

# Predicting Price Movements in High-Frequency Financial Data with Spiking Neural Networks

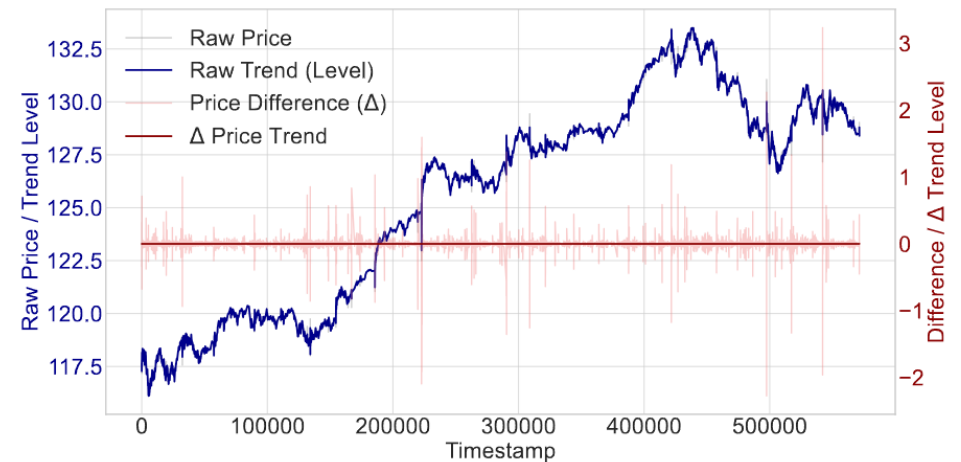
Brian Ezinwoke and Oliver Rhodes

Department of Computer Science, University of Manchester, UK

Neuro-Inspired Computational Elements 2026

# Background & Data

- **HFT Constraints:**
  - Microsecond timescales,
  - Extreme latency constraints
  - Vast amount of data to process
- **Market Dynamics:**
  - Nonstationary
  - Heavy tailed distributions
  - Highly noisy
- **Dataset:**
  - High frequency Apple price and volume data
  - 19 trading from Feb 2015



# Why use SNNs for this problem?

- **Temporal Encoding**
  - Financial price movements are sudden, discrete events
  - Naturally encodes temporal dependencies through sparse spiking activations
- **Neuromorphic hardware match**
  - Event Driven & Asynchronous
  - Theoretical latency + energy gains on neuromorphic hardware
- **Online Learning**
  - Local learning rules such as STDP

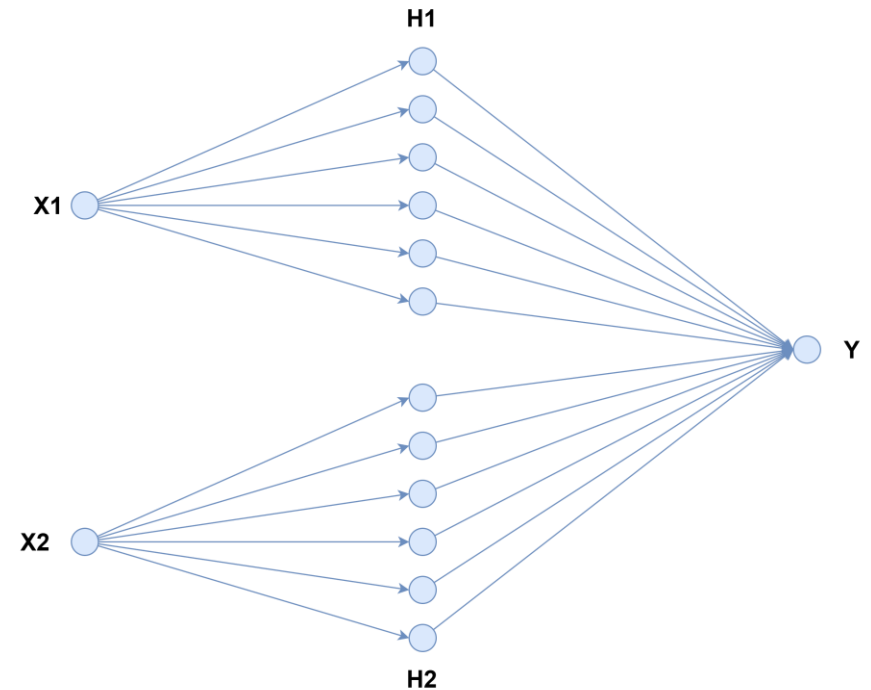


# Previous works

- **Polychronous Spiking Networks (PSNs)** - Reid et al. (2014):
  - Leveraged temporal capabilities of PSNs for financial time series prediction
  - Demonstrated favourable results against MLPs and linear models
  - Supervised approach on daily data
- **Unsupervised HFT Detection** - Gao et al. (2021):
  - Predicted price movements in high frequency data
  - Utilised unsupervised Spike-Timing-Dependent-Plasticity (STDP) method
  - The network became selective to temporal patterns in intra-day data
- **Current State of the art – Large Supervised ANNs:**
  - The current state of the art is primarily using supervised ANNs (e.g. LSTMs, Transformers...) on extremely large datasets run on thousands of GPUs
  - Specific details on architectures and training are kept proprietary within trading firms

# Gao et. Al.

- **Architecture:**
  - Leaky Integrate-and-Fire Neurons
  - Takes positive and negative price differences as input.
  - Clear separation between each channel
  - Learns with STDP rule
- **Weaknesses:**
  - Data normalisation used for preprocessing is highly susceptible to outliers.
  - Learning can be unstable and uninterpretable.



# Momentum Strategy

Classic **momentum logic** determines position direction based on position flag. The spike-based momentum strategy utilises signals from the SNN:

1. SNN takes in price data and outputs signal determining if there will be a significant movement in the price
2. Use standard momentum logic to determine the direction of the movement
3. Open position with the full capital amount, hold for 3 timestamps and then close position

---

## Algorithm 2 Spike Learning-Based Momentum Strategy

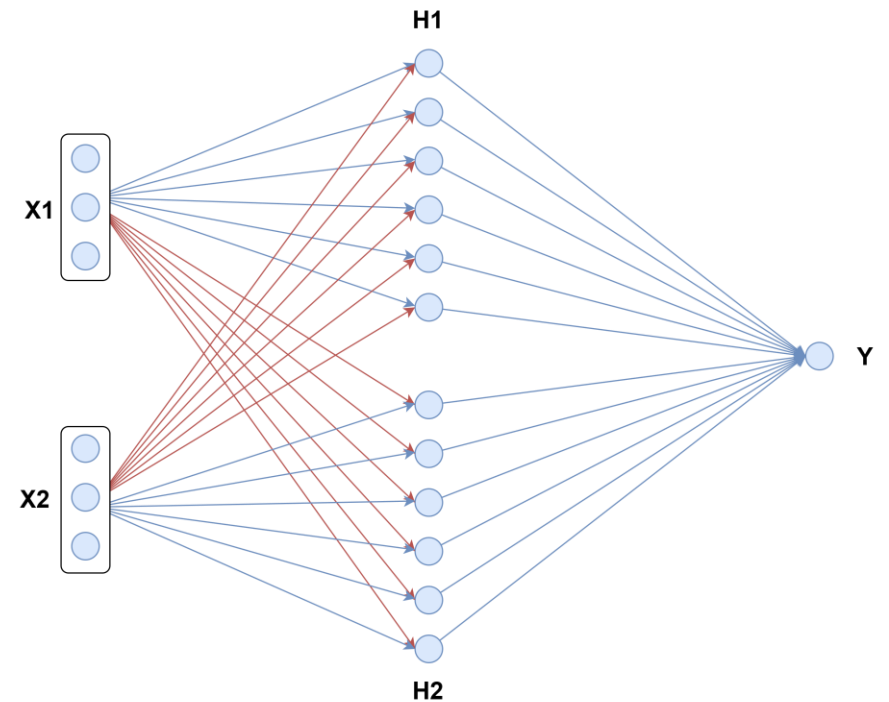
---

```
1: if position_flag > 0 then  
2:   Enter short position  
3: else if position_flag < 0 then  
4:   Enter long position  
5: else  
6:   No transaction  
7: end if
```

$$\text{position\_flag} = \frac{1}{n} \sum_{i=1}^n P_{t-i} - P_t$$

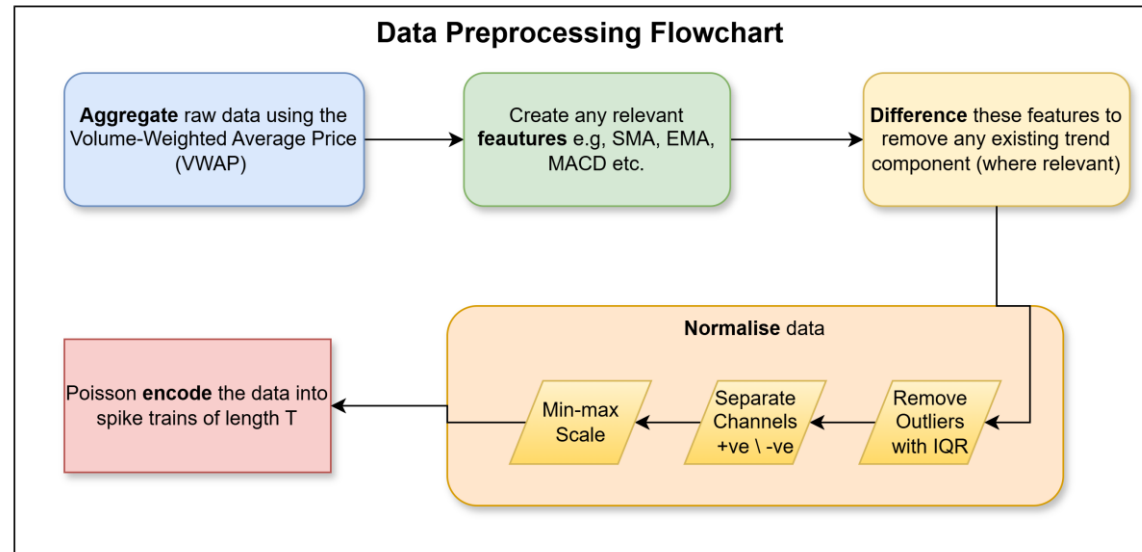
# Our approach

- **Enhanced Temporal Inputs:**
  - The input layer is expanded to incorporate multiple time lags for price differences
- **Explicit Inhibitory Connections:**
  - Cross-connections are integrated from X1 to H2 and from X2 to H1.
- **Inverted STDP Rule:**
  - Inhibitory synapses are governed by an inverted STDP update rule where correlated pre- and post-synaptic firing strengthens the inhibitory effect.
- **Normalisation:**
  - A three-step approach - channel separation, clipping to the 10th and 90th percentiles to mitigate outlier distortion, and final min-max scaling.



# Training procedure

- **Preprocessing**
  - Data aggregation with VWAP
  - Poisson Rate Encoding
  - 20 timesteps per spike train
- **Training & Regularisation**
  - STDP & Inverted STDP
  - Synaptic Homeostasis for weight regularisation
  - Threshold based decoding
- **Rolling evaluation:**
  - Uses day-by-day rolling window
  - Prevents future information from entering
- **Hyperparameter optimisation:**
  - Bayesian Optimisation with PSA objective
  - PSA yields more balanced spiking rates
  - SA optimises for precision – encourages conservative firing



$$W(\Delta t) = \begin{cases} A_+ \exp(-\Delta t/\tau_+) & \text{if } \Delta t > 0 \\ -A_- \exp(\Delta t/\tau_-) & \text{if } \Delta t < 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{PSA} = \text{Spike Accuracy} \times \text{Penalty Factor}$$

$$\text{Spike Accuracy (SA)} = \frac{N_{\text{predicted real}}}{N_{\text{predicted}}}$$

# Results

- **PSA beats SA**

- PSA generally yields better trading results than SA

- **Unsupervised beats Supervised**

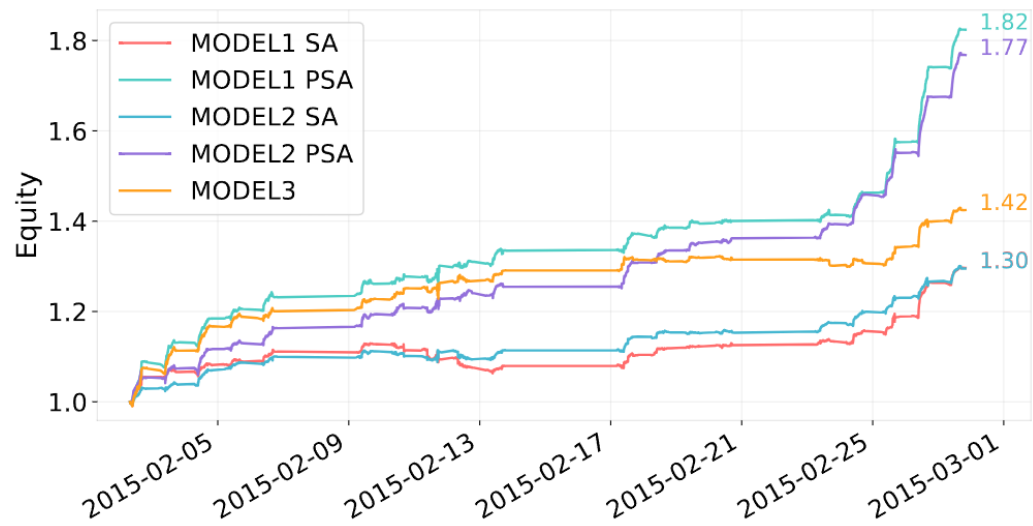
- STDP trained models were able to outperform the supervised and naïve baselines

TABLE IV  
TRADING PERFORMANCE COMPARISON ACROSS MODELS (RETURNS SCALED TO 1, 000 TRADES PER DAY)

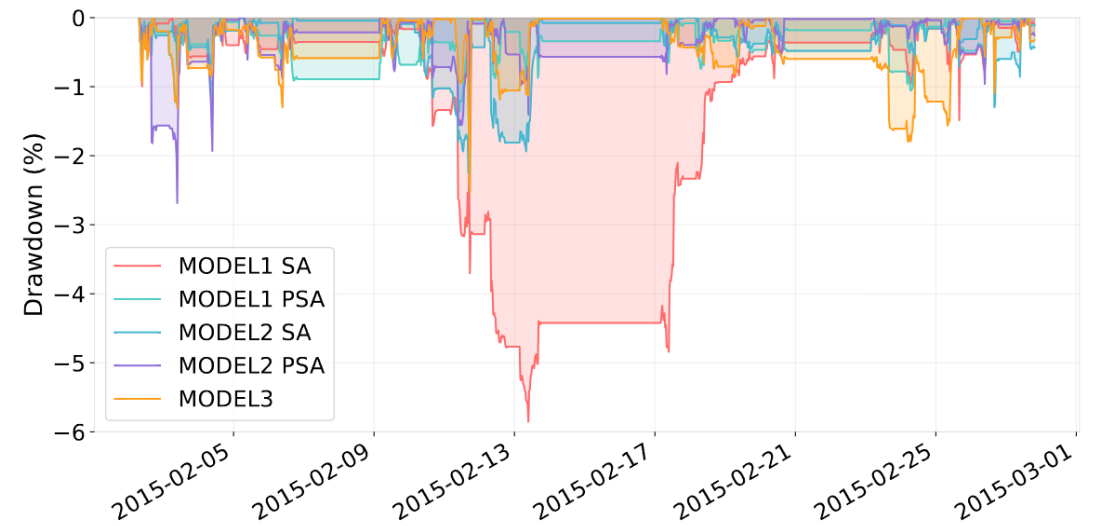
Metric	Model 1		Model 2		Model 3	Naive	Random
	SA	PSA	SA	PSA			
Cum. Return (%)	15.49	15.48	13.63	<b>17.44</b>	12.44	13.49	12.71
Sharpe Ratio	10.70	16.48	16.72	<b>19.72</b>	15.91	16.32	17.99
Max. DD (%)	5.85	2.52	2.25	2.69	2.95	<b>1.76</b>	2.77
Win Rate (%)	52.63	52.60	52.89	<b>53.49</b>	52.79	52.42	52.28
Profit Factor	1.15	1.21	1.16	<b>1.22</b>	1.13	1.18	1.17
Profit-Loss Ratio	1.00	<b>1.09</b>	1.03	1.06	1.01	1.07	1.07
Expectancy ( $\times 10^{-6}$ )	8.15	8.15	7.17	<b>9.18</b>	6.55	7.10	6.69

# Results

## Equity Curve



## Drawdown Curve



# Conclusion

## Findings

- STDP-trained SNNs, when robustly tuned with task-specific objectives like PSA can yield impressive results.
- Explicit inhibitory competition (Model 2) successfully isolates moments of clear price momentum.

## Future work

- Standardizing SNN-based metrics in computational finance and deploying these models on neuromorphic hardware for ultra-low latency execution.
- Conducting live experiments to further validate the techniques and architectures introduced here