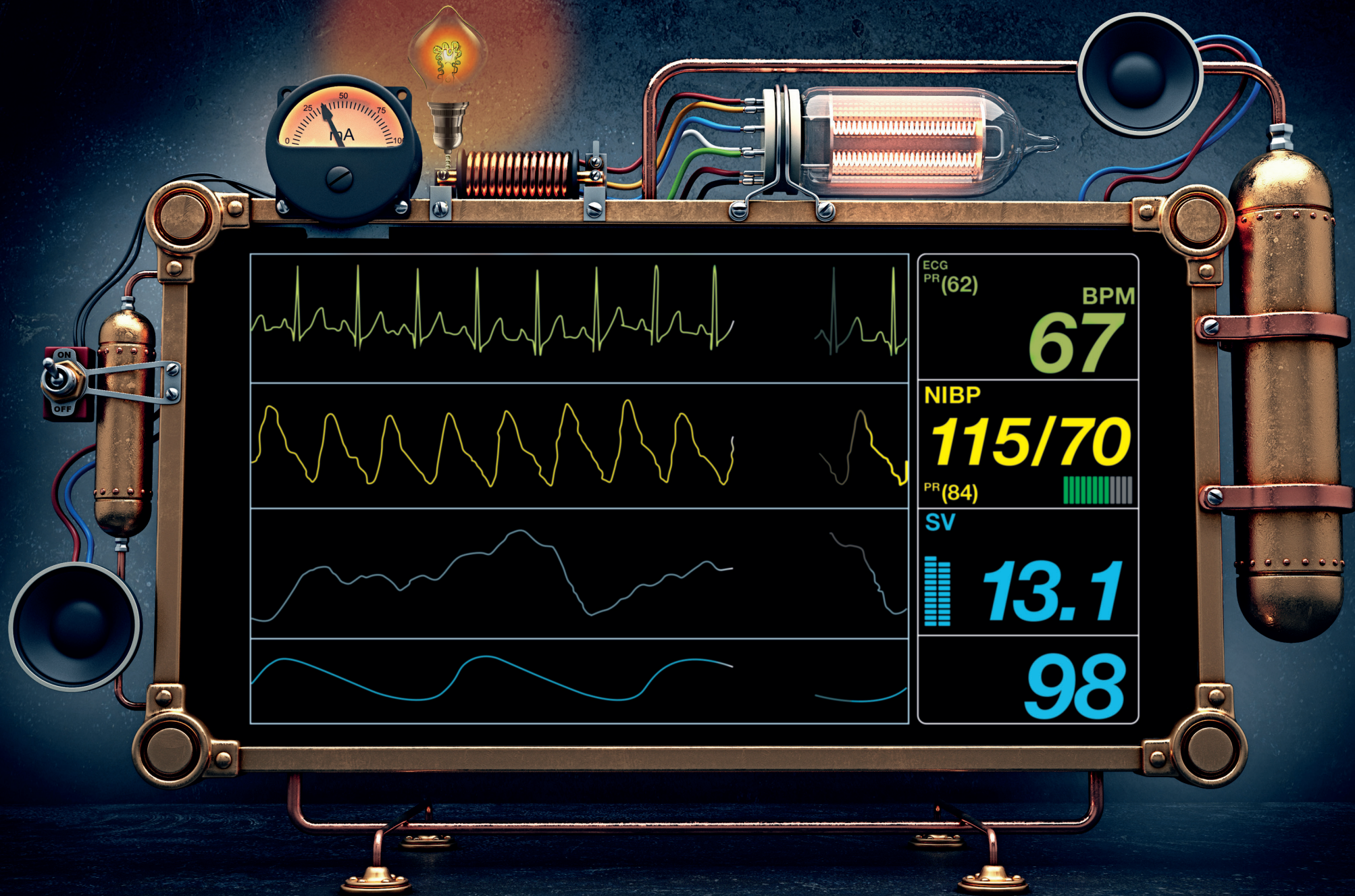


High-Frequency Trading and Financial Time-Series Prediction with Spiking Neural Networks

A novel new method is introduced – “unsupervised spike learning” – which predicts spikes in price time series instead of price movement and direction. Three rewarding high-frequency trading strategies are developed and backtested by Kang Gao, Wayne Luk, and Stephen Weston.



SPIKING NEURAL NETWORKS

This article presents a powerful and innovative approach to using spiking neural networks for financial time-series prediction. The novelty of our approach, which we call “unsupervised spike learning,” is that it predicts spikes in the price time series instead of price movement and direction. We adopt the unsupervised spike-timing-dependent-plasticity (STDP) learning method, such that teaching signals are no longer required. A new model topology for spiking neural networks is adopted in our approach and several spike definitions are presented. Based on this new approach, three rewarding high-frequency trading strategies are developed and backtested. Using intra-day high-frequency tick data for crude oil futures and gold futures prices, the results of various experiments using unsupervised spike learning are reported. The experimental results show that this approach performs well, achieving high accuracy in predicting price spikes. Moreover, our experiments demonstrate that using the STDP learning rule, a spiking neural network can recognize patterns hidden in historical price time series and can also be used to predict price spikes. We also report the results of applying novel intra-day high-frequency trading strategies that combine our unsupervised spike learning approach with traditional trading strategies. These new trading strategies are backtested in the crude oil futures market. The Sharpe ratio of each strategy is more than 20, showing that the new strategies are continuously profitable.

Introduction

Financial markets have been evolving for hundreds of years. Techniques for predicting the prices of financial instruments trading on the markets have been of great interest to theoreticians and practitioners alike. Such financial instrument price prediction remains a challenging task due to the non-stationary feature of the financial time series, as well as the wide range of factors that can impact the asset price. The distribution of asset returns is consequently also very difficult to model, being generally non-stationary and skewed. Despite – or perhaps because of – such challenges, the domain of financial instrument price prediction remains one of the most popular fields of research, with significant scope both for research and for commercial applications.

With the development of computer technology and its application by both traders and exchanges, together with the evolution of market micro-structure, price quotation and trade execution are consistently getting faster. In modern exchanges and trading firms, millions of orders and trades can be executed in a matter of seconds, enabling high-frequency/low-latency trading. Generally, high-frequency trading aims to submit large volumes of orders and execute trades at extremely high speeds, thereby profiting from adding up a large number of small quick profits. According to the law of large numbers, high-frequency trading is continuously profitable as long as the winning rate is more than 0.5. In the high-frequency world, the price is no longer affected by macroeconomic and political factors, where many factors are highly unpredictable. High-frequency financial time series tend to be dominated over extremely short timescales by market microstructures and the trading behavior of market participants, both of which are affected in turn by the relatively large number of electronic market participants. As a result, modern statistical and machine learning methods tend to be well suited to modeling high-

frequency financial time series, compared to low-frequency time series such as daily returns. This article focuses on exploring the applicability of statistical learning methods to predicting high-frequency financial time series.

In the past few years, artificial neural networks (ANNs) and deep learning have experienced significant success in many applications, from natural language processing and machine translation to computer vision and autonomous vehicles. However, though ANNs are inspired by the human brain, such networks operate quite differently from the biological brain. In ANNs and deep learning, linear layers and non-linear activation functions are used to form perceptrons. In addition, non-linear techniques, such as convolution and recurrence, are used in ANNs – unlike in the human brain. In contrast, in the human brain, neurons use spikes and timing to convey information to complete the learning task. Further, there is no backpropagation in the human brain, whereas most ANNs use backpropagation to learn statistical patterns in training data. To mimic the brain more closely, spiking neural networks (SNNs) were introduced. SNNs use spikes and spike timing to convey information and apply different learning rules – an approach which turns out to be far more biologically plausible. Consequently, SNNs have considerable potential for many applications. Moreover, SNNs have been implemented in hardware with high performance and low power consumption (Cheung *et al.*, 2016), enabling, for example, data centers to accelerate SNN applications based on field-programmable hardware technology.

As ANNs and deep learning are increasingly pervasive, there is a growing body of research concerned with the applications of deep learning in financial time-series prediction. However, there is limited research concerning the application of SNNs in the field of financial time-series prediction, and the work that is available is mainly concerned with supervised learning to predict the direction of the price series. Unfortunately, due to the lack of an effective learning method and the non-stationary nature of financial time series, existing research continues to exhibit limitations, with only a few publications

In the past few years, artificial neural networks (ANNs) and deep learning have experienced significant success in many applications, from natural language processing and machine translation to computer vision and autonomous vehicles

demonstrating results capable of practical application. Specifically, we believe there are two important challenges that need to be addressed:

- To demonstrate the widespread applicability of SNNs for the task of financial time-series prediction.
- To develop practical applications based on SNNs for financial time-series prediction and for risk management.

There is significant academic and industrial value of a proven approach to applying SNNs for financial time-series prediction, which is the main motivation for this article. We address the above two challenges, focusing on spiking neural networks in financial time-series prediction, by applying an innovative unsupervised spike learning method to the development of high-frequency trading strategies. Our contributions to addressing the above challenges are as follows:

- An innovative method called “unsupervised spike learning” for addressing challenge one. To predict price spikes in financial time series, this method applies unsupervised learning to training spiking neural networks, achieving high predictive accuracy in forecasting price spikes.
- Three-spike learning-based trading strategies for addressing challenge two. Each strategy is developed by combining our unsupervised spike learning method with a traditional momentum-related trading strategy. These strategies have been shown to be continuously profitable in the backtesting period for intra-day high-frequency trading. The Sharpe ratio of each strategy is more than 20 in backtested commodity futures.

Related work

Maass (1997) provides a thorough and complete computational model for spiking neural networks, showing that the spiking neural network is a universal function approximator. Since then, a huge amount of research work has been conducted and spiking neural networks have been successfully applied in various fields of machine learning and deep learning, making them the third-generation neural networks. However, there has been little research on adopting spiking neural networks for financial time-series prediction.

Sun *et al.* (2016) use spiking neural networks to forecast the carbon price time series in the InterContinental Exchange. They propose a model that combines the variational mode decomposition method and spiking neural networks. The carbon price time series is firstly decomposed into various relatively stable components through the variational mode decomposition method and then these components are input to the spiking neural network to obtain predictions. The learning algorithm employed for training spiking neural networks is spike propagation, which is a supervised learning algorithm proposed by Bohte *et al.* (2002). Simulation results suggest that their proposed joint model outperforms conventional models; however, the contributions made solely by spiking neural networks are not quantified. Another research work on spiking neural networks in finance is that by Reid *et al.* (2014). A novel type of spiking neural network, the polychronous spiking network, is proposed. During training, segments of price time series are input into spiking neural networks, and the label for a corresponding segment is the actual price movement direction. A supervised training method is used for training the spiking network. The proposed spiking neural network is tested on three financial time series: IBM stock data, US/Euro exchange rate,

There is significant academic and industrial value of a proven approach to applying SNNs for financial time-series prediction, which is the main motivation for this article

and Brent crude oil price data. The performance of spiking neural networks surpasses the performance of traditional multi-layer neural networks. Trading strategies are developed using the prediction results produced by polychronous spiking networks, and the performance is better than their baseline.

Background

Spiking neural network basics

SNNs were originally introduced to model neuron dynamics in the human brain due to their biologically realistic features. The SNN approach is based on three key components, namely spiking neurons, spike trains, and synaptic plasticity.

Spiking neuron

The spiking neuron mimics the neurons in the human brain. Spiking neurons are connected to other spiking neurons of other layers, with specified membrane potential and a threshold. The neuron accepts input from other neurons, which can increase (excitatory) or decrease (inhibitory) membrane potential. When the membrane potential exceeds the threshold of the neuron, it emits a spike. The most famous and popular model for spiking neurons is the leaky integrated-and-fire (LIF) neuron model. The dynamics of the LIF neuron model can be described by the following formula (Ponulak & Kasiński, 2011):

$$C \frac{du}{dt}(t) = -\frac{1}{R} u(t) + (i_o(t) + \sum_j w_j i_j(t))$$

Spike trains

A spike train is a form of language that encodes the information in the outside world or the data. A chain of 0s and 1s can illustrate this. For sample, in the sequence 00101000, the first two 0s are the normal stage before the first spike. Then a spike is emitted. After the first spike, the neuron returns to the former stage and after a small while (the 0), it spikes again. This sequence is called a spike train. Since the spikes are all in the same form for all the neurons, one spike cannot encode any useful information. Instead, it is the number of spikes and the frequency of spikes, as well as the timing of spikes, that convey the information.



SPIKING NEURAL NETWORKS

Synaptic plasticity

Synapses connect neurons, such that spikes are transmitted between neurons via synapses. Each synapse is associated with a weight, which is used to scale the incoming spike train. There are two types of synapses: excitatory and inhibitory. If the spikes transmit via an excitatory synapse, the membrane potential of postsynaptic neurons increases; otherwise, for inhibitory neurons, the membrane potential decreases. In biological brains, synapses are strengthened or weakened during the learning process, and this is known as synapse plasticity. Similarly, in spiking neural networks, during the training process the network adjusts the weights of the connections (synapses) so that the network learns to recognize certain patterns in the data.

SNN input encoding

Spike trains are the only carriers of information transferring inside the spiking neural networks. As a result, the inputs to spiking neural networks must also be spike trains. However, most of the time the raw input is a series of floating-point numbers, so an encoding scheme is required to convert the raw input into spike trains. The most popular method is the Poisson encoding method. Details about the Poisson encoding method can be found in Heeger and Heeger (2000).

SNN dynamics

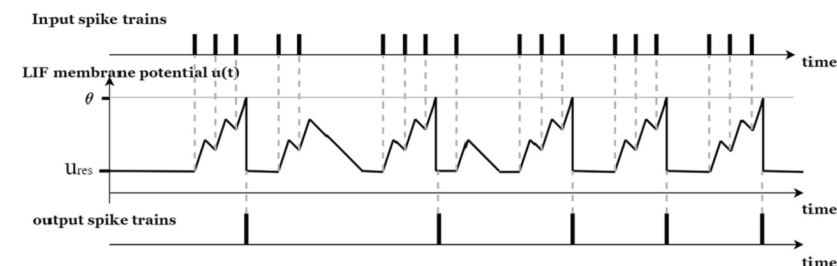
Figure 1 shows the dynamics of a spiking LIF neuron. In the beginning, the spiking neuron is at resting potential, and then the input spike trains arrive. The input spike trains provide stimulation, increasing the membrane potential of the neuron towards the threshold θ . Once the membrane potential exceeds the threshold θ , the spiking neuron emits a spike and the membrane potential is reset to resting level. The emitted spike trains are the input spike trains for the spiking neurons in the subsequent layer. In addition, for each timestamp, if there is no spike emitted, the membrane potential decreases by a small amount. In Figure 1, the middle part shows the change of membrane potential of a spiking neuron; in the lower part, the vertical bars represent the firing times of the spiking neuron.

Unsupervised STDP learning rule in spiking neural networks

Spike-timing-dependent plasticity (STDP), which is an unsupervised learning rule, is the most famous learning rule in spiking neural networks (Caporale & Dan, 2008). The main idea of STDP is to adjust the weights of the synapses according to the relative timing of their spikes. Intuitively, if a postsynaptic neuron spikes a little bit after a presynaptic neuron, the synapses between them are strengthened. On the contrary, if a postsynaptic neuron spikes a little bit earlier than a presynaptic neuron, the synapses between them are weakened. It has been shown that when used properly with appropriate encoding methods and network structures, STDP has mechanisms similar to those of expectation-maximization algorithms, so that it can learn the patterns of certain distributions. The mathematical formula for the STDP learning rule is (Tavanaei *et al.*, 2019):

$$\text{if } t_{pre} - t_{post} \leq 0$$

Figure 1: Schematic diagram for dynamics of LIF neurons in spiking neural networks. The diagram shows input spike trains, change of membrane potential, and output spike trains. Adapted from Ponulak and Kasiński (2011)



$$\Delta w = A \exp\left(\frac{-|t_{pre} - t_{post}|}{\tau}\right), \text{ where } A > 0$$

else

$$\Delta w = B \exp\left(\frac{-|t_{pre} - t_{post}|}{\tau}\right), \text{ where } B > 0$$

Unsupervised spike learning methodology

This section presents the innovative unsupervised spike learning method. The novel elements of the proposed approach are briefly listed as follows:

1. Use unsupervised learning instead of supervised learning.
2. Predict spikes in price time series instead of price movement directions.
3. Use innovative model topology, including local connections.
4. Use intra-day high-frequency data instead of daily close data.

Data

Description

High-frequency future-contract tick data are used in unsupervised spike learning experiments. We collected price tick data of two commodities traded on the Chicago Mercantile Exchange, crude oil futures and gold futures, for all 23 trading days in May 2017. The data comprise the transaction price and transaction size for all the contracts of crude oil and gold futures. Our experiments are carried out on dominant contracts, which are contracts with the most trading volume on a trading day.

Data pre-processing

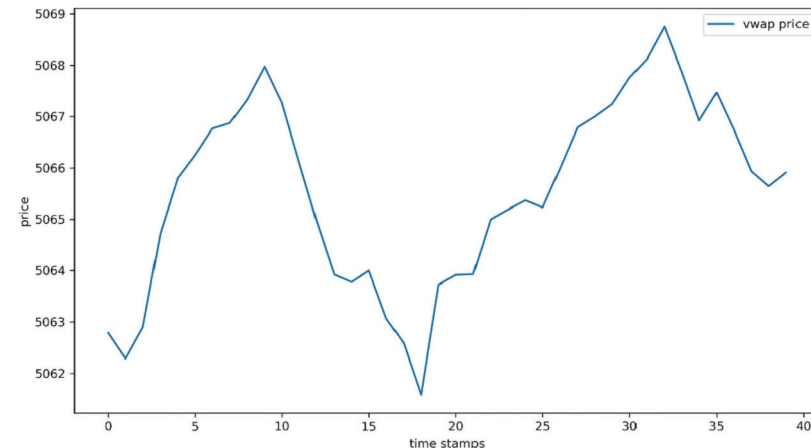
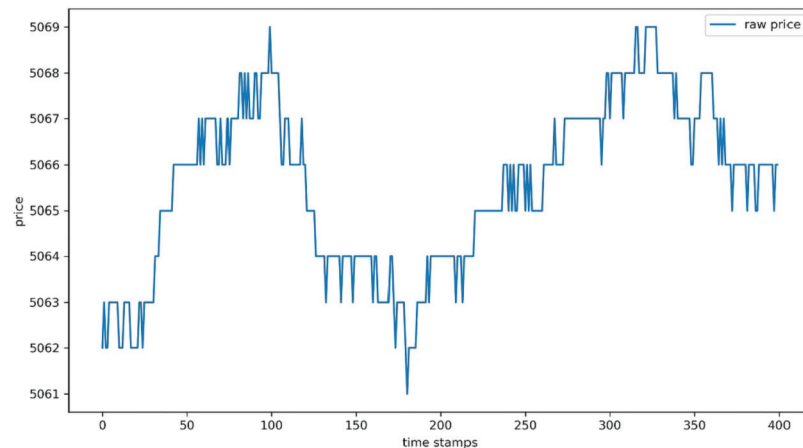
The raw price time series cannot be input directly to a spiking neural network. A series of pre-processing is conducted as follows.

Step 1: Volume-weighted average price. The volume-weighted average price (vwap) is used in the proposed unsupervised spike learning method. For a time period whose number of transactions is denoted by *num*, the formula for calculating the vwap is

$$vwap = \frac{P_1 * V_1 + P_2 * V_2 + \dots + P_{num} * V_{num}}{V_1 + V_2 + \dots + V_{num}}$$

SPIKING NEURAL NETWORKS

Figure 2: Comparison of the raw transaction price and the vwap. The two panels show the price move for the same contract for the same period. The vwap is smoother and the spikes are more obvious here. Note that the x-axis of the left panel is 10 times the x-axis of the right panel since we use window length 10 to aggregate the raw price into vwap.



The reason for converting the raw price to the vwap is that the vwap move is much more smooth and can avoid the “zig-zag” noise of the raw price. There is a large proportion of trading time in which the transaction price oscillates between the best bid and best ask prices, leading to the “zig-zag” behavior of the raw price. The noise resulting from these oscillations can affect the capability of a spiking neural network to identify price spikes and trends. Thus, converting to vwaps can significantly improve the performance of our spiking neural network. Besides, aggregating raw prices also greatly decreases the amount of data points, which makes training and testing faster. Furthermore, it binds volume information into the price series, which is more rational in an economic sense. Figure 2 compares the raw transaction price and the vwap.

Step 2: Price difference and negative price difference. The vwap is still not the input to the spiking neural network model. The next step is to take the price difference and the negative price difference between adjacent timestamps in the vwap time series. The reasons for this pre-processing technique are as follows.

First, differencing the price time series can remove the day-trend component in price time series so that a spiking neural network can focus on recognizing high-frequency intra-day local price spiking behavior. Even though the data in our unsupervised spike learning experiment are intra-day, meaning that there is no seasonal information in the price time series, there still exists a trend component in the price time series. This is the trend in the whole trading day. The trend component interferes with the recognition of local price information and causes a disturbance to the detection of price spikes. In Figure 3, unobserved components analysis is applied to price time series and price difference time series separately. The results show that differencing the price time series removes the trend component, so that the price difference time series is better than the price time series as input to spiking neural networks. For details about the unobserved components analysis method and the algorithmic process for generating Figure 3, refer to Seabold and Perktold (2010) and Durbin and Koopman (2012).

Second, the effect of the different magnitude of the price needs to be eliminated. The financial market is never static, sometimes the commodity price varies so much that there is a large gap between the open price and the close price. As a result, the input spike trains may have a systematic bias for different trading times. For example, if there is a great price drop in a day, the price near the opening time will have a much larger magnitude than the price near the closing time; this would cause the rate of the spike train in the beginning hour of trading to be systematically higher than the rate of the spike train in the hour before the market closing time. Consequently, the bias will lead to a systematic difference in spike frequency

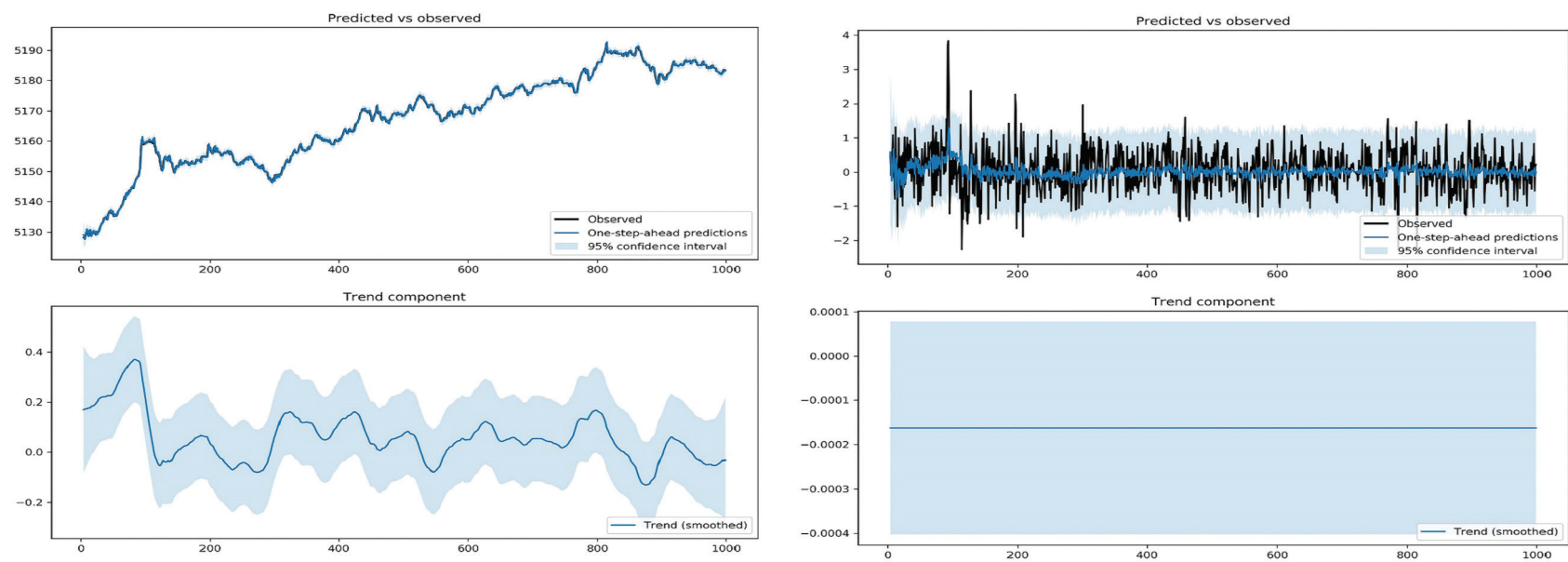
The financial market is never static, sometimes the commodity price varies so much that there is a large gap between the open price and the close price

for different trading hours. However, the price spikes are only related to the price change, not the magnitude of the price. Such bias can influence the performance of a spiking neural network. Taking price difference and negative price difference can eliminate this bias caused by the different price magnitudes. Using this technique, the input now represents the magnitude of the price change instead of the price itself, which is exactly what is needed to catch the price spike.



SPIKING NEURAL NETWORKS

Figure 3: Unobserved component model analysis for price time series and price difference time series. The left panel is for price time series and the right panel is for price difference time series. The figure shows that differencing the price time series removes the trend component in price time-series data.



For financial data, price spikes can be generated not only by price surges, but also by price plunges, hence both positive and negative price differences carry information. However, due to the features of spiking neural networks, their input data need to be normalized to non-negative intervals. If we feed only price difference data to the network, during data normalization, negative price trend information will be lost. Hence, we feed both positive price differ-

In the context of the proposed unsupervised spike learning method, one pitfall in the data normalization process is to use the min-max normalization method

ence and negative price difference into the spiking neural networks to identify price spikes due to both price surges and price plunges.

Step 3: Data normalization. The last pre-processing step is to normalize the data. The data normalization process is essential before input to spik-

ing neural networks. The reason is that the input trains to spiking neural networks involve Poisson encoding, which requires a positive rate, while the price change can be negative. The normalization approach in our unsupervised spike learning method is similar to z-score normalization, but the new mean value is positive, not zero. Given the target mean value and standard deviation, the formula for the normalization method is

$$z_i = \max \left(\frac{x_i - \text{mean}(\mathbf{x})}{\text{stdev}(\mathbf{x})} * \text{new_stdev} + \text{new_mean}, 0 \right)$$

where $\text{mean}(\mathbf{x})$ and $\text{stdev}(\mathbf{x})$ are the mean and standard deviation of the initial data, while new_mean and new_stdev are the mean and standard deviation of the data after normalization. new_stdev and new_mean are two hyperparameters. In particular, new_mean requires attention since it is directly related to the spiking frequency of a spiking neural network. Since the Poisson encoding requires the input to be non-negative, the max operator is applied to clip the negative value among normalized data to zero.

In the context of the proposed unsupervised spike learning method, one pitfall in the data normalization process is to use the min-max normalization method. The reason is that this method can be seriously affected by the extreme values in time-series data. For example, if there is an abnormally large value in the data, then after min-max normalization, most of the data will be in a small range near the new_min value, while this large value will be $\text{new_min} + \text{new_range}$. Hence, the normalized data would be affected by this abnormal value, which distorts the information contained in the price time-series data, resulting in an adverse impact on the predictive performance of spiking neural networks.

SPIKING NEURAL NETWORKS

Spike definitions

In this section, several important definitions about spikes in price data are introduced. The definition of price spikes here is not the same as the literal meaning of “spike.” These definitions will be used to evaluate the performance of the unsupervised spike learning method.

Real spike vs. fake spike

To identify the spikes in price movement, the first thing to do is to define the criterion for a “real” spike. Thus, when a spiking neural network emits a spike signal, we can judge whether this spike is a valid spike with this criterion, making it possible to calculate the accuracy of spike prediction. The intuition for classifying spikes as real spikes or fake spikes is mainly based on the return between adjacent price timestamps. If the average return in the extent of the spike is larger than a threshold, the spike is classified as a real spike,

The intuition for classifying spikes as real spikes or fake spikes is mainly based on the return between adjacent price timestamps

otherwise it is regarded as a fake spike. Since the price series can be both up and down in the whole interval of a spike, the absolute value of returns is used in our calculation, allowing a strong trend in the same direction and strong ripples (fluctuations) in both directions to be considered.

The criterion is related to the absolute value of the percentage return of the intra-day price series. The price data have been pre-processed and aggregated to produce vwap data. Then we use this time series to get all the percentage returns between adjacent vwaps. Taking the absolute value of the percentage return yields the required absolute percentage return. Let X_t denote the vwap time series at time t . The formula for the absolute percentage return is then

$$r_t = \left| \frac{X_{t+1}}{X_t} - 1 \right|, \quad \text{where } t = 1, 2, 3, \dots, n-1$$

Now, the r_t ($t = 1, 2, 3, \dots, n-1$) time series is the intra-day absolute return series of the vwap. We define the “pivot return” as the median of the r_t series:

$$r_{\text{pivot}} = \text{median}(r_t), \quad \text{where } t = 1, 2, 3, \dots, n-1$$

Notice that the median return of the entire day is used. Since the definition of a real spike is used for evaluation after the whole experiment, it does not introduce future information into the experimental process. When a spiking

neural network gives out a spike at a certain timestamp, we use the average vwap return (absolute value) of a certain time period immediately after the spike timestamp as the strength of this spike. So, for the spike S :

$$S_{\text{strength}} = \frac{|r_{t+1}| + |r_{t+2}| + |r_{t+3}| + \dots + |r_{t+\text{window}}|}{\text{window}}$$

In the above formula, t is the timestamp of the spike S , and window is the length of the period, based on the number of vwap data updates. If S_{strength} is greater than r_{pivot} , this spike is defined as a “real” spike, otherwise the spike is regarded as a “fake” spike. The intuition behind this criterion is that, for a spike to be real, the price needs to change drastically around the timestamp of the spike.

Momentum vs. reversion spike

Another classification of the price spike is determined by the vwap both before and after the timestamp of the spike signal. The two categories are called “momentum” spike and “reversion” spike. The motivation behind these definitions is as follows.

As before, we first get the $P_{\text{prior_avg}}$:

$$P_{\text{prior_avg}} = \frac{P_{t+\text{window}} + P_{t+\text{window}+1} + \dots + P_{t-2} + P_{t-1}}{\text{window}}$$

where t is the timestamp of the spike signal, and window is the length of the time period prior to the spike. Similar to $P_{\text{prior_avg}}$, we define $P_{\text{post_avg}}$ as the average vwap of a time period immediately after the timestamp of the spike signal:

$$P_{\text{post_avg}} = \frac{P_{t+1} + P_{t+2} + \dots + P_{t+\text{window}}}{\text{window}}$$


where t is the timestamp of the spike, and window is the length of the time period after the spike. Using P_{spike} to denote the price at the timestamp of the spike, we calculate the difference between $P_{\text{prior_avg}}$ and P_{spike} , as well as the difference between $P_{\text{post_avg}}$ and P_{spike} . The sign of the product of the two resulting differences is the foundation to classify the price spike as “momentum” or “reversion”:

$$\text{mom_rev_flag} = (P_{\text{prior_avg}} - P_{\text{spike}}) * (P_{\text{post_avg}} - P_{\text{spike}})$$

If mom_rev_flag is greater than 0, the spike is defined to be a reversion spike, otherwise the spike is a momentum spike. The insights for this definition are that, if mom_rev_flag is positive, meaning the price before and after the spike lies on the same side of the price at the timestamp of the spike signal, then the direction of the price move changes after the spike. However, if mom_rev_flag is negative, the two mean prices lie on both sides of the price at the spike signal time, meaning that the direction of the price movement doesn’t change after the spike.

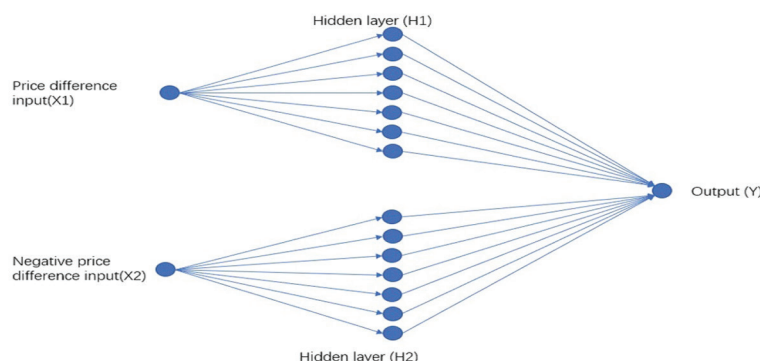
Network topology

Basic components

This article adopts the LIF model for the neurons in spiking neural networks. All neurons in a spiking neural network are of the same type. The connections between different layers are called synapses. Every synapse connects two 

SPIKING NEURAL NETWORKS

Figure 4: The double-input model topology. The blue solid circles are LIF neurons and the arrows are connections with the STDP learning rule. Spikes are conveyed in the directions of the arrows.



neurons between adjacent layers by a weight w , and the weights are updated according to the unsupervised STDP learning rule.

Double-input model

We adopt the double-input model for the proposed unsupervised spike learning method, which has a more complicated structure than a naive feedforward neural network. As it is called, the input layer has two neurons (called $X1$ and $X2$, respectively) and is of dimension two. The input to a double-input model is the price difference series and the negative price difference series, as discussed in the section on experimental results. It has one hidden layer and one output layer. The hidden layer is divided into two parts, $H1$ and $H2$. One of the neurons in the input layer is fully connected to all neurons in $H1$, while the other input neurons are connected to all neurons in $H2$. As a result, the topology is not in the complete fully connected style commonly found in tra-

After a neuron spikes, there is a refractory time for this neuron in which the neuron cannot spike again no matter what the potential is

ditional ANNs. The numbers of neurons in $H1$ and $H2$ are both hyperparameters, and in our experiments their default values are determined heuristically to be 64. As usual, no connection exists between neurons of the same layer; also, there is no connection between $H1$ and $H2$ in the hidden layer.

As for the output layer, it has only one LIF neuron. Only the spikes emitted by the neuron in the output layer are considered a spike signal of the network,

which is treated as the prediction for a price spike at that trading timestamp. The connections between the hidden layer and the output layer are of fully connected style, with synapses from all neurons in the hidden layer (including those in $H1$ and $H2$) to the output neuron. Since the hidden layer is divided into two parts and both parts are fully connected to the output layer, the two parts have a competing relationship for spikes due to the mechanism of the STDP learning rule. To some extent, each part plays an “inhibiting” role with regard to the other part. Figure 4 depicts the topology of our double-input model.

Simulation process

The whole simulation process of the proposed unsupervised spike learning method is composed of encoding the input into spike trains, propagating spikes along the network, and catching the output spike for analysis.

Input encoding

For double-input model topology, two time series are pre-processed. One is the price difference and the other is the negative price difference time series, obtained by multiplying the price difference by -1 . The pre-processing techniques are the same for the two time series. For every time series, Poisson encoding is carried out to generate spike trains. In the end, if we have N timestamps, and the simulation time is denoted by T , then we have $2N$ spike trains of length T as input to the spiking neural network. For each simulation step, two spike trains are input to the network.

Spike propagation and learning

After spike trains are input to the spiking neural network, the spikes propagate through the network. The propagation follows the direction of the connections inside the network, as well as the spiking rule of LIF nodes. Once a LIF neuron receives a spike, the membrane potential of that neuron increases by the corresponding weight of the associated synapse, and once the membrane potential exceeds the pre-determined threshold, the neuron gives out a spike, which is propagated by the synapse to the subsequent neuron. For every simulating time step, if the neuron does not emit a spike, the membrane potential decreases by a small amount, where the value of this amount is a hyperparameter. If a spike is emitted, the membrane potential of the emitting neuron is reset to the initial resting potential. After a neuron spikes, there is a refractory time for this neuron in which the neuron cannot spike again no matter what the potential is. The length of the refractory time is also a hyperparameter. As for learning, all connection weights are updated by the STDP learning rule.

Output and performance analysis

For the spiking neural network in our approach, the output layer has only one LIF neuron and only the spikes emitted by the output neuron are considered a price spike signal emitted by the spiking neural network. Throughout the simulation process, all price spike signals and the corresponding timestamp are recorded. Using the recorded price spike signals and the price time series, performance metrics of the method can be calculated to evaluate the prediction accuracy.

One important aspect of our approach is that no future information is used when predicting price spikes. In our unsupervised spike learning

method, the input to a spiking neural network is a time series. Thus, every price spike signal that a spiking neural network emits is based on a historical time series. No future information is available at that timestamp.

Performance metrics

The performance metrics in unsupervised spike learning are mainly about the percentage of several types of price spikes among all price spike signals. The most important metric is spike accuracy, defined to be the percentage of real spikes among all price spike signals emitted by the spiking neural network. For unsupervised spike learning, the ability to predict price spikes can be confirmed by high spike accuracy. Other metrics include the percentage of momentum spikes among all emitted spikes. The percentage of reversion spikes is not used, since it is given by 100 percent minus the momentum spike percentage, as a result of our definitions. The momentum spike percentage is important in shedding light on the logic in developing spike learning-based trading strategies. Table 1 summarizes the performance metrics used in the experiments.

Table 1: Performance metrics in the unsupervised spike learning method

Performance metrics	Calculation formula
Spike accuracy	Real spike number/total spike number
Momentum spike percentage	Momentum spike number/total spike number

Unsupervised spike learning experimental details and results

Two experiments are carried out to verify the effectiveness of the unsupervised spike learning method. The two experiments are the train-test experiment and the transfer experiment. The model topology and the main performance metrics used for each experiment are listed in Table 2.

Table 2: Model topology and main performance metrics used in the experiments of unsupervised spike learning

Experiment	Model topology	Performance metrics
Train-test experiment	Double-input model	Spike accuracy, momentum spike percentage
Transfer experiment	Double-input model	Spike accuracy

Experimental details

Train-test experiment

The train-test experiment applies a common method for testing the performance of the unsupervised spike learning method. The data are split into a training set and a testing set, where the training set is used to train the model while the testing set is used to test the model. In unsupervised spike learning experiments, the data are price time-series data of 23 consecutive trading days in May 2017. Since the experiments are focused on intra-day price behavior, the price series in one day is regarded as the smallest unit. Thus, the method to split the dataset here in the train-test experiment is to use the price time series in one day as the training set, and the data in the following day become testing data for the model trained. The training set and testing set

The momentum spike percentage is important in shedding light on the logic in developing spike learning-based trading strategies

are selected on a rolling basis. Specifically, the first model is trained using the price time series of the first day and tested on the second day's price time series. Next, the second day's price data become the training set and the trained model is tested on the price time-series data of the third day, and so on. In this way, the price time-series data of all trading days, except for the first and last day, will act as both training set and testing set, each for one time. The price time series of the first day only performs the role of training set, while the price time series of the last day only acts as testing set. In this way no future time series is used for predicting past time series.

Transfer experiment

The train-test experiment is carried out on both crude oil future price data and gold future price data. However, since the two time series are both commodity future price time series, common features should exist in their price behavior. Take technical analysis for example: the classical technique can be applied to all kinds of future price time series. It is of great interest whether the spiking neural network trained on one future price time series is able to predict price spikes in the price time series of another kind of future. To figure out this issue and to further investigate the potential financial time-series prediction power of our unsupervised spike learning method, a transfer experiment is carried out based on both price time series.

The fundamental idea for the transfer experiment is to train the spiking neural network on one future price dataset and test the trained spiking neural network on another future price dataset. For all trading days, both crude oil future price data and gold future price data are available. For every trading day, a spiking neural network is trained using the crude oil price data of that day, and the trained spiking neural network is tested on gold price data of the same day. After that, training data and testing data are interchanged. For each trading day, gold future price data becomes training data and crude oil future price data becomes testing data. Performance metrics on testing data for both rounds are calculated and the test results are evaluated.

Experimental results

The results for the three experiments are shown in Table 3. The results show that the spike accuracy is around 66 percent for crude oil and around 60 percent for gold in both training and testing experiments, which is significantly greater than 50 percent and illustrates the potential of our unsupervised spike



SPIKING NEURAL NETWORKS

learning method. The momentum spike percentage is continuously greater than 50 percent, which inspires the development of the strategies in the following sections. As for the comparison between the train-test experiment and the transfer experiment, there is no significant difference between the results of the two experiments, which shows that the two securities behave the same from a micro-transaction perspective.

Table 3: Results for train-test experiment and transfer experiment

Experiment	Category	Spike accuracy (avg)	Momentum spike percentage (avg)
Train-test experiment	Crude oil-train	66.07%	53.66%
	Crude oil-test	65.94%	54.00%
	Gold-train	58.92%	54.74%
	Gold-test	58.70%	54.52%
Transfer experiment	Crude oil-test(trained on gold)	65.83%	53.28%
	Gold-test(trained on crude oil)	59.78%	53.86%

Spike learning-based trading strategies and backtest performance

Strategy details

The novel unsupervised spike learning method addresses the first challenge mentioned in the introduction section. The method innovatively solves the prediction problem by predicting price spikes instead of price movement direction. Meanwhile, it applies the unsupervised STDP learning method to train the spiking neural networks, which no longer require teaching signals. The experimental results suggest that the method achieves high prediction accuracy in predicting price spikes, and the momentum spike percentage is

These three strategies are developed by combining the novel unsupervised spike learning method with classical trading strategies and technical analysis indicators

systematically greater than 0.5. Based on the successful results of the unsupervised spike learning experiments, three high-frequency trading strategies are put forward to address the second challenge. These three strategies are developed by combining the novel unsupervised spike learning method

with classical trading strategies and technical analysis indicators. The general procedure for establishing the novel strategies is to use price spike signals for strategic timing of entering a position, followed by the direction determination (long or short) according to the logic of traditional strategies. Specifically, the first step is to run the unsupervised spike learning method for target price time series. Whenever a price spike signal is generated, a position is ready to be established at the corresponding timestamp. The next step is to determine the direction of the position according to the logic of the traditional strategy. Finally, the position is entered and is held for a specific period of time.

The three strategies share some common assumptions. The initial capital value is 1. For the sake of convenient calculation of strategy profits, the order size of each order is the exact quantity such that the notional value of the order is 1. The order price is the vwap at the timestamp that is one step later than the price spike signal. Each position is held for n vwap timestamps and closed afterwards, using the vwap at the corresponding closing timestamp. The hyperparameter n in the backtest has value 3. The position is closed using market order should there be a remaining position at the end of each day.

Spike learning-based momentum strategy

The first strategy is called the spike learning-based momentum strategy, which is a combination of our unsupervised spike learning method and the traditional momentum strategy. The traditional momentum strategy believes that the strong movement in the market is most likely followed by another strong movement in the same direction. The price movement has a moving momentum, which is similar to the notion of “momentum” in physics. Thus, in momentum strategy a long position is established after a sharp price rise, while a short position is entered after a price plummet, hoping to profit from the momentum of the price movement.

When combined with traditional momentum strategy, the price spike signals in the unsupervised spike learning method are regarded as “momentum” signals for entering a position, where the direction of the position is determined by the price movement prior to the price spike signal. The logic is that a surge or plummet in price will cause a price spike, and the momentum of the price movement still exists after the price spike signal, thus the price will continue to move in the same direction as before. In this strategy, a price spike signal emitted by the spiking neural network leads to a portfolio position consistent with the direction of the prior price movement. Specifically, multiple spikes are emitted by the spiking neural network in our unsupervised spike learning method. For each spike signal, the strategy enters a position at the corresponding timestamp. The long/short direction of the position is determined by *position_flag*:

$$position_flag = \frac{P_{t+window} + P_{t+window+1} + \dots + P_{t-2} + P_{t-1}}{window} - P_t$$

where t is the timestamp when the spike signal is emitted, *window* is a parameter for the strategy, and the default value in backtest is 3. The procedure for determining the direction of the position is in Strategy Logic 1. Other configurations include those where the default holding period is three and no leverage is used.

SPIKING NEURAL NETWORKS

Strategy Logic 1: Spike learning-based momentum strategy

- 1: Calculate *position_flag*
 - 2: **if** *position_flag* > 0 **then** Enter short position
 - 3: **else if** *position_flag* < 0 **then** Enter long position
 - 4: **else** No transaction
 - 5: **end if**
-

Spike learning-based Alexanders filter strategy

The second strategy is called the spike learning-based Alexanders filter strategy, which is a combination of our unsupervised spike learning method and the Alexanders filter indicator. The logic of this strategy is similar to the logic of the first strategy; however, the new strategy uses a different indicator called the Alexanders filter to determine the directions of entered positions. The Alexanders filter is a simple technical analysis indicator defined mainly by a quotient of two prices:

$$ALF = \left(\frac{P_t}{P_{t-n}} - 1.0 \right) * 100$$

where *t* is the timestamp when the spike signal is emitted, *n* is a parameter for the strategy, and the default value in backtest is 1. The strategy follows the logic shown in Strategy Logic 2.

Strategy Logic 2: Spike learning-based Alexanders filter strategy

- 1: Calculate *ALF*
 - 2: **if** *ALF* > 0 **then** Enter long position
 - 3: **else if** *ALF* < 0 **then** Enter short position
 - 4: **else** No transaction
 - 5: **end if**
-

Spike learning-based stochastic oscillator strategy

The third strategy is called the spike learning-based stochastic oscillator strategy, which is a combination of our unsupervised spike learning method and the stochastic oscillator indicator. The stochastic oscillator, usually denoted %K, is also a technical analysis indicator with values ranging from 0 to 100:

$$\%K = \left(\frac{P_t - L_n}{H_n - L_n} \right) * 100$$

where *t* is the timestamp when the spike signal is emitted, *L_n* is the lowest price during the past *n* timestamps, and *H_n* is the highest price during the past *n* timestamps. Here, *n* is a parameter for this strategy, and the default value in backtest is 3. The strategy follows the logic shown in Strategy Logic 3.

Strategy Logic 3: Spike learning-based stochastic oscillator strategy

- 1: Calculate %K
 - 2: **if** %K > 50 **then** Enter long position
 - 3: **else if** %K < 50 **then** Enter short position
 - 4: **else** No transaction
 - 5: **end if**
-

Comparative performance of popular time-series models

To quantify the improvements and advantages of spike learning-based trading strategies over the trading strategies based on popular time-series models, strategies based on two popular time-series models are also backtested in the same price dataset. The two time-series models are the long short-term memory (LSTM) model and the autoregressive integrated moving average (ARIMA) model. For the LSTM model, future price movement direction is predicted based on past *n* observations of the price time series. The prediction is represented in the form of the probability of price rising. If the probability is higher than a threshold, a long position is established; if the probability is lower than another threshold, a short position is entered. The *n* and the two thresholds are all hyperparameters. The hyperparameters are tuned such that the LSTM-based strategy has similar average daily transaction numbers to the spike learning-based strategies. For the ARIMA model, the price of the next timestamp is predicted using past *n* observations of the price time series. If the predicted price is higher than the current price, a long position is established; otherwise, a short position is entered. An interval of fixed length is introduced between two predictions to make the ARIMA strategy have proper average daily transaction numbers. The two strategies work as the control group and their backtest performances are given below.

Comparative performance of naive versions of proposed strategies

Besides the performance comparison with popular time-series models, for each proposed spike learning-based trading strategy, a naive version of the corresponding strategy is also backtested in the same price dataset. This kind of comparison quantifies the improvement that the unsupervised spike learning method provides in each trading strategy. The naive version of each strategy removes the spike signal information provided by the unsupervised spike learning method; instead, transaction timestamps are randomly chosen from the price time series, ensuring that the transaction number is the same as in the corresponding spike learning-based strategy version. Take momentum strategy for example. If in one trading day the spiking neural network generates *n* spike signals such that the spike learning-based momentum strategy has *n* transactions, the naive version of momentum strategy randomly chooses *n* timestamps for trading transactions in this day, where at each timestamp the direction is determined by the rules of momentum strategy. The same logic applies to naive versions of the Alexanders filter strategy and the stochastic oscillator strategy.

Strategy performance

The three proposed strategies are backtested in crude oil future price time series for all trading days in May 2017. The price data in this month cover various market conditions. There are eight days when the price time series is in a trending market state, with four days of uptrending and four days of downtrending; for the remaining trading days the price time series fluctuates in the day to a varying extent. The data also incorporate both quiet market periods and volatile market periods. Using the high-frequency intra-data, the volatility for each trading day can be calculated. For the whole backtesting period, there are five days that have volatilities smaller than 20 percent and four days that have



SPIKING NEURAL NETWORKS

Table 4: Backtest performance of all strategies

Strategy	Accumulated return (one month)	Annualized volatility	Sharpe ratio	Win rate	Profit/loss ratio	Transaction number (daily avg)
Momentum – spike based	105.17%	59.17%	21.28	52.42%	1.022:1	2586
Momentum – naive version	76.51%	51.09%	17.91	51.20%	1.013:1	2586
Alexanders filter – spike based	134.14%	66.27%	24.24	53.05%	1.050:1	2586
Alexanders filter – naive version	98.41%	53.24%	22.12	51.39%	1.048:1	2586
Stochastic oscillator – spike based	119.89%	59.98%	23.94	52.84%	1.031:1	2586
Stochastic oscillator – naive version	78.21%	48.55%	19.27	51.03%	1.024:1	2586
LSTM model	101.72%	120.30%	10.12	53.53%	1.007:1	2158
ARIMA model	46.34%	32.21%	17.17	49.73%	1.016:1	2782

volatilities greater than 30 percent, with the remaining days having volatilities between 20 and 30 percent. Overall, the price data selected for backtesting are able to represent various market conditions and different market regimes.

The corresponding naive versions of the strategies and the strategies based on two popular time-series models are backtested in the same price dataset. To reduce uncertainty in backtesting performances of the naive strategy versions, each naive strategy is backtested 100 times and the performance metrics are calculated based on the average statistics of the repetitive backtesting. All the performance statistics are listed in Table 4. It turns out that the backtest results of the three strategies are incredibly remarkable. Each strategy makes continuous profits in the consecutive 23 trading days, with the

In practice, the three-spike learning-based strategies are all analogous to a strategy called “scalping,” popular in the futures markets

Alexanders filter strategy accumulating the most profit and having the highest Sharpe ratio. For profitability and Sharpe ratio, the proposed spike learning-based trading strategies perform much better than both LSTM and ARIMA strategies. Compared with the LSTM strategy, the spike learning-based strategies have similar win rates, but higher profit/loss ratios and lower volatilities. Though the ARIMA strategy has lower volatility, the proposed strategies have higher win rates and profit/loss ratios. The comparison with naive strategy versions shows that the spike learning-based strategy versions are systematically better than the corresponding naive strategy versions, which confirms the significant contributions that the unsupervised spike learning method makes for the proposed trading strategies.

Insights and evaluation

Unsupervised spike learning experimental results and strategy performance analysis

The experimental results involving the unsupervised spike learning method show that the method performs very well. The spike accuracy is around 66 percent in crude oil future and around 60 percent in gold future. Due to the usage of “median return” in defining “real spike,” the bottom line for the performance of any method is 50 percent. The spike accuracy results in these experiments, which are significantly more than 50 percent, confirm that the unsupervised spike learning method is capable of predicting price spikes precisely.

The momentum spike percentage is about 54 percent in our experiments. We found this performance metric to be stably greater than 50 percent, which indicates that the unsupervised spike learning method tends to detect a price spike at an early stage. This could also be the reason for the good performance of trading strategies based on momentum or momentum-related indicators. The proposed three-spike learning-based trading strategies achieve strong performance in backtesting results, with the Alexanders filter strategy working best. All strategies at least double their net value in one month, with Sharpe ratios all being more than 20. The win rates are all greater than 50 percent and the profit/loss ratios are all greater than 1. As our spiking neural network generates thousands of trading signals per day, each strategy is able to make continuous small profits every day due to the law of large numbers. The comparison with popular strategies based on time-series models, as well as naive versions of the proposed strategies, also favors the proposed spike learning-based trading strategies.

In practice, the three-spike learning-based strategies are all analogous to a strategy called “scalping,” popular in the futures markets. Scalping profits from small price changes, usually at sub-minute level. A successful scalper has a relatively high ratio of winning trades versus losing ones, while keeping profits roughly equal or slightly bigger than losses. The spike learning-based strategies have many features in common with scalping. The win rates are greater than 50 percent, and the profit/loss ratios are slightly more than 1. However, in the proposed three-spike learning-based strategies, trading transactions happen at

the seconds level, not at the minutes level. By combining traditional strategies with our unsupervised spike learning method, the manual “scalping” is carried out automatically at higher frequency by computers.

Relationship to technical analysis

During the STDP learning process of the unsupervised spike learning method, neurons inside spiking neural networks become more and more selective to hidden patterns inside the inputs, which are salient and presented consistently in inputs (Masquelier & Thorpe, 2010). Consequently, the STDP learning rule makes the spiking neural network increasingly sensitive to patterns repetitively presented in the input. For the spiking neural network in our unsupervised spike learning method, the improved ability to recognize specific patterns results in successful predictions in financial price time series and profitable trading strategy. One hypothesis for this success is that in the proposed unsupervised spike learning method, the spiking neural network effectively performs technical analysis on the price time series. The transfer experiment described earlier in this article appears to confirm this hypothesis.

The price data in the transfer experiment involve different types of futures in training and testing. The spiking neural network is trained by crude oil future price and tested by gold future price, or vice versa. Results in Table 3 show that the testing results have no significant difference from the training results. This implies that spiking neural networks trained by one future price time series can be successfully applied to another kind of future price time series, even though there are different characteristics between the two future price time series. This is evidence supporting the existence of similar patterns in different future price time series.

There are many features in common between the innovative unsupervised spike learning method and technical analysis. Results in the transfer experiment provide evidence for the argument that the spiking neural network performs technical analysis in the context of our unsupervised spike learn-

The price data in the transfer experiment involve different types of futures in training and testing

ing method. Technical analysis is an analytical method for examining and predicting price movements in financial markets, using historical price time-series charts. The principle for technical analysis is that when a particular pattern appears in historical price charts, the following price movement is predictable. The economic reason for technical analysis is that special patterns in historical price series imply special conditions of factors that affect price movements, such as investor behavior, or market supply and demand. When it comes to our unsupervised spike learning method, spiking neural networks are trained to recognize specific patterns which repetitively appear in price


If a spiking neural network is trained to perform a task analogous to technical analysis, it must have extended capability to perform well in various future price time series

time series. The specific patterns found by spiking neural networks are analogous to the specific shapes in technical analysis. Thus, the learning process of spiking neural networks can be considered as performing technical analysis in historical price series. Technical analysis is not restricted to any asset class, thus it can be performed in different price time series. If a spiking neural network is trained to perform a task analogous to technical analysis, it must have extended capability to perform well in various future price time series. The success of the transfer experiment justifies this extended capability and shows the generalization power of the unsupervised spike learning method.

Conclusion and future work Summary of achievements

Inspired by SNNs and spiking behavior in financial price time series, this article investigates how to use spiking neural networks in financial time-series prediction. The innovative unsupervised spike learning method and the proposed spike learning-based trading strategies together successfully address the challenges introduced in the introduction section.

Our unsupervised spike learning method makes use of the unsupervised STDP learning rule. It no longer predicts the direction of price movement but predicts price spikes in price time series. In this way, it introduces a new approach in financial time-series prediction, without the need to label training data. Meanwhile, the unsupervised spike learning method deals with high-frequency intra-day transaction data, showing its capability to study high-frequency price behavior. Various spike definitions are detailed, and an innovative spiking neural network topology is proposed. Customized performance metrics are proposed to evaluate experimental results.

Experiments using the unsupervised spike learning method are carried out in price time series of crude oil futures and gold futures. Two experiments are carried out to evaluate spike learning performance and investigate the mechanism inside the unsupervised spike learning method. The train-test experiment is aimed at testing the performance of the unsupervised spike learning method, while the transfer experiment is aimed at finding out the principles of spiking neural networks and their learning process in the unsupervised spike learning method. The experimental results of the unsupervised spike learning method show that the method achieves excellent performance. 

SPIKING NEURAL NETWORKS

It is confirmed that the pattern recognition process inside the spiking neural networks is analogous to technical analysis in price time series.

As far as practical application of spiking neural networks for financial time-series prediction is concerned, three-spike learning-based trading strategies are proposed by combining the unsupervised spike learning method with traditional trading strategies. Each strategy is backtested on crude oil futures using historical intra-day price time series. Backtest results of the strategies are extremely encouraging, with Sharpe ratios uniformly greater than 20. The comparison with two popular strategies demonstrates and quantifies the advantages of the proposed spike learning-based trading strategies. The comparison with the corresponding naive versions of the strategies shows the improvements that the unsupervised spike learning method brings to trading strategies.

Future work

Despite the success of the unsupervised spike learning method, there are still three areas in which further research is needed. The first concerns understanding the spikes in price time series and the process of making more precise predictions. Currently, the proposed unsupervised spike learning method can indicate a spike, but is not capable of predicting a specific type of spike, for example, momentum spike or reversion spike. Only by comprehending the price spike behavior can we predict the specific type of spike precisely, which leads to a robust and profitable trading strategy.

The second area is to optimize the parameter tuning process when applying spiking neural networks to price time-series predictions in various assets and more complex assets. For example, different futures have different behaviors, so the hyperparameters can be very different in different futures. There are so many hyperparameters in the setting of spiking neural networks that it is quite difficult to find the optimum hyperparameters for a particular asset. The problem is even more difficult for more complex assets, such as options with various maturities and strike prices.

The third area concerns computational performance and efficiency for spiking neural networks. Given a requirement for accuracy, it is essential to accelerate the computations and optimize the run time in spiking neural networks to meet the requirements of high-frequency trading. In the context of high-frequency trading, the latency is at the microsecond level, or even the nanosecond level. The computing process inside spiking neural networks is required to be ultra-efficient to meet the requirement of high-frequency trading. The trade-off between accuracy and run time will be the focus of future research.

Acknowledgments

The authors thank Kaveh Aasaraai and Javier A. Varela for their excellent suggestions. The support of the UK EPSRC (Grant Nos. EP/L016796/1, EP/N031768/1, EP/P010040/1, and EP/S030069/1), Xilinx, and Intel is gratefully acknowledged.

REFERENCES

- Bohte, S. M., Kok, J. N., and La Poutre, H. 2002. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing* 1–4, 17–37.
- Caporale, N. and Dan, Y. 2008. Spike timing-dependent plasticity: A hebbian learning rule. *Annual Review of Neuroscience* 1, 25–46.
- Cheung, K., Schultz, S. R., and Luk, W. 2016. Neuroflow: A general purpose spiking neural network simulation platform using customizable processors. *Frontiers in Neuroscience* 9, article 516.
- Durbin, J. and Koopman, S. J. 2012. *Time Series Analysis by State Space Methods*. Oxford: Oxford University Press.
- Heeger, D. and Heeger, P. D. 2000. Poisson model of spike generation. <http://www.cns.nyu.edu/~david/handouts/poisson.pdf>
- Maass, W. 1997. Networks of spiking neurons: The third generation of neural network models. *Neural Networks* 9, 1659–1671.
- Masquelier, T. and Thorpe, S. J. 2010. Learning to recognize objects using waves of spikes and spike timing-dependent plasticity. *2010 International Joint Conference on Neural Networks*, pp. 1–8.
- Ponulak, F. and Kasiński, A. 2011. Introduction to spiking neural networks: Information processing, learning and applications. *Acta Neurobiologiae Experimentalis* 71, 409–433.
- Reid, D., Hussain, A. J., and Tawfik, H. 2014. Financial time series prediction using spiking neural networks. *PLOS ONE* 8, 1–13.
- Seabold, S. and Perktold, J. 2010. Statsmodels: Econometric and statistical modeling with Python. *Proceedings of the 9th Python in Science Conference*, pp. 57–61.
- Sun, G., Chen, T., Wei, Z., Sun, Y., Zang, H., and Chen, S. 2016. A carbon price forecasting model based on variational mode decomposition and spiking neural networks. *Energies* 9(1), 54.
- Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., and Maida, A. 2019. Deep learning in spiking neural networks. *Neural Networks* 111, 47–63.

About the Authors

Kang Gao is currently a PhD student in the Department of Computing at Imperial College London. His doctoral research covers modeling and simulation techniques in financial services. He takes a multidisciplinary approach that encompasses the fields of artificial intelligence, agent-based modeling, and financial mathematics. He holds a Master's degree in machine learning from Imperial College London.

Wayne Luk is Professor of Computer Engineering at Imperial College London and Director of the EPSRC Centre for Doctoral Training in High Performance Embedded and Distributed Systems. His research focuses on the theory and practice of customizing hardware and software for specific application domains, such as computational finance, climate modeling, and genomic data analysis. He pioneered the optimization of parametric hardware descriptions of neural network designs, and the acceleration of financial simulation targeting field-programmable technology. He is a fellow of the Royal Academy of Engineering, IEEE, and BCS.

Stephen Weston is a partner in the Risk Advisory practice at Deloitte, focusing on valuation and risk modelling, as well as AI. He has over 25 years experience in investment banking working with many of the largest investment banks, as well as hedge funds and start-ups. Immediately prior to joining Deloitte he spent four years at Intel focusing on modelling, fintech and HFT. His experience spans all areas of trading, risk management and quantitative research. In addition, he is also a visiting professor at Imperial College in computational finance. He holds a PhD in mathematical finance from London University. He also has a particular penchant for red trousers and bright socks.

