Reg.	488/489	NA	64/489	2/489
Neural Net.	488/489	425/489	NA	99/489
Spatial Neural Net.	488/489	487/489	390/489	NA

Table 3: Number of stocks out of 489 total stocks where Model 1 has a lower out-of-sample error than Model 2: $\frac{1}{489}\sum_{j=1}^{489} \mathbf{1}_{\mathcal{E}^j_{\text{Model 1}} < \mathcal{E}^j_{\text{Model 2}}}$. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	-5.84	-8.33	-9.63
Logistic Reg.	5.45	NA	-2.33	-3.59
Neural Net.	7.56	2.25	NA	-1.25
Spatial Neural Net.	8.71	3.45	1.20	NA

Table 4: Average percent decrease in error for Model 1 versus Model 2: $\frac{1}{489}\sum_{j=1}^{489} \frac{\mathcal{E}^j_{\text{Model 2}} - \mathcal{E}^j_{\text{Model 1}}}{\mathcal{E}^j_{\text{Model 2}}} \times 100\%$. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

6.2 Case [2]: Random Time Horizon at the Next Change of Best Bid or Ask Prices

Out-of-sample performance of the models is reported for the prediction case of the next change of the best bid or best ask prices. The models seek to predict the joint distribution of the best ask and best bid prices

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	220/489	1/489	1/489
Logistic Reg.	269/489	NA	2/489	0/489
Neural Net.	488/489	487/489	NA	50/489
Spatial Neural Net.	488/489	489/489	439/489	NA

Table 5: Number of stocks out of 489 total stocks where Model 1 has a lower out-of-sample error than Model 2. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	0.49	-12.53	-14.27
Logistic Reg.	-0.41	NA	-12.00	-13.74
Neural Net.	10.79	10.40	NA	-1.63
Spatial Neural Net.	12.25	11.87	1.57	NA

Table 6: Average percent decrease in error for Model 1 versus Model 2. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

upon the *next price move*. The next price move is defined as the first time at which the best bid price or best ask price changes.

Tables 7 and 8 compare the out-of-sample performance of the different models for the marginal distribution of the best ask price. Tables 9 and 10 compare the out-of-sample performance of the different models for the *joint distribution* of the best ask and best bid prices. The neural networks consistently have lower error than the naive empirical model and the logistic regression. The spatial neural network consistently has lower error than the standard neural network.

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	3/489	0/489	0/489
Logistic Reg.	486/489	NA	119/489	0/489
Neural Net.	489/489	370/489	NA	23/489
Spatial Neural Net.	489/489	489/489	466/489	NA

Table 7: Number of stocks out of 489 total stocks where Model 1 has a lower out-of-sample error than Model 2: $\frac{1}{489}\sum_{j=1}^{489}\mathbf{1}_{\mathcal{E}^j_{\text{Model 1}}<\mathcal{E}^j_{\text{Model 2}}}$. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	-13.18	-24.36	-30.85
Logistic Reg.	11.51	NA	-9.87	-15.62
Neural Net.	18.57	7.99	NA	-5.69
Spatial Neural Net.	23.07	13.06	5.11	NA

Table 8: Average percent decrease in error for Model 1 versus Model 2: $\frac{1}{489}\sum_{j=1}^{489}\frac{\mathcal{E}^j_{\text{Model 2}}-\mathcal{E}^j_{\text{Model 1}}}{\mathcal{E}^j_{\text{Model 2}}}\times 100\%$. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	482/489	0/489	0/489
Logistic Reg.	7/489	NA	0/489	0/489
Neural Net.	489/489	489/489	NA	9/489
Spatial Neural Net.	489/489	489/489	480/489	NA

Table 9: Number of stocks out of 489 total stocks where Model 1 has a lower out-of-sample error than Model 2. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

Model 1/Model 2	Naive empirical model	Logistic Reg.	Neural Net.	Spatial Neural Net.
Naive empirical model	NA	13.43	-36.95	-46.21
Logistic Reg.	-16.35	NA	-60.08	-70.68
Neural Net.	24.77	34.65	NA	-7.52
Spatial Neural Net.	30.26	39.51	6.68	NA

Table 10: Average percent decrease in error for Model 1 versus Model 2. "Neural Net." is the standard neural network architecture described in Section 4.1. "Spatial Neural Net." is the computationally efficient neural network architecture for spatial distributions developed in Section 4.3.

6.3 Out-of-sample Accuracy

The cross-entropy error (i.e., the negative log-likelihood) is the best metric to evaluate model performance since it measures how well the model fits the empirical distribution of the data. However, it does lack some intuition in the sense that it is unclear how practically significant a reduction of 1% in cross-entropy error is. A more interpretable metric is the *accuracy* of the model. Model accuracy is the percentage of the time where the model correctly predicts the outcome. The predicted outcome is taken as the most likely event according to the model-produced distribution. In this section, we report the accuracy of neural networks in the limit order book setting.

In some settings, such as image classification, accuracy is an extremely good metric which in practice closely coincides with the cross-entropy error. However, in general, this may not be the case and we caution that accuracy may be an imperfect metric for many financial applications. Financial applications typically

have a large amount of noise; modeling the distribution of the noise is just as important as modeling the most likely outcome. A simple example is the prediction of the binary event $Y \in \{0, 1\}$ where the true probability of event $Y = 1$ is $\frac{99}{100}$. The two models $\mathbb{P}[Y = 1] = \frac{51}{100}$ and $\mathbb{P}[Y = 1] = \frac{98}{100}$ both have the same accuracy (99 %). However, the second model is clearly superior and has a much smaller cross-entropy error. Nonetheless, accuracy is an easily interpreted metric and thus can be worthwhile examining.

Figures 7 and 8 compare the out-of-sample accuracy of the naive empirical model, logistic regression, and standard neural network for Case [2]. Figures 7 and 8 are histograms for the increase in accuracy of the standard neural network over the naive model and logistic regression, respectively. Accuracies are measured in percent (i.e., if the neural network has an accuracy of 60% and the naive model has an accuracy of 51%, the increase in accuracy is 9%). The neural network offers signficant improvement over both logistic regression and the naive empirical model.

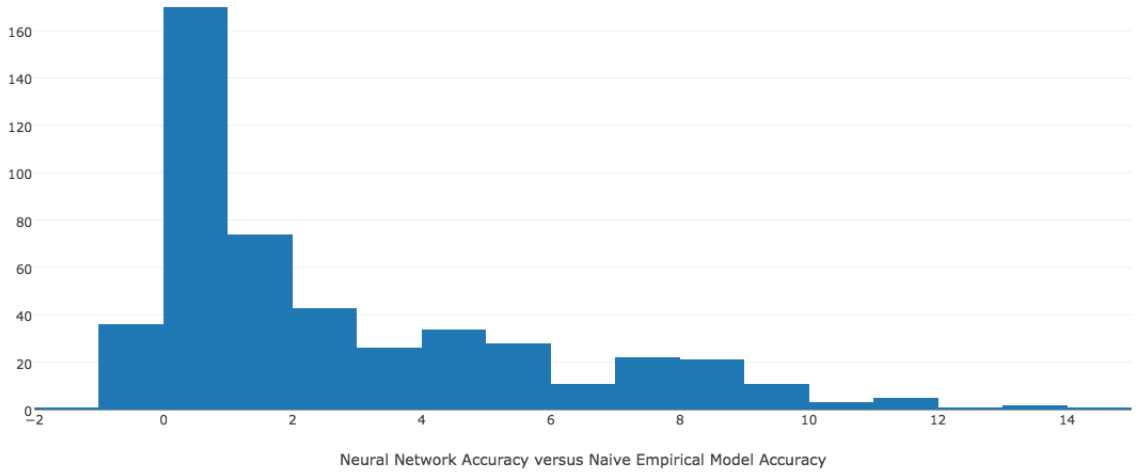

Figure 7: Increase in accuracy of neural network over naive empirical model. Accuracies are measured in percent.

The difference between the standard neural network and the spatial neural network is in the tails of the distribution. Conditional on the best ask price moving, how far does it move? We examine the *top-k accuracy* for the upper tail of the best ask price. We define a model's top-k accuracy as the percent of time the actual outcome is in the model's top k most likely outcomes. The top-1 accuracy is simply the model's accuracy. Table 11 reports the top-k accuracy for the different models conditional on the best ask price increasing. The spatial neural network outperforms the standard neural network. Table 12 directly compares the top-k accuracy of the spatial neural network and the standard neural network by reporting the fraction of stocks where the spatial neural network's top-k accuracy is greater than the standard neural

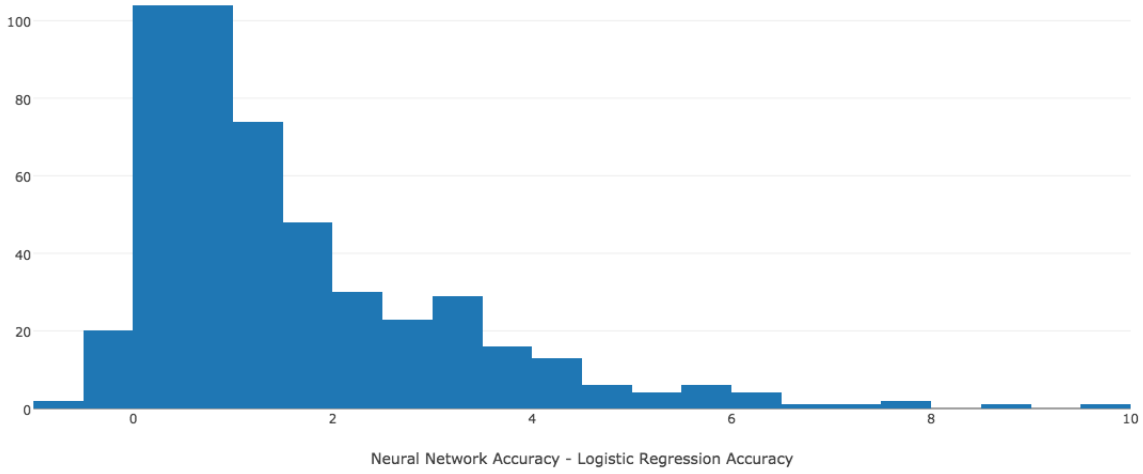

Figure 8: Increase in accuracy of neural network over logistic regression. Accuracies are measured in percent.

k/Model	Naive empirical model	Logistic Regression	Neural Net.	Spatial Neural Net.
1	59.92	60.25	62.50	68.17
2	76.55	76.84	78.80	83.61
3	83.82	84.07	85.65	89.72
4	88.02	88.22	89.49	92.89
5	90.96	91.13	92.06	94.78
6	92.64	92.70	93.49	95.93
7	93.75	93.85	94.53	96.74
8	94.62	94.78	95.36	97.34
9	95.43	95.54	96.02	97.80
10	96.14	96.18	96.58	98.16

Table 11: Average top-k accuracy (in %) for different models conditional on the best ask price increasing. Top-k accuracy is the percent of time the actual outcome is in the model's top k most likely outcomes.

network's top-k accuracy conditional on the best ask price increasing.

k	Spatial Neural Net. vs. Neural Net.
1	88.5
2	88.5
3	87.1
4	88.5
5	90.0
6	95.7
7	98.6
8	97.8
9	97.8
10	99.3

Table 12: Percent of stocks where the spatial neural network's top-k accuracy is greater than the standard neural network's top-k accuracy conditional on the best ask price increasing. Top-k accuracy is the percent of time the actual outcome is in the model's top k most likely outcomes.

A List of Stocks

LLTC, JD, NVDA, COG, BBBY, HAS, BRK.B, AES, ADT, HRS, GILD, ABBV, BA, ALXN, ALKS, HOG, BCR, AAL, CTSH, HON, AIZ, INTC, YHOO, HOLX, ILMN, INCY, BBT, BXP, CAM, BIIB, KLAC, ABT, LVNTB, HRL, BSX, CHKP, AGN, NXPI, BAX, CVC, BBY, CHTR, TSLA, HPQ, BHI, APD, IBKR, HCA, CBS, ISRG, ALLE, CTXS, BWA, ACE, ACN, ADBE, ADP, CCL, GWW, CPB, BMY, HIG, CA, AIV, HAL, BRCM, ADS, CBG, BDX, SCTY, LMCA, AA, HSIC, AMGN, ADI, AMZN, HAR, ADM, AVY, NFLX, MXIM, CAT, MSFT, ENDP, MAR, COF, AAP, NTES, BLL, FOXA, AEE, DISH, CTRP, KMX, AMAT, HCP, HBI, CHRW, BK, ATVI, BAC, ADSK, AAPL, CAH, HRB, GOOGL, BMRN, FB, GT, MKC, QCOM, MET, NDAQ, SBUX, STT, NOV, MYL, SYF, TGT, JWN, SNI, PNR, MHFI, PH, TEL, MCD, NLSN, MCO, MS, MUR, ORCL, SYK, SLB, PCAR, PCL, SO, OI, PPG, NFX, NI, SIG, SE, PRGO, SPLS, TAP, PKI, RL, PBI, PDCO, PXD, MCHP, PCG, ROST, MCK, SEE, PSX, MON, SWN, R, STI, MOS, RCL, SLG, CRM, PPL, MRK, MAT, MSI, PNC, PBCT, SCG, NEM, PNW, PM, PEP, SNA, NKE, SYY, MA, TE, NTAP, NAVI, SHW, PFE, POM, SWK, MJN, SPG, COL, SRE, ROP, HOT, MNST, SJM, STJ, PAYX, NEE, NWSA, OKE, MHK, MDT, NWL, CSC, DISCA, LEG, DVN, HES, PGR, OMC, MMC, COH, DUK, JPM, DG, CNP, DFS, GMCR, MAC, CCI, CMG, CVS, MTB, DRI, CCE, DE, ICE, MAS, DGX, PEG, CSX, JBHT, DNB, COP, COST, DD, MPC, DISCK, DHR, DLTR, CLX, CELG, CL, CMS, CB, LB, PG, KR, HD, LH, HBAN, PFG, LLL, PVH, CMA, KEY, HP, KIM, CVX, PWR, LUK, OXY, DHI, JNPR, D, ITW, DOV, DO, CME, CMI, LEN, PHM, DPS, KSU, CMCSA, DOW, O, KMB, PRU, CTAS, PLD, MRO, LOW, DTE, CINF, L, M, K, C, CHK, LNC, CF, LM, CI, MLM, HST, CERN, LVLT, DAL, LYB, LLY, JNJ, KSS, CAG, KMI, IR, PSA, CTL, DLPH, LMT, DVA, IBM, ATML, TMO, NRG, TXN, UAL, YUM, TWX, HSY, HCN, WEC, VAR, UHS, VLO, NSC, WYN, FLS, WDC, UNH, FLIR, WU, FE, THC, ZTS, ZION, FIS, TSO, TRV, RIG, ECL, A, WM, TIF, WAT, VFC, XRX, WFC, FOSL, WMB, FCX, NUE, TSN, USB, EA, WBA, XYL, TSS, NTRS, WYNN, WY, UNM, UA, XL, VZ, TXT, BEN, PCLN, TWC, ED, URI, UTX,

FLR, TDC, UPS, NOC, EL, NVAX, UNP, FISV, VMC, FDX, WMT, VNO, DIS, TMK, V, XEL, REGN, FITB, TJX, WFM, TYC, VTR, XEC, FFIV, SCHW, EIX, WHR, URBN, FAST, GRPN, FSLR, VIAB, ETN, RTN, JBLU, VRSN, EXPD, EQT, ETR, KO, EMC, SYMC, AKAM, GD, EXPE, SWKS, DOX, JAZZ, STX, RRC, FOX, EQIX, XLNX, IPG, GE, SBAC, GME, LRCX, EMR, CSCO, RF, TSCO, TRIP, MU, QVCA, XOM, TROW, GRMN, IP, JCI, ESRX, GPS, INTU, ROK, MDLZ, FTI, EMN, ESS, LULU, GS, SNDK, JEC, SHPG, PX, RSG, BIDU, VOD, RAI, EXC, VRSK, EW, GIS, FTR, SRCL, IRM, GM, ORLY, EQR, F, ETFC, RHT, IFF, HUM, GGP, RHI, LKQ, VIA, TMUS, SIRI, ULTA, AVGO, IVZ, VRTX, EOG, EBAY, NCLH, XRAY, GPC

References

Alfonsi, A., A. Fruth & A. Schied (2010), 'Optimal execution strategies in limit order books with general shape functions', *Quantitative Finance* **10**(2), 143–157.

Alfonsi, A. & A. Schied (2010), 'Optimal trade execution and absence of price manipulations in limit order book models', *SIAM Journal on Financial Mathematics* **1**(1), 490–522.

Avellaneda, M., J. Reed & S. Stoikov (2011), 'Forecasting prices from level-1 quotes in the presence of hidden liquidity', *Algorithmic Finance* **1**(1).

Avellaneda, M. & S. Stoikov (2008), 'High-frequency trading in a limit order book', *Quantitative Finance* **8**(3), 217–224.

Bayraktar, E. & M. Ludkovski (2014), 'Liquidation in limit order books with controlled intensity', *Mathematical Finance* **24**(4), 627–650.

Bengio, Y. (2012), 'Practical recommendations for gradient-based training of deep architectures', *Neural Networks: Tricks of the Trade* pp. 437–478.

Biais, B., P. Hillion & C. Spatt (1995), 'An empirical analysis of the limit order book and the order flow in the paris bourse', *Journal of Finance* pp. 1655–1689.

Blanchet, J. & X. Chen (2013), Continuous-time modeling of bid-ask spread and price dynamics in limit order books.

Bouchaud, J., M. Mezard & M. Potters (2002), 'Statistical properties of stock order books: empirical results and models', *Quantitative Finance* **2**(4), 251–256.

Butaru, F., Q. Chen, B. Clark, S. Das & A. Lo (2015), Risk and risk management in the credit card industry, Technical Report w21305, National Bureau of Economic Research.

Cao, C., O. Hansch & X. Wang (2009), 'The information content of an open limit-order book', *Journal of futures markets* **29**(1), 16–41.

Cartea, A., R. Donnelly & S. Jaimungal (2015), Enhancing trading strategies with order book signals.

Chinco, A., A. Clark-Joseph & M. Ye (2015), Sparse signals in the cross-section of returns.

Cont, R. (2011), 'Statistical modeling of high-frequency financial data', *Signal Processing Magazine, IEEE* **28**(5), 16–25.

Cont, R. & A. De Larrard (2012), Order book dynamics in liquid markets: limit theorems and diffusion approximations.

Cont, R. & A. De Larrard (2013), 'Price dynamics in a markovian limit order market', *SIAM Journal on Financial Mathematics* **4**(1), 1–25.

Cont, R., A. Kukanov & S. Stoikov (2014), 'The price impact of order book events', *Journal of financial econometrics* **12**(1), 47–88.

Cont, R., S. Stoikov & R. Talreja (2010), 'A stochastic model for order book dynamics', *Operations Research* **58**(3), 549–563.

Demuynck, K. & F. Triefenbach (2013), 'Porting concepts from dnns back to gmms', *IEEE Workshop on Automatic Speech Recognition and Understanding* pp. 356–361.

Deng, L. & J. Chen (2014), 'Sequence classification using the high-level features extracted from deep belief networks', *Proceedings of IEEE ICASSP* .

Donier, J., J. Bonart, I. Mastromatteo & J. Bouchaud (2014), A fully consistent, minimal model for non-linear market impact.

Eisler, Z., J. Bouchaud & J. Kockelkoren (2012), 'The price impact of order book events: market orders, limit orders and cancellations', *Quantitative Finance* **12**(9), 1395–1419.

Fletcher, T. & J. Shawe-Taylor (2013), 'Multiple kernel learning with fisher kernels for high-frequency currency prediction', *Computational Economics* **42**(2), 217–240.

Gao, X., J. Dai, T. Dieker & S. Deng (2014), Hydrodynamic limit of order book dynamics.

Gould, M. & J. Bonart (2015), Queue imbalance as a one-tick-ahead price predictor in a limit order book.

Gould, M., M. Porter, S. Williams, M. McDonald, D. Fenn & S. Howison (2013), 'Limit order books', *Quantitative Finance* **13**(11), 1709–1742.

Ioffe, S. & C. Szegedy (2015), Batch normalization: Accelerating deep network training by reducing internal covariate shift.

Kearns, M. & Y. Nevmyvaka (2006), Reinforcement learning for optimized trade exectuion, 23rd international conference on Machine Learning, ACM.

Kempf, A. & O. Korn (1999), 'Market depth and order size', *Journal of Financial Markets* **2**(1), 29–48.

Kercheval, A. & Y. Zhang (2015), 'Modeling high-frequency limit order book dynamics with support vector machines', *Quantitative Finance* **15**(8), 1315–1329.

Khandani, A., A. Kim & A. Lo (2010), 'Consumer credit-risk models via machine-learning algorithms', *Journal of Banking and Finance* **34**(11), 2767–2787.

Likas, A. (2001), 'Probability density estimation using artificial neural networks', *Computer Physics Communications* **135**, 167–175.

Lipton, A., U. Pesavento & M. Sotiropoulos (2013), Trade arrival dynamics and quote imbalance in a limit order book.

Maglaras, C., C. Moallemi & H. Zheng (2015), Optimal execution in a limit order book and an associated microstructure market impact model.

Mamaysky, H. & P. Glasserman (2015), Does unusual news forecast market stress?

Moallemi, C., B. Park & B. Van Roy (2012), 'Strategic execution in the presence of an uninformed arbitrageur', *Journal of Financial Markets* **15**(4), 361–391.

Paulik, M. (2013), 'Lattice-based training of bottleneck feature extraction neural networks', *INTERSPEECH* **89-93**.

Potters, M. & J. Bouchaud (2003), 'More statistical properties of order books and price impact', *Physica A: Statistical Mechanics and its Applications* **324**(1), 133–140.

Predoiu, S., G. Shaikhet & S. Shreve (2011), 'Optimal execution in a general one-sided limit-order book', *SIAM Journal on Financial Mathematics* **2**(1), 183–212.

Sainath, T., B. Kingsbury & B. Ramabhadran (2012), 'Auto-encoder bottleneck features using deep belief networks', *Proceedings of IEEE ICASSP* pp. 4153–4156.

Schied, A. & T. Schoneborn (2009), 'Risk aversion and the dynamics of optimal liquidation strategies in illiquid markets', *Finance and Stochastics* **13**(2), 181–204.

Sirignano, J., A. Sadhwani & K. Giesecke (2016), Deep learning for mortgage risk.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever & R. Salakhutdinov (2014), 'Dropout: A simple way to prevent neural networks from overfitting', *The Journal of Machine Learning Research* **15**(1), 1929–1958.

Stoikov, S. & R. Waeber (2015), Reducing transaction costs with low-latency trading algorithms.

Sutskever, I., J. Martens, G. Dahl & G. Hinton (2013), 'On the importance of initialization and momentum in deep learning', *Proceedings of the 30th international conference on machine learning (ICML-13)* pp. 1139–1147.

van den Oord, A. & B. Schrauwen (2014), 'Factoring variations in natural images with deep gaussian mixture models', *Advances in Neural Information Processing Systems* pp. 3518–3526.

Variani, E., E. McDermott & G. Helgold (2015), 'A gaussian mixture layer jointly optimized with discriminative features within a deep neural network architecture', *2015 IEEE International Conference on Acoustics, Speech and Signal Processing* pp. 4270–4274.

Yang, T. & L. Zhu (2015), A reduced-form model for level-1 limit order books.

Yu, D. & M. Seltzer (2011), 'Improved bottleneck features using pretrained deep neural networks', *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy* pp. 237–240.