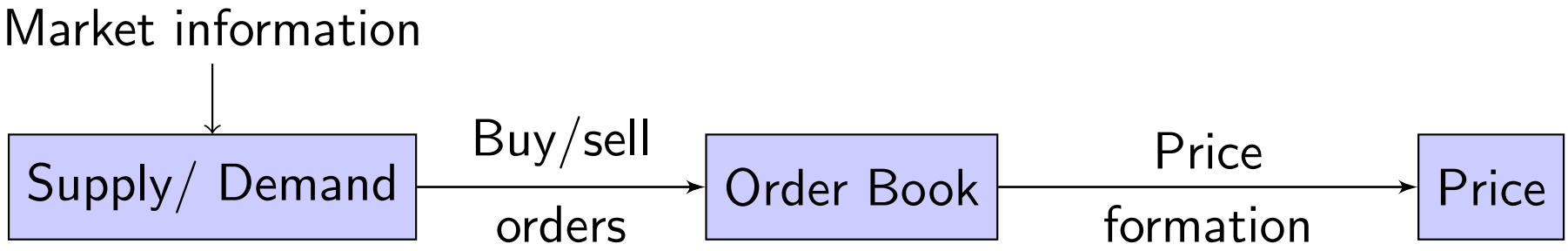
Universal features of intraday price formation Perspectives from Deep Learning

Rama Cont
(Mathematical Institute, University of Oxford))
&
Justin Sirignano
(University of Illinois, Urbana Champaign)

Based on:

Universal Features of Price Formation in Financial Markets:
Perspectives From Deep Learning
https://ssrn.com/abstract=3141294

Price formation



- Market prices react to fluctuations in supply and demand. But how *exactly* do they react?
- We call 'price formation mechanism' the correspondence between the history of supply and demand (orders and transactions) and the (next) market price:

$$\text{Price}(t + \Delta t) = \text{F}(\text{Price history}([0,t]), \text{Order Flow}([0,t]))$$

- Econometric models, market microstructure models, stochastic models and machine-learning price prediction models can all be thought of as models for this map F, at various frequencies Δt.

Universal vs asset-specific modeling

$$\text{Price}(t + \Delta t) = \text{F}(\text{Price history}(0..t), \text{ Order Flow}(0..t))$$

- Is F specific to each asset or 'universal'?
- Theoretical microstructure models implicitly assume 'universality' holds
- Some empirical evidence points to existence of universal relations between volume, order flow and price dynamics: Kyle and Obizhaeva (2016), Benzaquen et al (2017),...
- Yet empirical modeling and market practice in use of statistical models remains asset-specific: **a model for asset A is estimated/ trained on time series of the asset A.**
- Does F vary with time? **Nonstationarity** is often cited as a reason for using *recent* data histories for estimating such models, lest some latent factors/parameters change with time.

Using HF order book data to explore price formation

- Computerization of markets has unleashed PetaBytes of high frequency data on transactions, order flow and order book dynamics in listed markets.
- Intuitively, this huge amount data contains a lot of information which one should be able to use to investigate models of price formation and build/improve intraday risk models and price prediction models.
- Yet most statistical models are based on price/returns only, ignoring the wealth of information contained in order flow.
- **Machine learning** (ML) can potentially be helpful for exploring such large data sets. ML approaches are increasingly used in trading for short term prediction but there has been no systematic study of their performance and stability.
- Most case studies using ML on high frequency data are limited to a few assets and periods less than a month.

Limit order book: snapshot of supply and demand

	Size	Price
Asks	200	$80.03
	0	$80.02
	400	$80.01
	1100	$80.00
Bids	1000	$79.99
	500	$79.98
	50	$79.97
	400	$79.96

In electronic markets we observe the limit order book every microsecond. What can we learn about price formation from this data?

Estimation of high-dimensional functions

- Abstract formulation: estimating a (scalar-valued) function defined on a (very) high dimensional (but bounded) domain

$$F : [0,1]^N \to A \subset \mathbb{R}$$

  from data set $(x_i, F(x_i)), i = 1..n$. A is often a finite or bounded set.
- Problem: Data dimension N is large.
- Linear representation: approximate F by linear combination $\widehat{F}_N = \sum_{k=1}^{N} a_k f_k$ then estimate a_k from data
- Curse of dimensionality: to get $\|F - \widehat{F}_N\| \leq \epsilon$ we need $N \sim \epsilon^d$: exponentially many elements

Nonlinear representation of high-dimensional functions

Kolmogorov Representation theorem (Kolmogorov 1956, Kolmogorov & Arnold 1957, Lorentz 1962, Sprecher 1965, Ruschendorf & Thomsen 1997)
Any continuous functions $F : [0,1]^N \to A \subset \mathbb{R}$ can be expressed as a finite composition of linear functions and (at most $2N$) continuous functions of a **single** variable:

$$F(x) = \sum_{k=0}^{2N} \varphi \left(\sum_{p=1}^{N} \phi_{k,p}(x_p) \right)$$

where $\varphi : \mathbb{R} \to \mathbb{R}$ increasing and nonlinear; ex:
$\varphi(x) = 1/(1 + \exp(-x))$
$\phi_{k,p} : \mathbb{R} \to \mathbb{R}$ Lipschitz
$\to$ representation of high dimensional functions as **iterated compositions** of linear+ 1-d nonlinear functions
Related to Hilbert's 13th problem.

Approximation by iterated composition of scalar functions

Activation function: Let $\varphi : \mathbb{R} \to [0,1]$ be a monotone increasing function with $\varphi(-\infty) = 0, \quad \varphi(\infty) = 1$.
Example: $\varphi(x) = 1/(1+\exp(-x))$.
Denote by L_φ the set of functions of the form

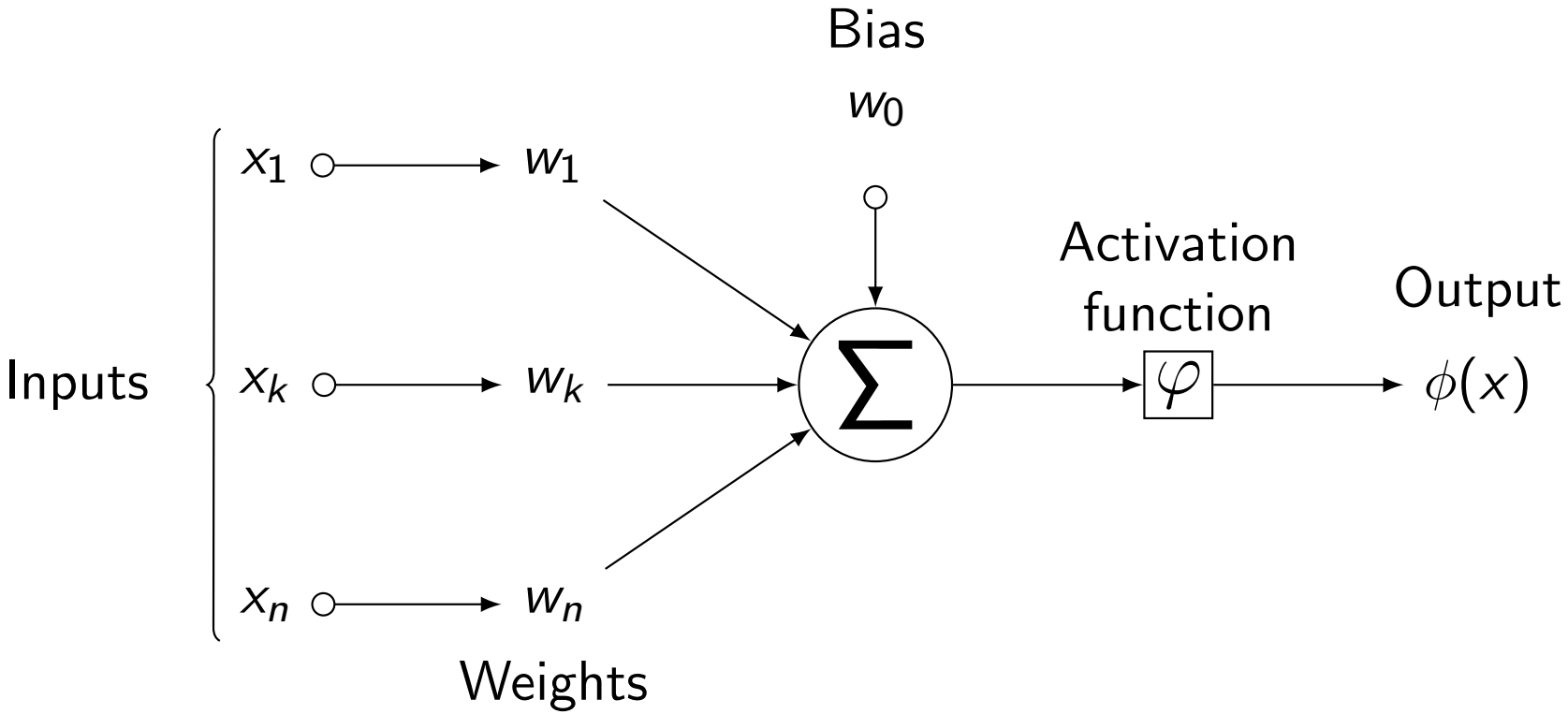
$$\phi(x) = \sum_{i=1}^{p} a_i \quad \varphi\left(b_i + < W_i, x >\right)$$

where $W_i \in \mathbb{R}^N, a, b \in \mathbb{R}^N$
Idea: approximate an arbitrary (continuous) functions
$F : [0,1]^N \to A \subset \mathbb{R}$ by **iterated compositions** of functions in L_φ:

$$F \simeq F_k = \phi_1 \circ \phi_2 \circ .... \circ \phi_k \qquad \phi_i \in L_\varphi$$

Neural network representation

A function $\phi(x) = \varphi\left(w_0 + \sum_{k=1}^{n} w_k x_k\right)$ in L_φ may be represented as a **neuron** with **synaptic weights** $W = (w_1, \ldots w_n)$ and **activation function** φ



Universal approximation theorem

Any continuous function can be approximated by **composing** such functions (Hecht-Nielsen 1989, Hornik 1991, Kurkova 1992,...):

Theorem (Universal approximation theorem)

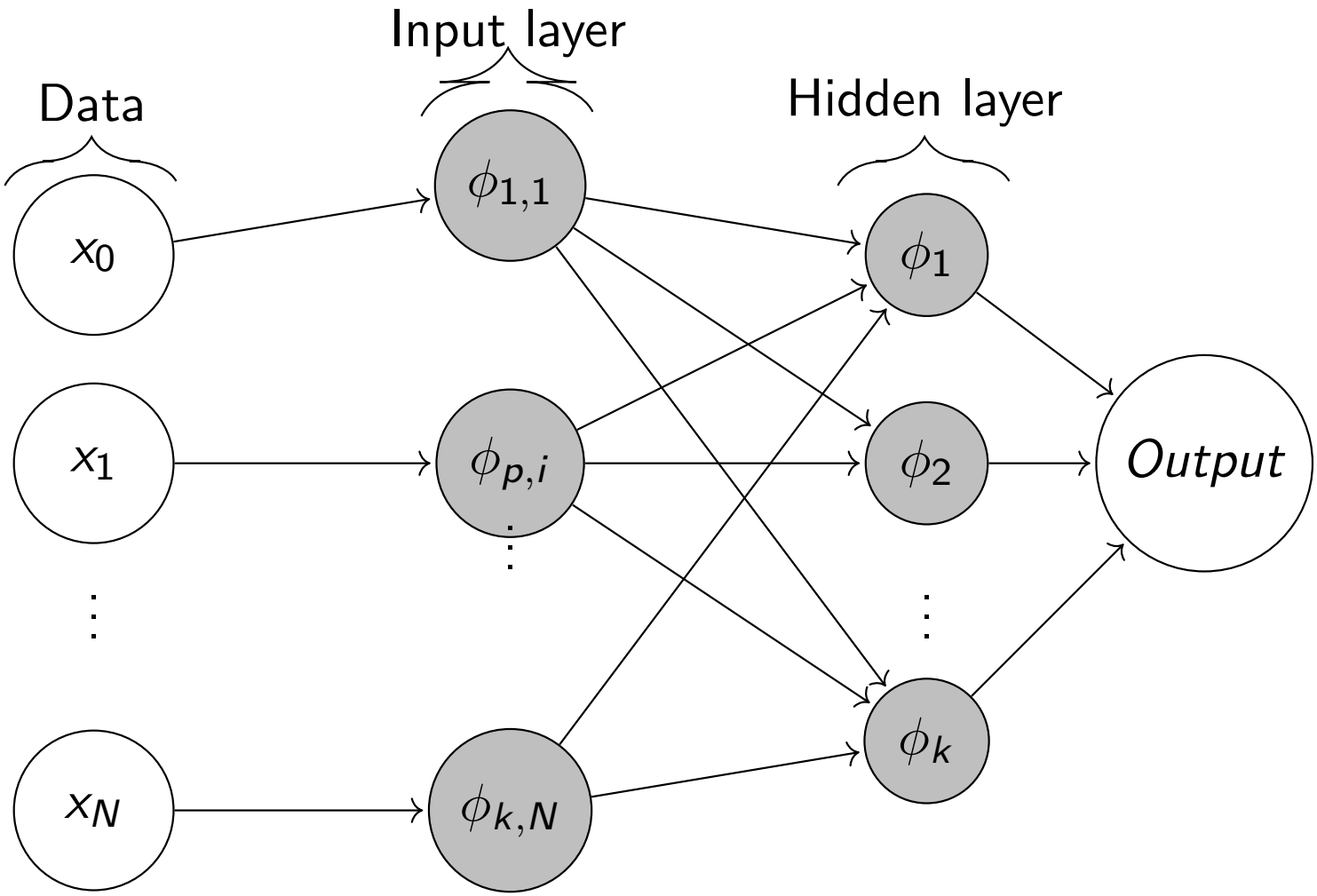
Let $F : [0,1]^N \to A \subset \mathbb{R}$, $\varphi : \mathbb{R} \to [0,1]$ *an activation function and* $\epsilon > 0$. *There exist* $k \in \mathbb{N}$, $\phi_{p,i} \in L_\varphi$, $\phi_1, ..., \phi_k \in L_\varphi$ *such that*

$$\left| \quad F(x_1, ..., x_N) - \sum_{p=1}^{k} \phi_p \left(\sum_{i=1}^{N} \phi_{p,i}(x_i) \right) \quad \right| \leq \epsilon$$

This can be represented to a (two-layer) **neural network** :
Input layer with kN neurons $\phi_{p,i}, p = 1..k, i = 1..N$
'Hidden' layer with k neurons: $\phi_1, ..., \phi_k$

Neural network representation

This approximation corresponds to a 2-layer neural network with one input layer and one hidden layer:

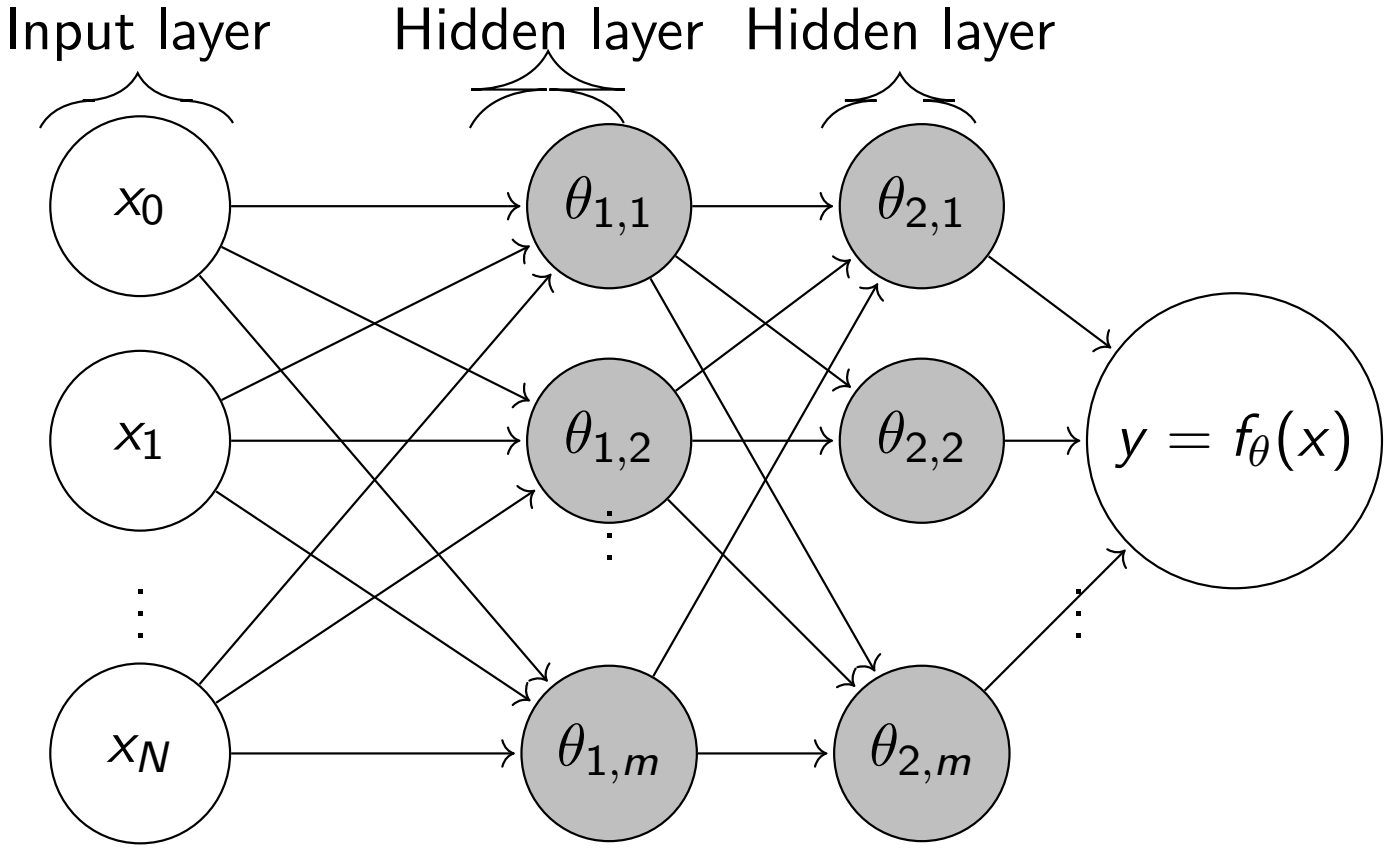


Deep neural networks

More generally networks with many layers ('deep') have been shown to be very effective for learning high dimensional functions from extremely large data sets: these lead to apprximations of type

$$f_\theta = \varphi \circ g_{\theta_k} \circ \varphi \circ g_{\theta_{k-1}} \circ \ldots \circ \varphi \circ g_{\theta_1}$$

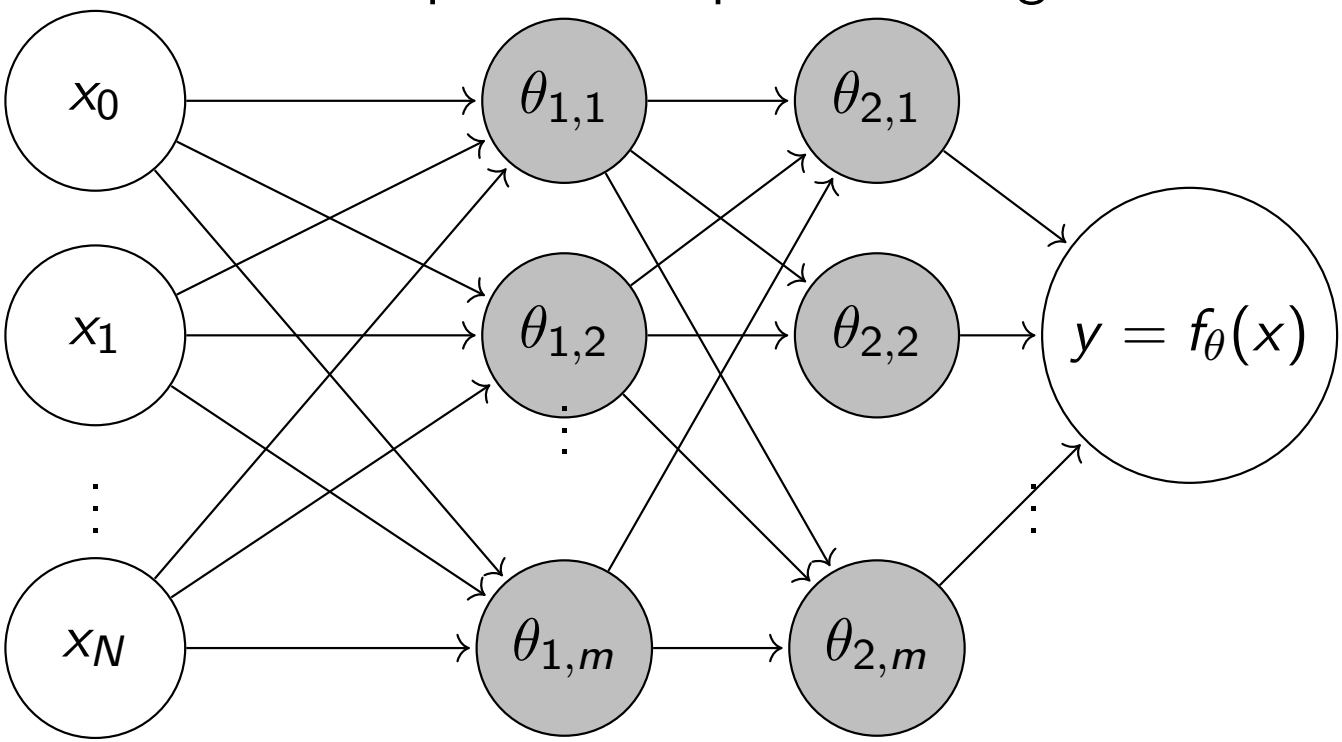
where $g_{\theta_i}(x) =< \theta_i, x >$, $\varphi(y) = 1/(1 + \exp(-y))$



Supervised Deep learning

$$f_\theta(x) = \varphi \circ g_{\theta_k} \circ \varphi \circ g_{\theta_{k-1}} \circ \ldots \circ \varphi \circ g_{\theta_1}(x) \qquad g_{\theta_i}(x) = < \theta_i, x >$$

The weights θ_i are estimated by minimizing an objective function (negative log-likelihood, relative entropy) over data sample.
K layers, m neurons per layer $\Rightarrow Km^2 \simeq 10^5 - 10^6$ weights
Suitable for non-parametric pattern recognition in **large** data sets.



Data:

- 3 years of tick-by-tick NASDAQ limit order book data for 1000 NASDAQ stocks (Jan 1, 2014- March 31, 2017)
- All quotes and trades for NASDAQ stocks: $>$**hundred billion** order book events
- Training set: Jan 2014- Dec 2015, Test set: 2016-2017 months
- Let $\tau_1, \tau_2, \ldots$ be the times at which price changes occur.
- Prediction target used to assess out of sample performance:

$$\mathbb{P}[\Delta\text{Mid-Price}_{\tau_{k+1}} > 0 | \text{Order Book History for } t < \tau_k]$$

Prediction of price moves from order flow

$\mathbb{P}[\Delta\text{Mid-Price}_{\tau_{k+1}} = k | X_{0:\tau_k}] = f_\theta(X_{\tau_k}, h_{\tau_k})$

- Classification problem : e.g. $k \in \{< -1, -1, +1, > +1\}$
- X_t: state of order book
- h_t: **features** learned from past order flow $< t$

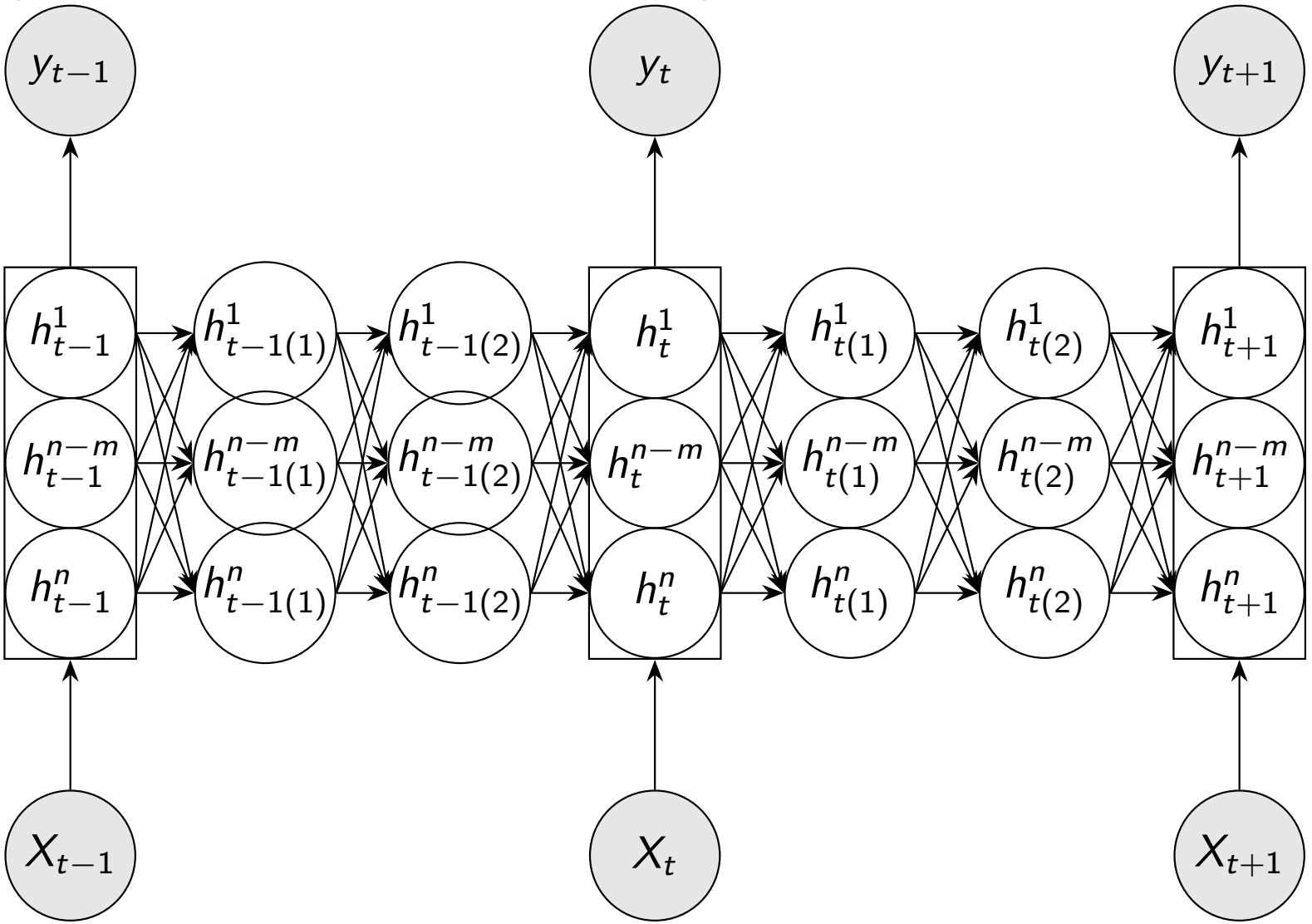
Different choices for model specification

- Cox process with Linear (Vector AutoRegressive) features:
  $h_t = AX_{t-T:t} + D$, $f_\theta =$ logistic function
  $\mathbb{P}[\Delta\text{Mid-Price}_{\tau_{k+1}} = k | X_{0:t}] = \frac{1}{1+\exp(-\theta.h_t)}$
- Multilayer ('deep') neural network: iteration of
  linear combination $g_{\theta_i}(x) =< \theta_i, x > +$
  activation function $\varphi(y) = 1/(1+\exp(-y))$:

$$f_\theta = \varphi \circ g_{\theta_k} \circ \varphi \circ g_{\theta_{k-1}} \circ \ldots \circ \varphi \circ g_{\theta_1} \qquad \varphi(y) = \frac{1}{1+\exp(-y)}$$

$$h_{t-1} = \varphi \circ g_{\theta'_n} \circ \ldots \circ \varphi \circ g_{\theta'_1}(h_{t-1}, X_{t-T:t})$$

Architecture: Recurrent Long-Short Term Memory network (Hochreiter & Schmidhuber 1997)



. . .

Network architecture: Long/ Short Term Memory

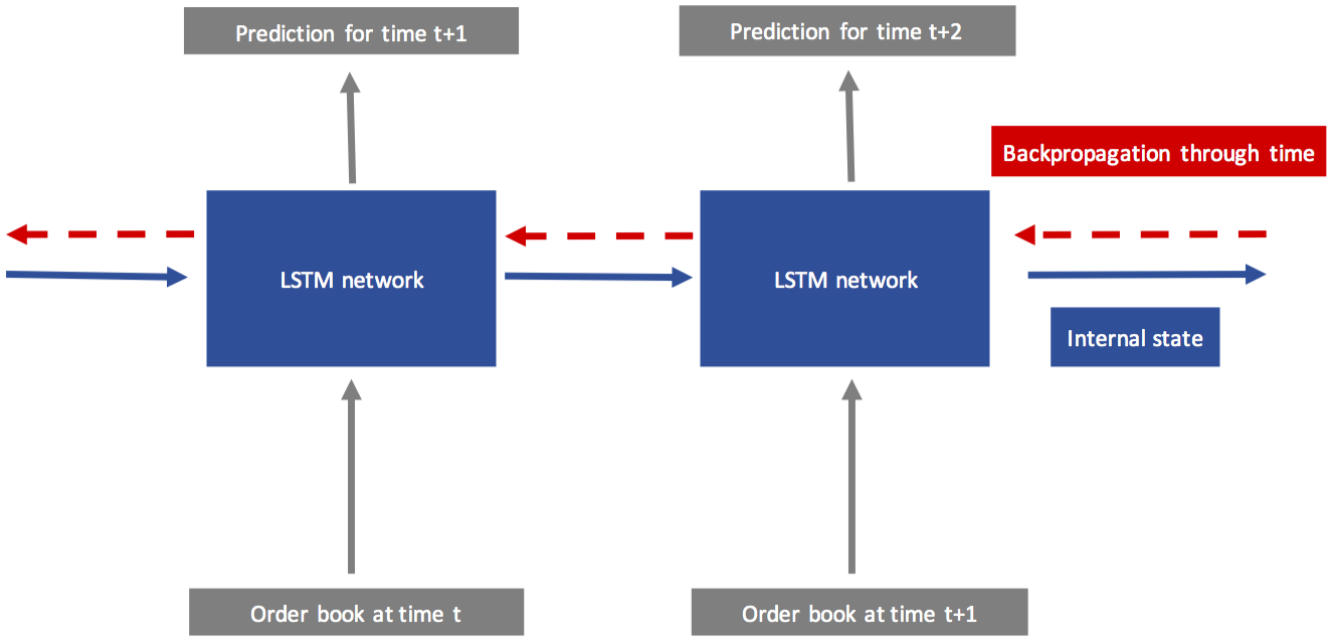


Figure: At each time the LSTM network accepts as an input the current order book data and the internal state from the previous times. Training of the model requires backpropagation back through previous times.

Supervised learning

$\mathbb{P}[\Delta\text{Mid-Price}_{\tau_{k+1}} = k | X_{0:\tau_k}] = f_\theta(X_{\tau-k}, h_{\tau_k})$

- Supervised learning= estimating θ using a (large) data sample in order to minimize an in-sample loss function
- Here: minimize Relative entropy of forecasted distribution with respect to empirical distribution
- Large models : hundreds of thousands of parameters.
- Optimization method: stochastic gradient descent wrt θ

- For such large data sizes, naive training/ optimization algorithms do not scale properly: need to use carefully designed, scalable algorithms which can handle memory constraints and computational speed in an efficient manner: **Asynchronous Stochastic Gradient Descent.**
- Gradient computed using 'BackPropagation in time' = chain rule applied recursively to $\varphi, g_{\theta_k}, \varphi, ...$
- Large computational expense: use a cluster of 500 GPUs

- Each GPU allows for parallelization across thousands of cores

Figure: Calculations were done on the National Center for SuperComputing Applications (NCSA) BlueWaters supercomputer in Illinois (USA).

Asynchronous stochastic gradient descent

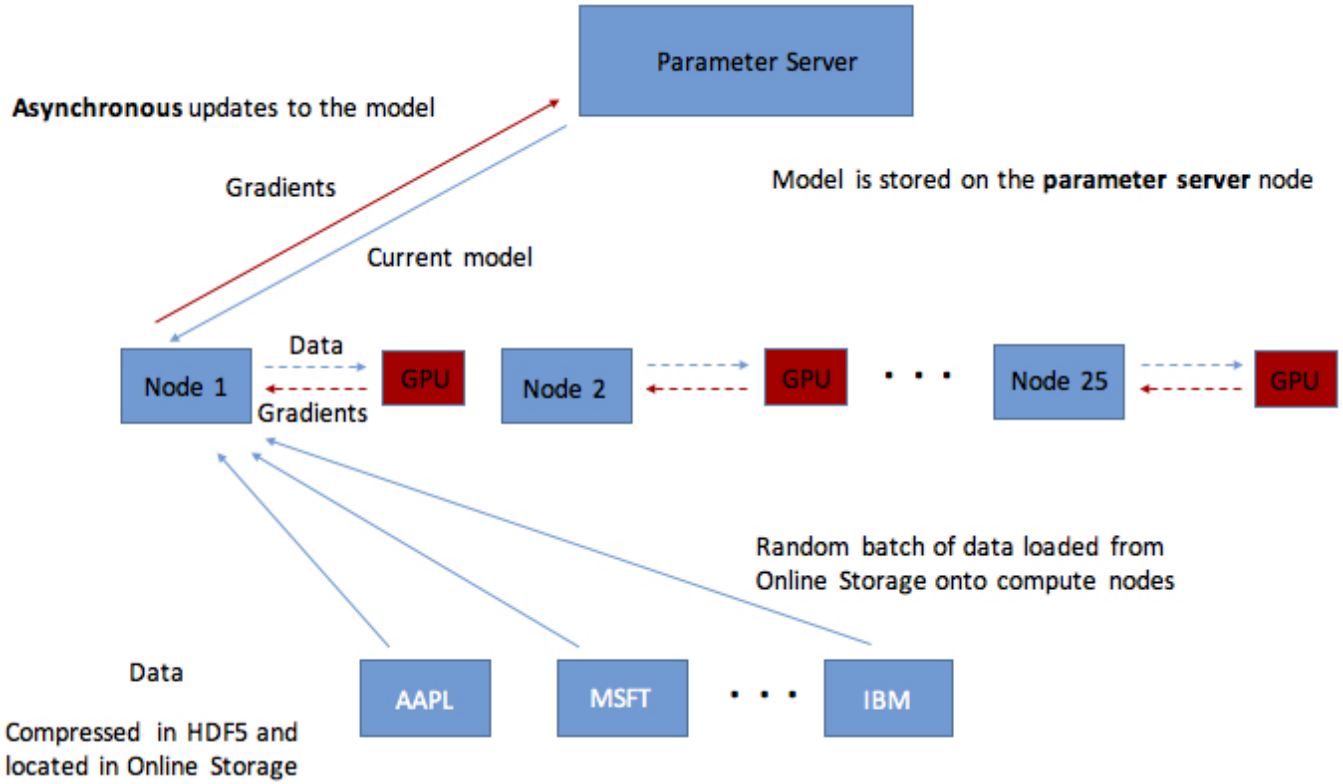


Figure: The dataset, which is too large to be held in the nodes' memories, is stored on the Online Storage system. Batches of data are randomly selected from all stocks and sent to the GPU nodes. Gradients are calculated on the GPUs and then the model is asynchronously updated.

Comparison of Single-Stock Model with Joint Model

- Train individual models for each stock (single-stock model)
- Train a joint model for all stocks using data from all stocks
- Objective function: cross-entropy (i.e., average negative log-likelihood)
- Evaluation metric: **out-of-sample forecast accuracy of direction of next price move** = % of out-of-sample events where model predicts correct direction of price move
- Benchmark: Random forecast (coin flip) leads to close to 50% accuracy
- If joint model is more accurate than the single-stock model, this implies:
  - Universal structure of limit order books for different stocks
  - Universal structure can be taken advantage of to develop more accurate models

Comparison with linear models

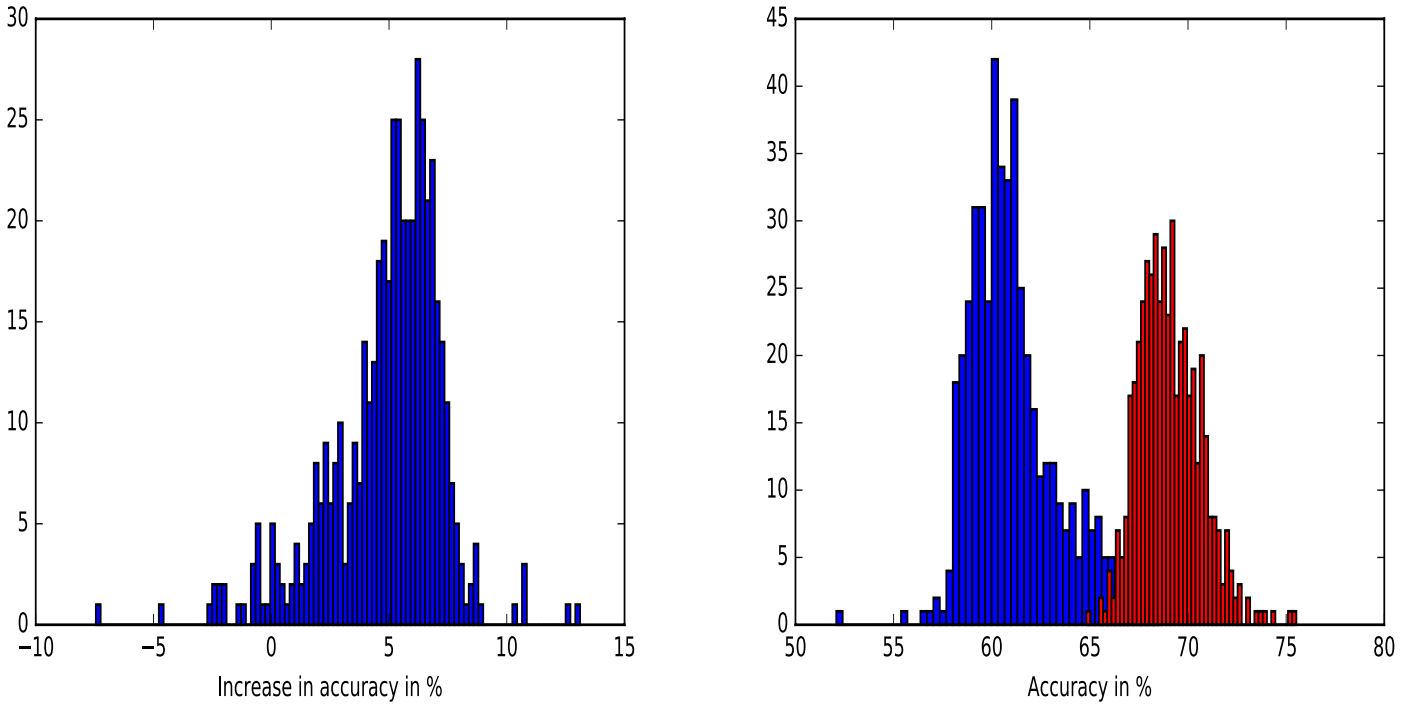


Figure: Comparison with linear models: out of sample prediction accuracy for direction of next price move across 500 stocks and out-of-sample results reported for June-August, 2015. Left: increase in accuracy for stock-specific deep neural networks vs stock-specific linear models. Right: universal deep neural network (red) vs stock-specific linear models (blue).

Universality of price formation mechanism

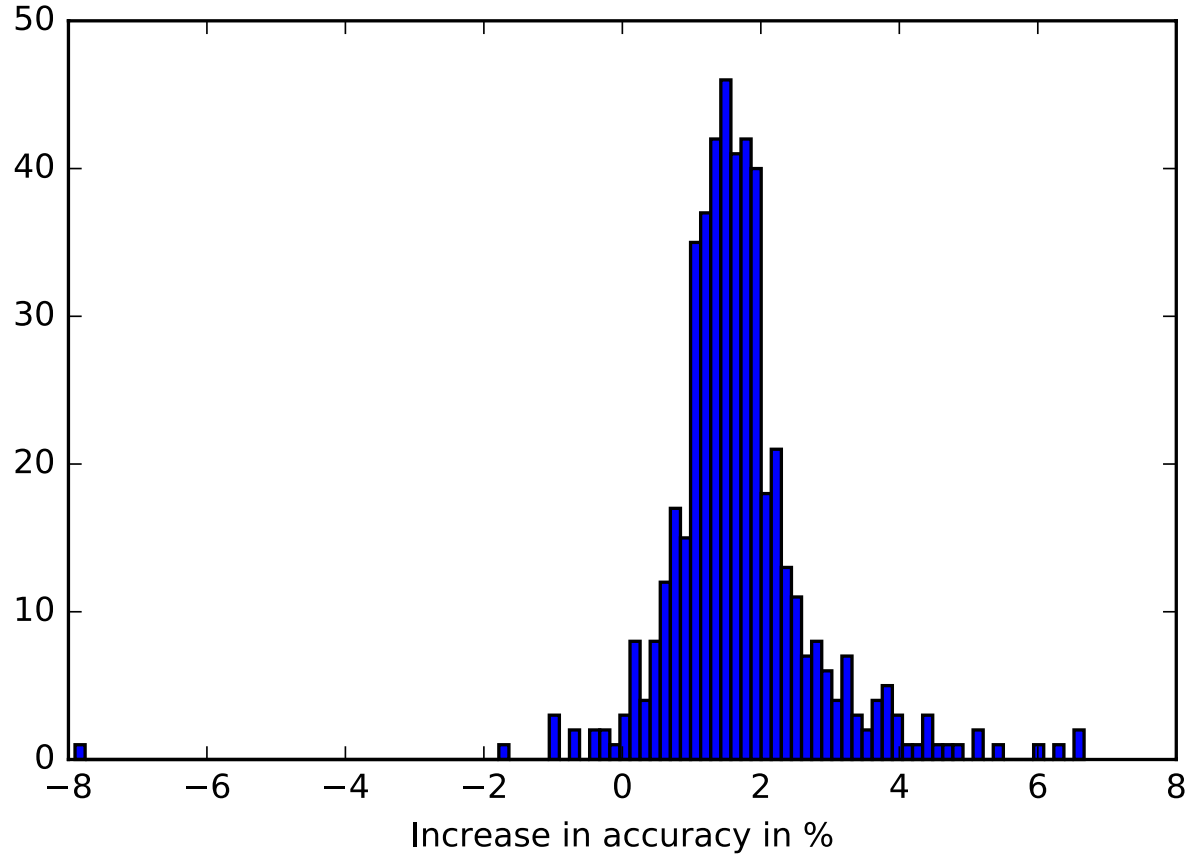
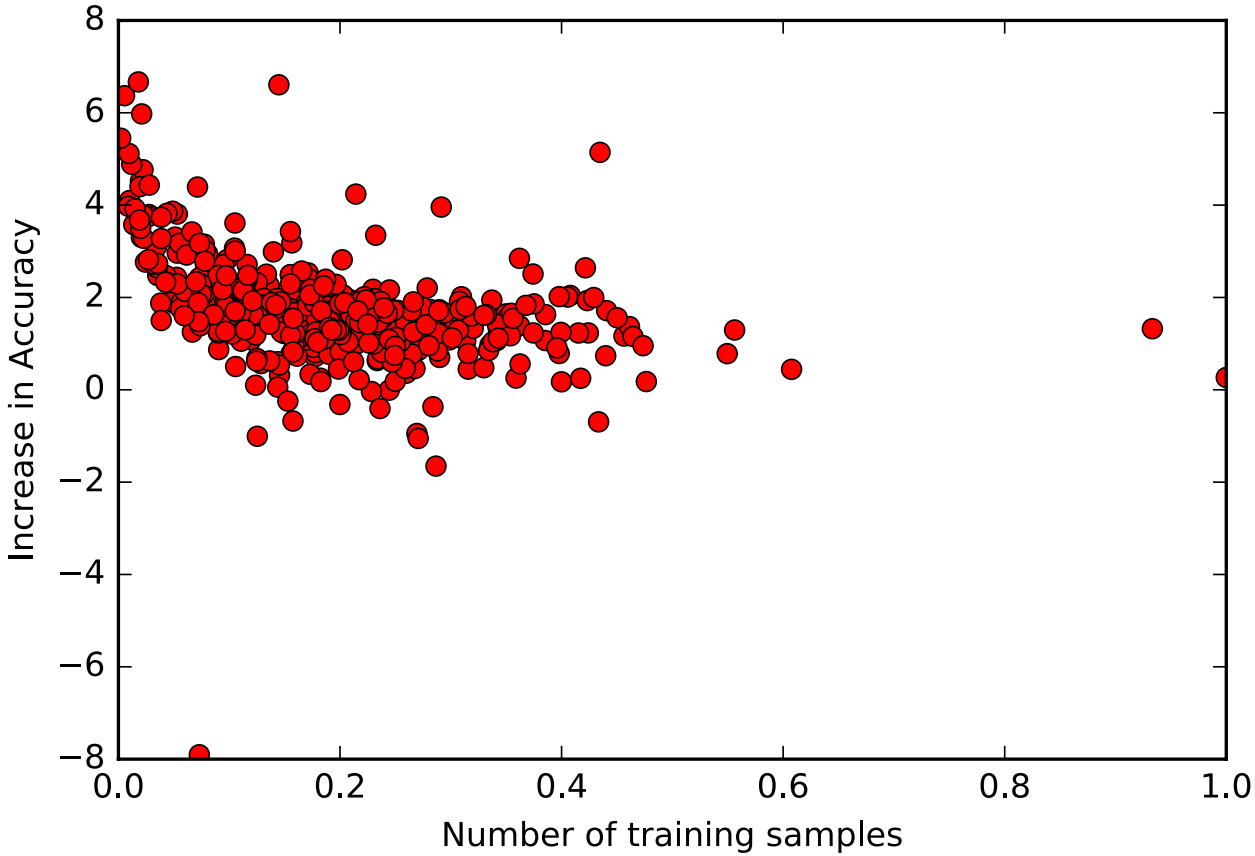


Figure: Universal vs stock-specific models, both estimated via deep networks with 3 LSTM layers followed by a ReLU layer of 50 units. Out-of-sample price prediction accuracy across 489 stocks, June-August, 2015.

Universal vs stock specific models

Superiority of the universal model trained on all stocks can be traced back to the availability of a larger, richer and more diverse set of scenarios for the universal model: improvement in forecast is highest for stocks with lower sample size.



Universal model generalizes to stocks it was not trained on

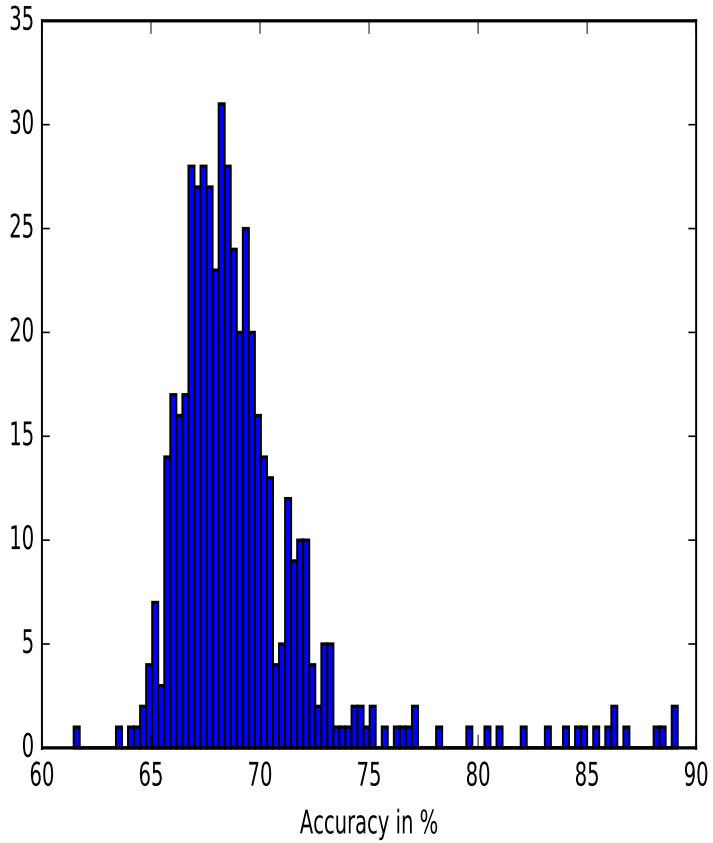


Figure: Performance on approximately 500 new stocks which the model has **not been trained on**. Out-of-sample accuracy for June-August, 2015. Training sample: January 2014-May 2015.

Transfer Learning: Universal model generalizes to stocks it was not trained on

Model	Comparison	Average increase in accuracy
Stock-specific	25/25	1.45%
Universal	4/25	-0.15%

Table: Comparison of universal model trained on stocks 1-464 versus (1) stock-specific models for stocks 465-489 and (2) universal model trained on all stocks 1-489. Second column = fraction of stocks where the universal model trained only on stocks 1-464 outperforms models (1) and (2). 3rd column = average increase in accuracy. Out-of-sample results for June-August, 2015.

Why does the universal model outperform stock-specific models?

- Universal model is trained on a sample which roughly 500 times larger than any stock-specific sample and contains a much richer and more diverse set of scenarios.
- For example, while a given stock may not have experienced high volatility over a given year, there are always stocks in the sample which will have experienced flash crashes, high volatility, rally etc during the same period.
- This is roughly equivalent to training a single stock model on 1000 years of data!
- Diversity + data size outweigh other learning strategies such as training sector by sector, separating into large/small tick stocks etc. Such *ad-hoc*/ a priori clustering methods reduce sample size but do not improve performance.

Stationarity across time

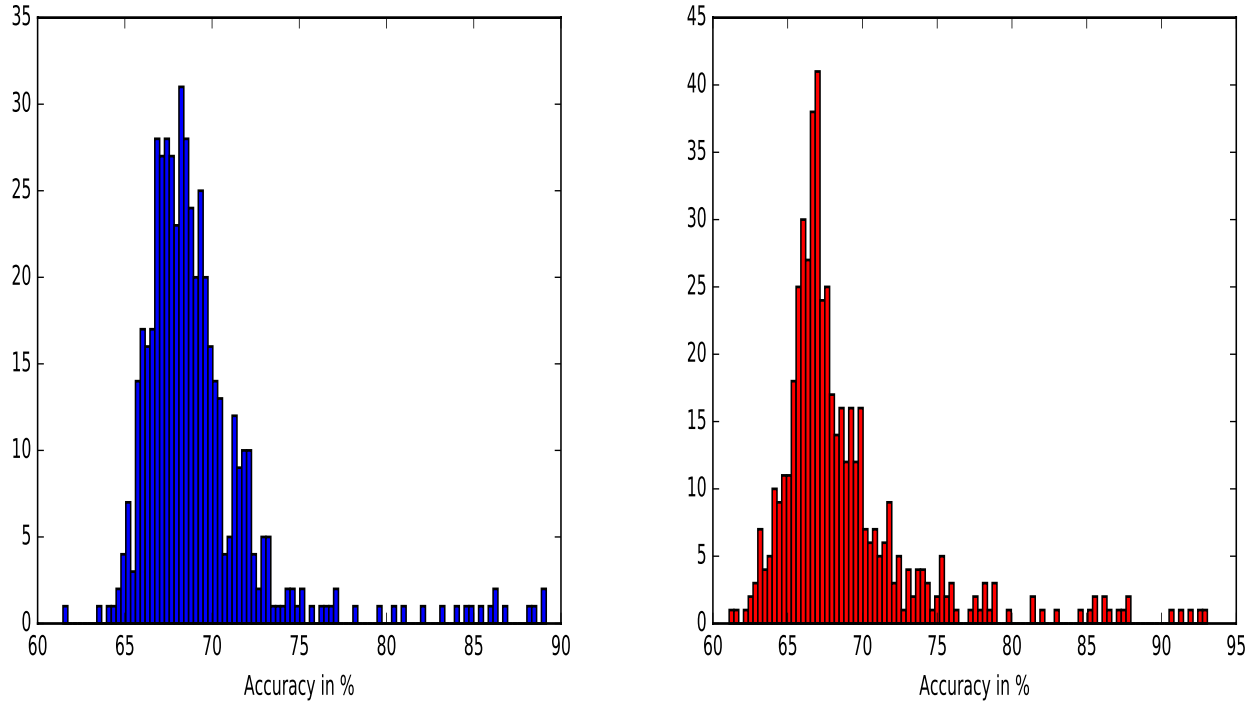


Figure: Performance on 500 out-of-sample stocks.
Left: out-of-sample accuracy reported for June-August, 2015.
Right: out-of-sample accuracy reported for **January-March, 2017**.
Training data: January 2014-May 2015.

Stationarity argues for longer training history

- The vast majority of time series models used in trading and risk management are estimated over short time windows due to worries about non-stationarity.
- For intraday modeling often this window is extremely short.
- Our results show otherwise: once the nonlinearities are correctly taken into account, the underlying model is **stable** and **stationary** months out of sample!

Size of training set	Average increase in accuracy
1 month	7.2%
3 months	3.7%
6 months	1.6%

Table: Comparison of deep learning models trained on entire training set (19 months) against deep learning models trained for shorter time periods. Models are trained to predict the direction of next mid-price move. Second column shows average increase in accuracy from using longer history. Out of sample results for 50 stocks, August, 2015.

Price formation is history-dependent

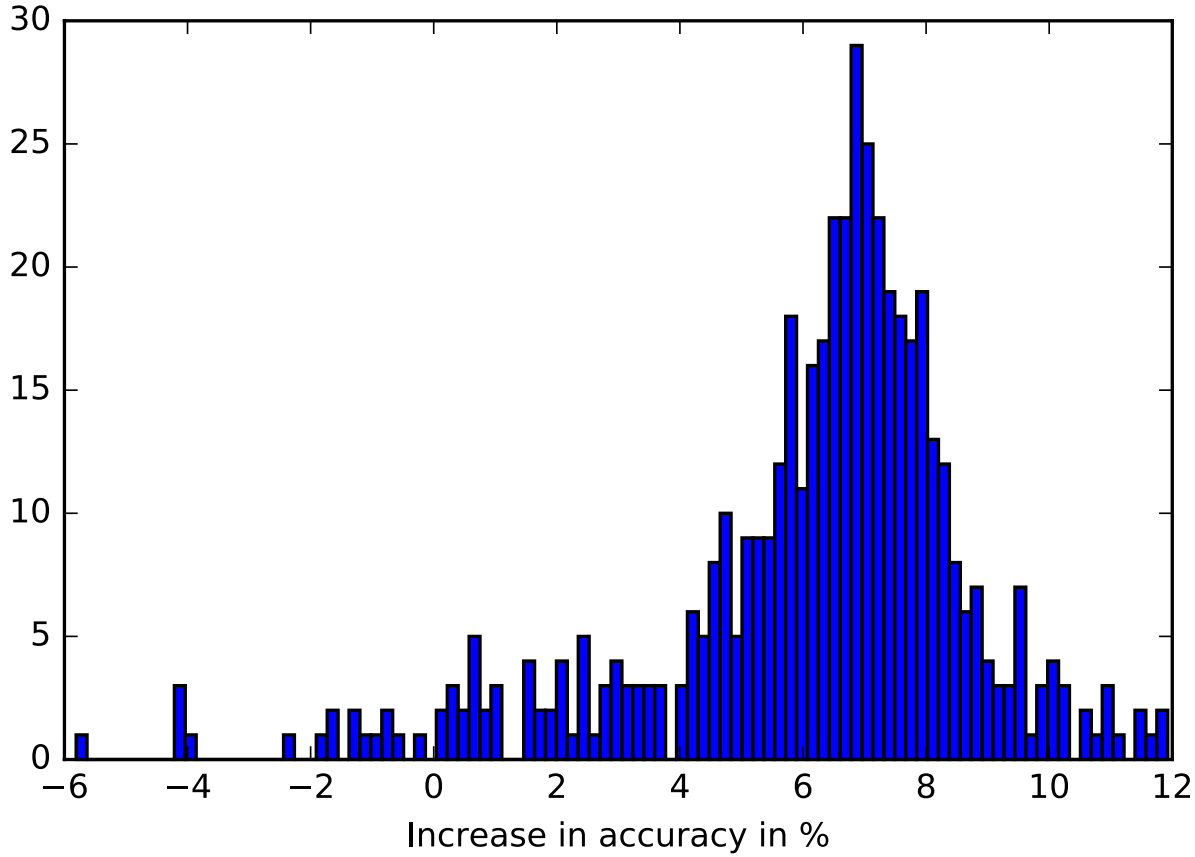


Figure: Comparison of a LSTM network with a feedforward neural network. Models are trained to predict the direction $\{-1, +1\}$ of next mid-price move. Comparison for approximately 500 stocks and out-of-sample results reported for June-August, 2015.

Price formation is history-dependent

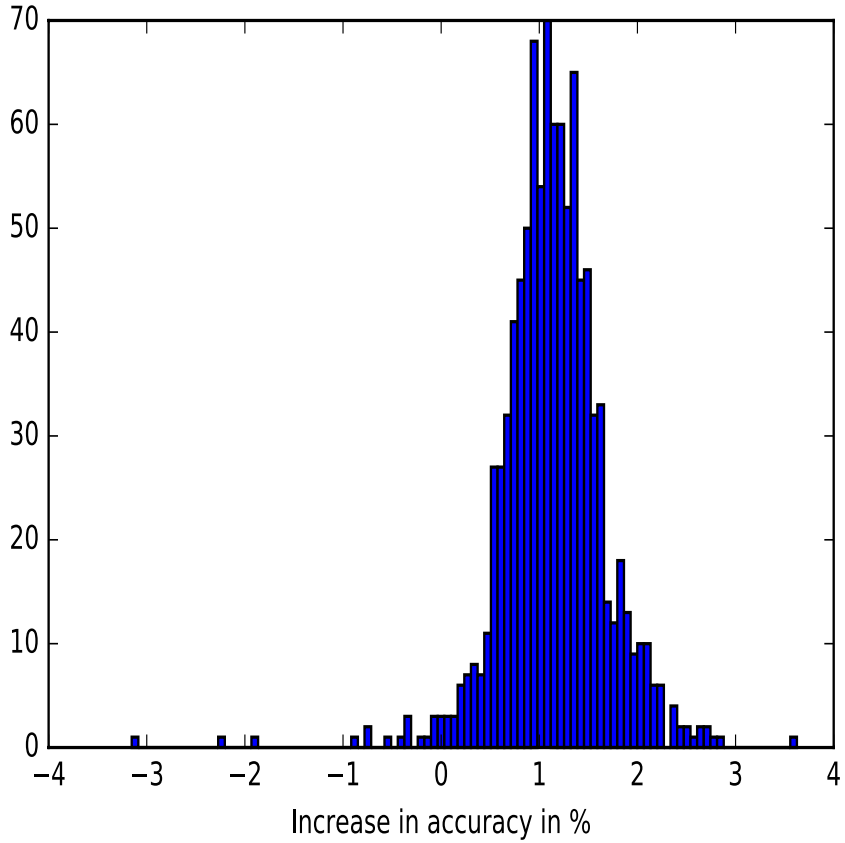


Figure: Increase in accuracy for 5000 events ($\sim$ 2 hours) versus 100 events. $1,000$ stocks in out-of-sample period June-August 2015.

Insights from Deep Learning: summary

- Using a large-scale **Deep Learning** approach applied to a large high-frequency database of electronic market transactions and quotes for US equities, we uncover non-parametric evidence for the existence of a **universal** and **stationary** price formation mechanism, **stable across stocks, sectors and time**, which captures the nonlinear relations between price fluctuations and order flow.

- This **universal price formation model** exhibits stable out-of-sample prediction accuracy across time, for a wide range of stocks from different sectors.

- Interestingly, these results also hold for stocks which are not part of the training sample!

Summary

- Deep Learning approach applied to high frequencyt order flow uncovers evidence of a **universal price formation model** mapping the recent order book history and order flow to price.
- **Universality**: model is stable across stocks and sectors, insensitive to large tick/small tick features etc. Universal model beats stock-specific models, even for stocks **not in training sample**, showing that features captured are not stock-specific.
- **Stationarity**: model performance is stable across time, even a year out of sample **without any adjustment** to model.

- Evidence of **path-dependence** in price formation: prices depend on the history of supply and demand, not just the instantaneous snapshot of orders. Including several hours of order flow history as input improves prediction accuracy.

- **Ability to generalize** ('transfer learning'): model extrapolates well to new/out of sample stocks and out-of-sample periods.

For more details:

R Cont, J Sirignano (2018):
Universal Features of Price Formation in Financial Markets: Perspectives From Deep Learning
https://ssrn.com/abstract=3141294

To appear in: **Quantitative Finance**.

Köszönöm szépen!