Price graphs: Utilizing the structural information of financial time series for stock prediction

Junran Wu , Ke Xu , Xueyuan Chen , Shangzhe Li **, *Jichang Zhao***

GIT: https://github.com/BUAA-WJR/PriceGraph
arXiv: https://arxiv.org/pdf/2106.02522.pdf

Google Scholar

Jichang Zhao

Associate Professor of Economics and Management, Beihang University
Email verificata su buaa.edu.cn - Home page

computational social sciences   complex systems

SEGUI

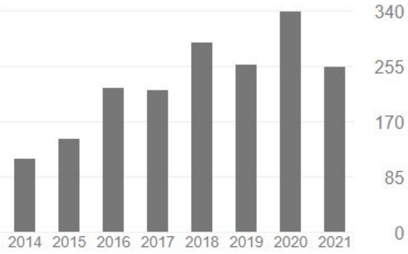
CREA IL MIO PROFILO

TITOLO	CITATA DA	ANNO
MD-MBPLS: A novel explanatory model in computational social science S Lu, J Zhao, H Wang Knowledge-Based Systems 223, 107023		2021
Price graphs: Utilizing the structural information of financial time series for stock prediction J Wu, K Xu, X Chen, S Li, J Zhao arXiv preprint arXiv:2106.02522		2021
Predicting long-term returns of individual stocks with online reviews J Wu, K Xu, J Zhao Neurocomputing 417, 406-418	2	2020
Fake news propagates differently from real news even at early stages of spreading Z Zhao, J Zhao, Y Sano, O Levy, H Takayasu, M Takayasu, D Li, J Wu, ... EPJ Data Science 9 (1), 1-14	61	2020
Can sentiments on macroeconomic news explain stock returns? Evidence form social network data Y Xu, J Zhao International Journal of Finance & Economics	1	2020
Trading imbalance in Chinese stock market—A high-frequency view S Lu, J Zhao, H Wang Entropy 22 (8), 897	1	2020

Citata da

VISUALIZZA TUTTO

	Tutte	Dal 2016
Citazioni	1974	1595
Indice H	19	17
i10-index	24	24



Accesso pubblico

VISUALIZZA TUTTO

7 articoli   28 articoli

Non disponibili   Disponibili

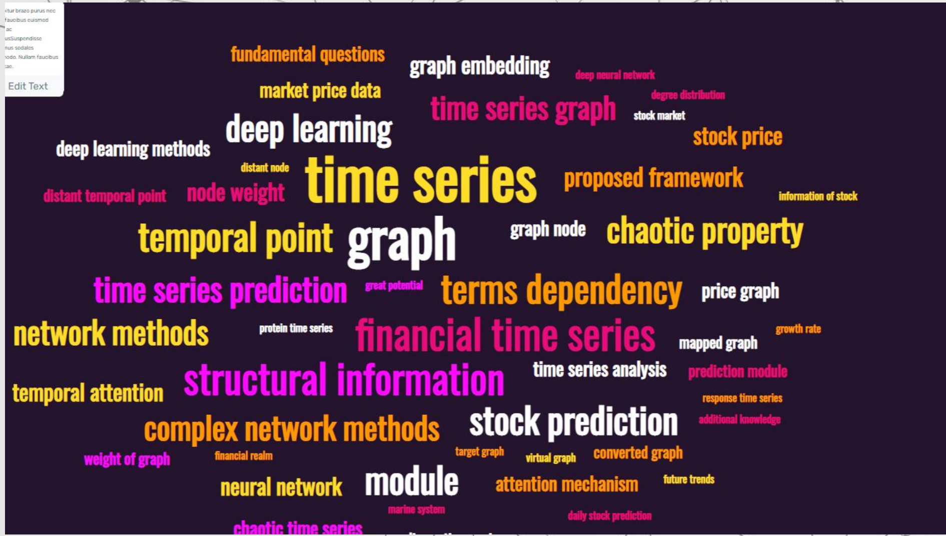
In base ai mandati di finanziamento

It's a PrePrint!

Under review…. submitted to Elsevier

Today we are going to talk about:

- time series
- graph
- forecasting
- cross asset relationship



Abstract

- **Goal:** Predict the rise or fall of stocks prices on a daily basis
- Binary Classification problem in a supervised learning environment

- **How**: Including the structural information of time series as highlighted by the **VG**, and modelling both graph representation (**struc2vec**)long term dependencies and the relationship between different stocks (**DARNN**)

- **Results:** promising techniques!

TABLE OF CONTENTS

01 Time series prediction

Stock prices prediction is modelled as time series forecasting using

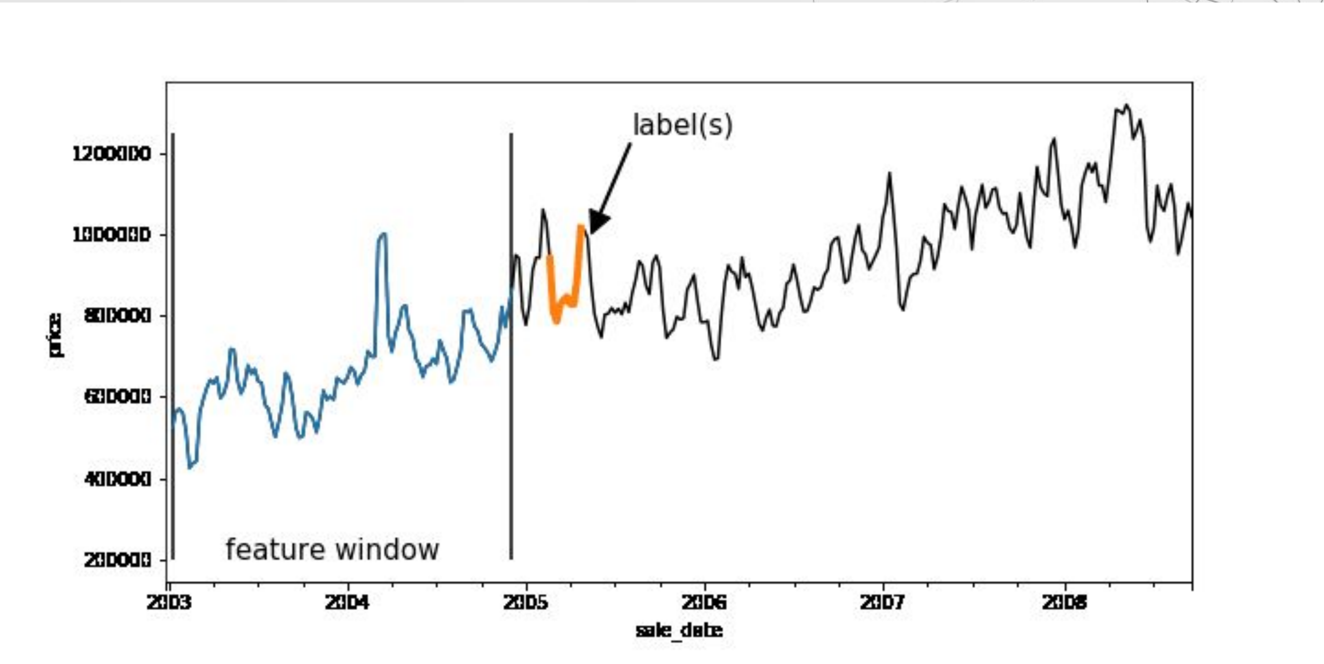
- **ARMA**: study correlation (in time), address the stationary, evaluate based on residuals
- **Deep Neural Network: Supervised learning in classification/regression problem (LSTM CNN, GCN, Attention)**
- Technical Analysis

Modelling the future based on the past behaviour!!!!

Example: Univariate TS forecasting

Define:

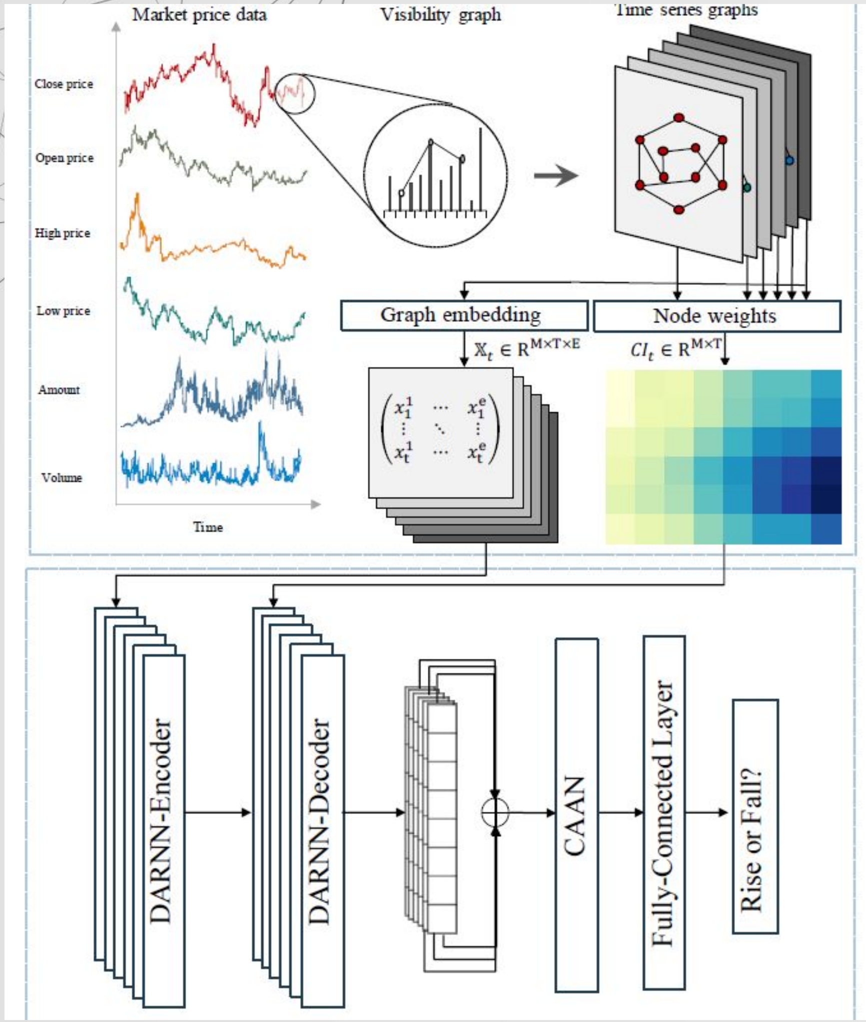
- Look back
- Single or Multi Step pred
- Input features (uni-multivariate TS)
- Regression or Classification



02 Model Components

2 Main Components:

- **Time Series Embedding:**
  - Visibility graph
  - Graph Embedding (struc2vec)
  - Nodes Weights (Collective Influence)
- **Prediction**
  - Multiple Dual Attention Recurrent nn (DARNN) layers (6)
  - Encoder-Decoder structure
  - Used in both E-D: LSTM + Attention
  - Cross asset attention



Time series embedding

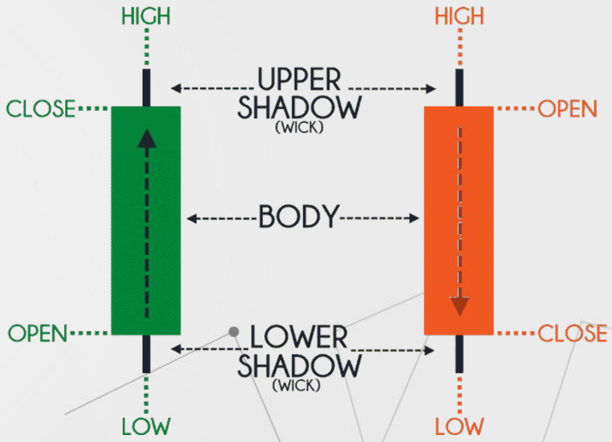
Looks like a preprocesing step

Prediction
Encoder Decoder and
cross asset attention
+
Fully connected

03 Time Series embedding

Input

- For each stock the following features : **OCHLV**
- Time series lookback: **T** (number of past step used to build the VG)
- Graph embedding size

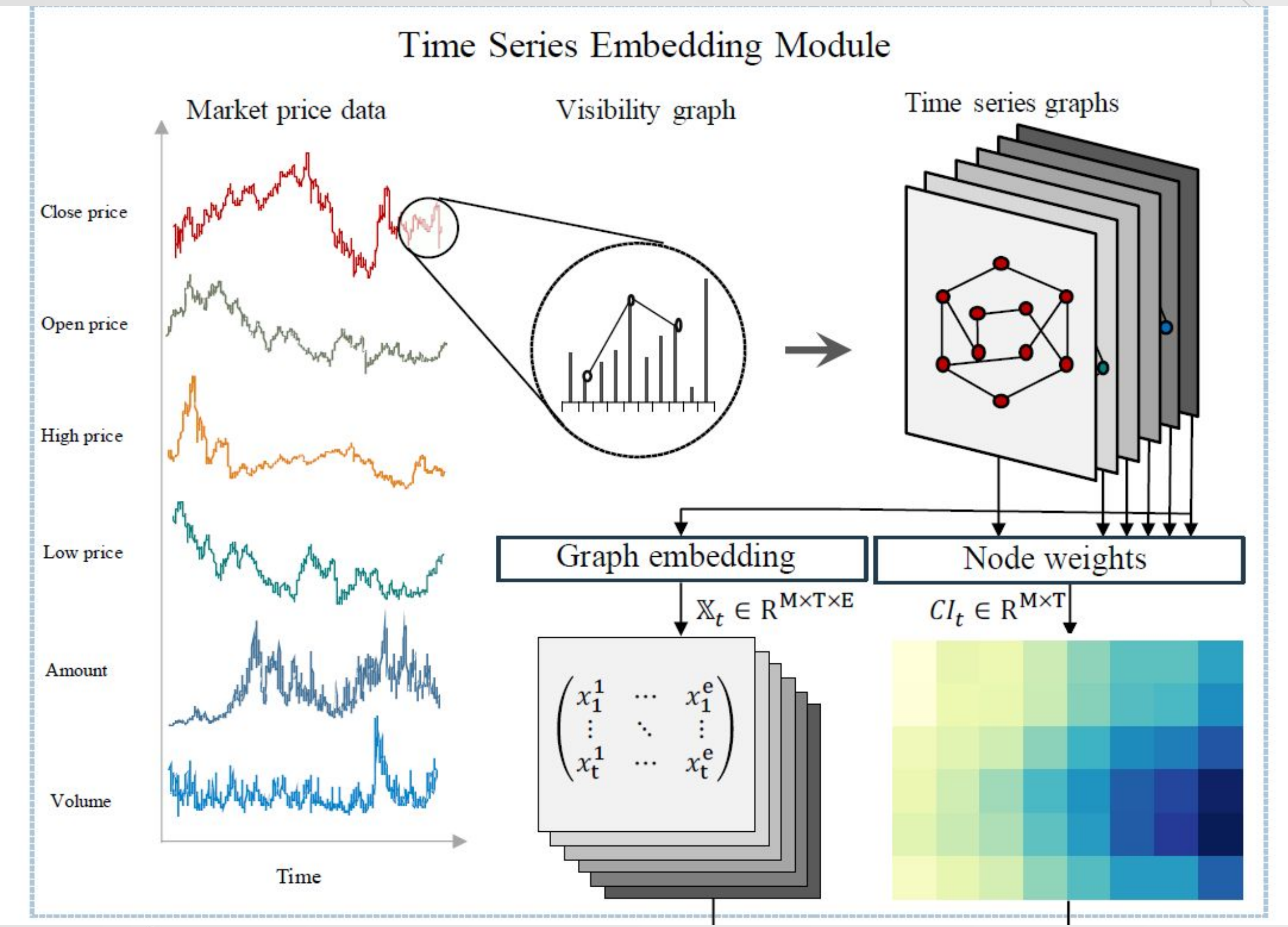


output

- Matrix with a given emb_size for each graph
- Nodes weights from Collective Influence

[1] ref

For each Stock

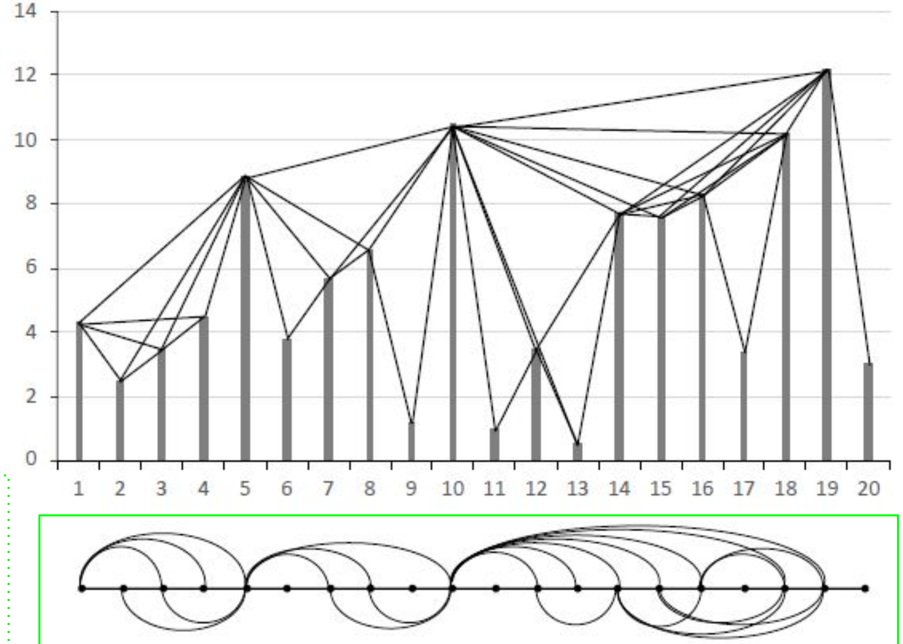


Visibility Graph: convert price data into graphs

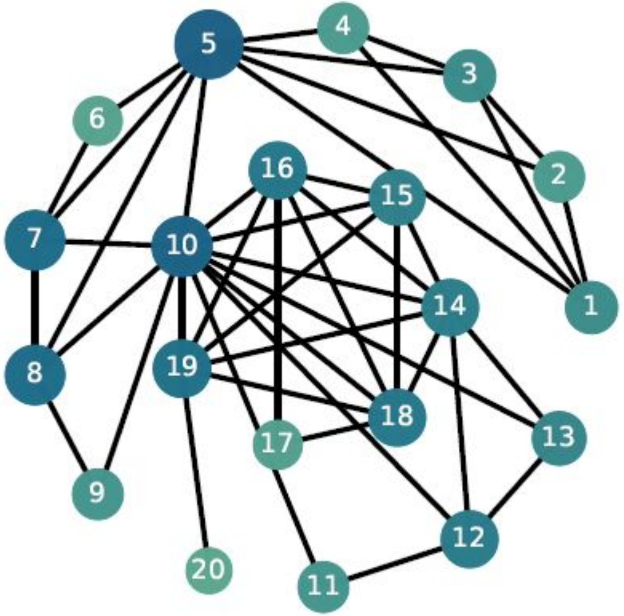
Goal: integrate the structural information and measure the weights of graph nodes (time instant)

Why: Edges among distant nodes models the long-term dependencies in TS and can directly capture the associations among distant temporal points.

How: maps time series into scale-free graphs. *Based on the principle that if two data points can mutually be seen in the bar chart of corresponding time series, an edge between the two points is established* edges allow new paths



(a) Visibility graph



(b) Time series graph with node weights

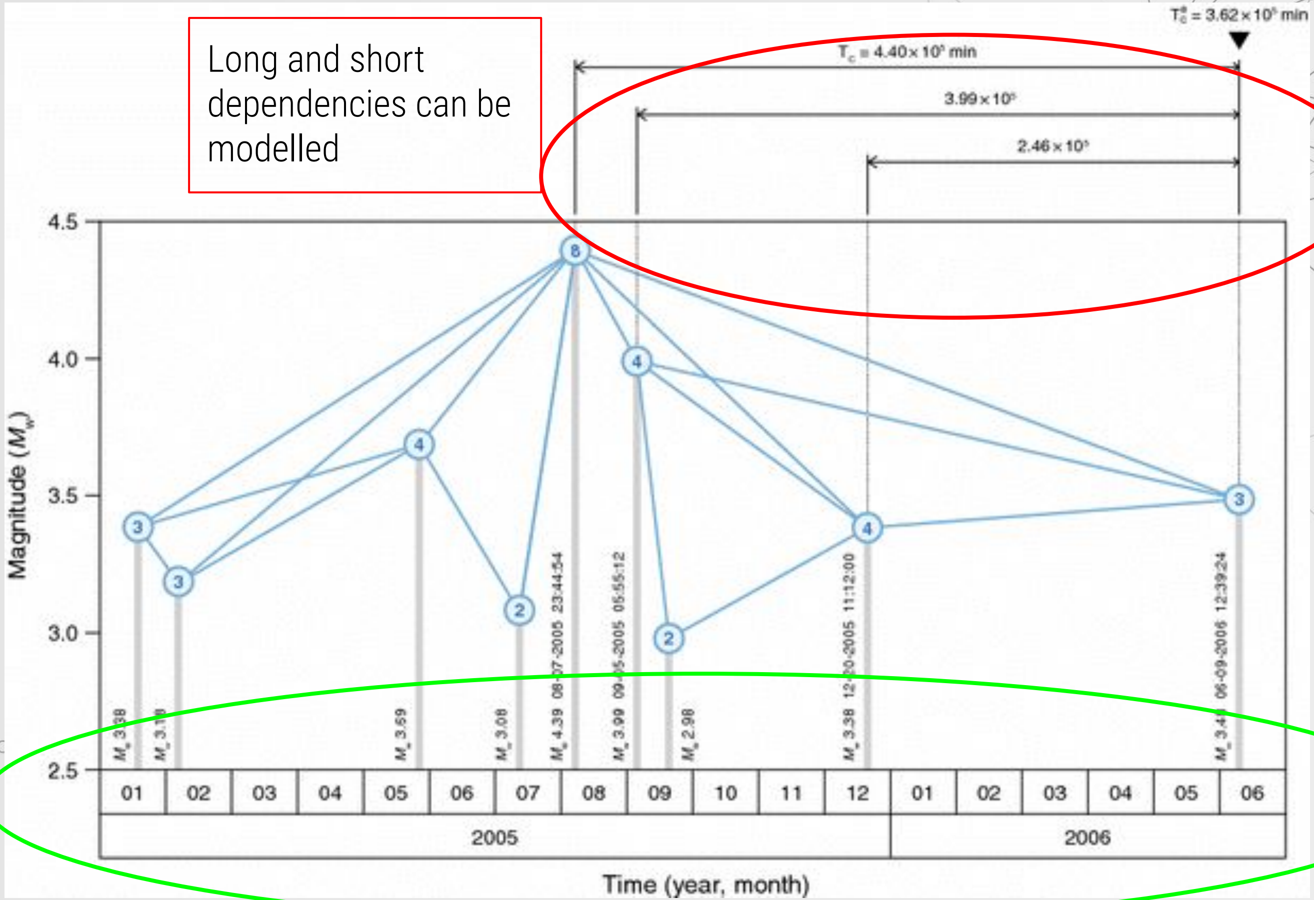
Figure 2: **Instance of a time series containing 20 temporal points and the associated VG derived from the VG algorithm.**

Advantages:

- Connected
- Undirected
- Unweighted
- Scale Free⟶ hubs

The **shortest path** between any two temporal points in the converted graph is definitely **shorter than in the original TS**

Long and short dependencies can be modelled



Each time step has a value

Nodes' weights: Collective Influence

$$CI_l(i) = (d_i - 1) \sum_{j \in \partial Ball(i,l)} (d_j - 1),$$

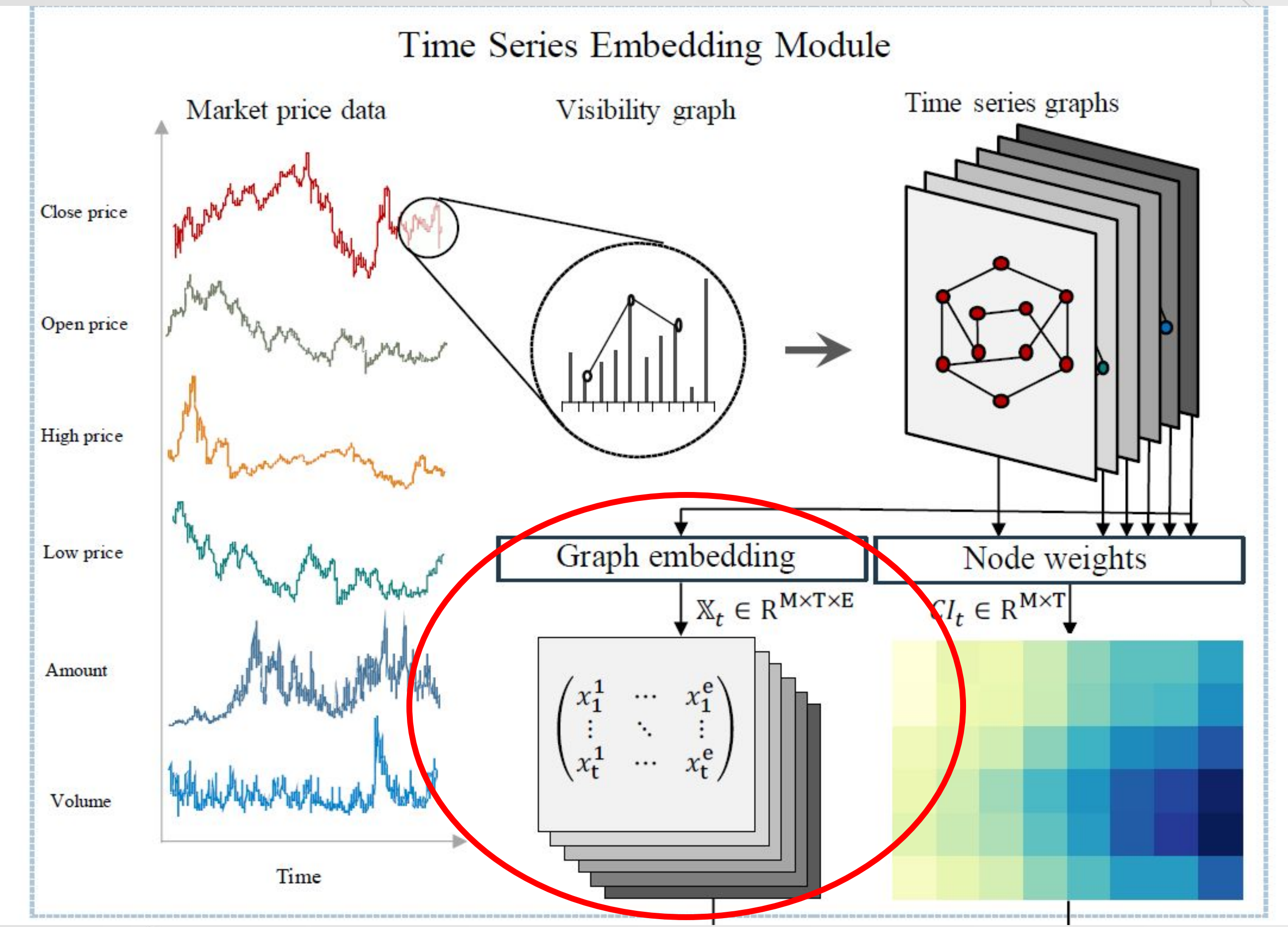
Why: Takes into account not only node's degree but also the degree of connected nodes.

How: Collective Influence is a metric which is better than simple node degree (local metrics)

Results: Greater the number of my friends and the number of my friends of friends, greater the CI value will be.

Which is a drawback of the Visibility Graph?

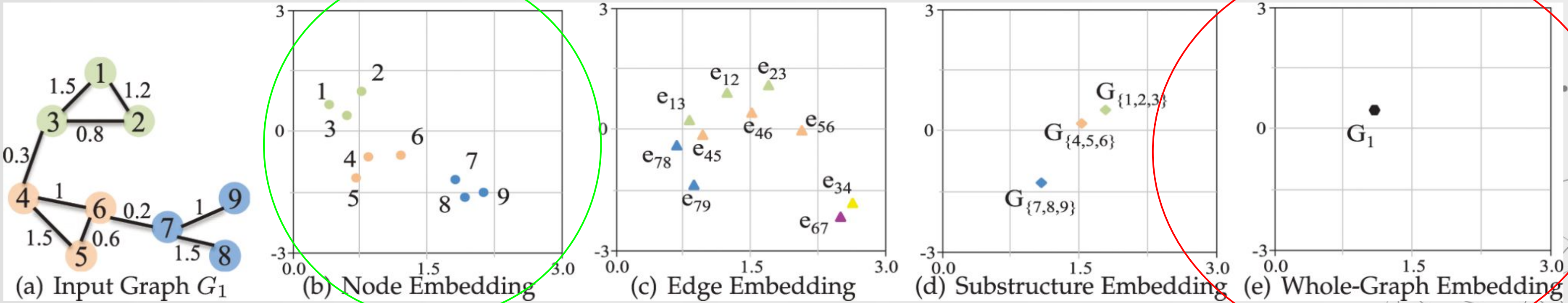
For each Stock



Node, edge, graph Embedding

Goal: obtain a vector representation of a graph/node edge based on the structural similarity.Embedding graph nodes into latent, low-dimensional spaces [

Why: Not only complements the loss of mapped price graphs in the temporal sequence but also incorporates structural information into deep learning methods.

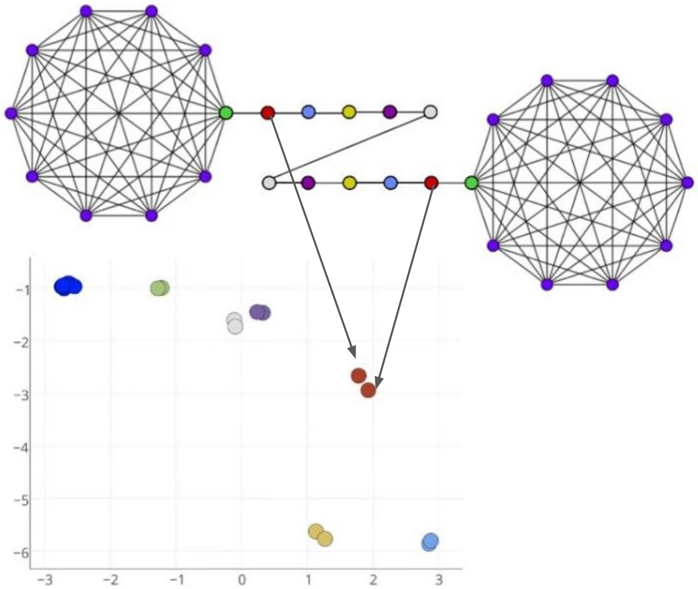


2d vector representation of each node

2d vector representation of the graph

Stuc2vec

struc2vec in Action



- Nodes with identical colors structurally equivalent
  - belong to automorphism
- Placed very close together in latent representation

- Useful for classification tasks that require roles

Graph Embedding

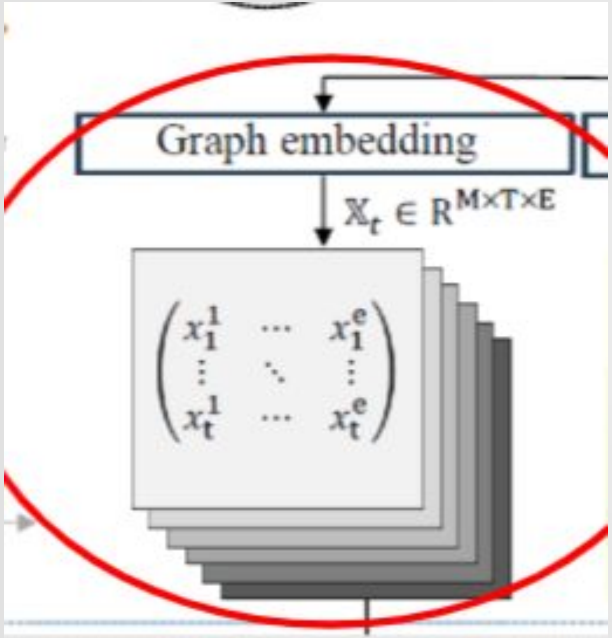
Based on nodes' k-hop neighborhoods,

- **struc2vec** is able to learn a vector-based representations that capture the structural roles of the nodes.
- **struc2vec** involves executing numerous random walks over the graph from each node.

The co-occurrences of nodes in a short window are captured based on the sequences of these walks, and the likelihood to appear in the same walk can be computed.

For each graph in each time step for each feature a tensor representation (size = emb_size) is obtained

Output of Struc2vec



M = 6 = OCLHVA
T= time steps
E = embedding size

Is this a huge preprocessing step?

....yes...

04 Prediction module

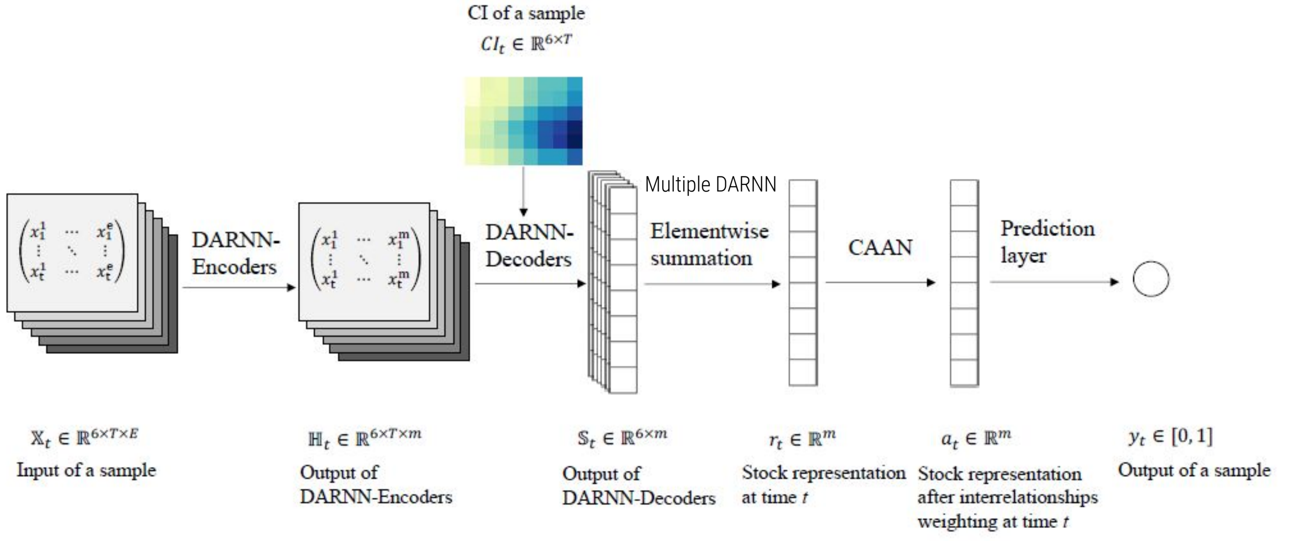


Figure 3: **Computational flow of the prediction module.** Here, we take a sample at time t for example. In which T is the lookback window of history prices at time t. E is the embedding size of struc2vec for each graph node. In experiments, we set hidden vectors of our deep learning method to a fixed size m.

[1] ref

More info about DARNN

[Blog with implementation]
[Official Paper]

DARNN (1)

- **Who:The Dual-Stage Attention-Based RNN (a.k.a. DARNN)** model belongs to the general class of *Nonlinear Autoregressive Exogenous (NARX)* models
- **What:** Predict the current value based on historical values of this series plus the historical values of multiple exogenous time series.
- A linear counterpart of is the ARMA model with exogenous factors
- **How:** DARNN is an ***encoder-decoder network*** and have the same learning process with different inputs.
- DARNN introduce an **input attention** mechanism to adaptively extract relevant driving series (a.k.a., input features) at each time step by referring to the previous encoder hidden state. **In the second stage, we use a temporal attention** mechanism to select relevant encoder hidden states across all time steps.

[1] ref

DARNN

Encoder

- Encoder stage, LSTM network encodes information among historical exogenous data, and its input attention (calculated through a deterministic attention model),performs feature selection to select the most important time series. OUTPUT: context vector(e)

$$[H_t^C, H_t^O, H_t^H, H_t^L, H_t^A, H_t^V] \in \mathbb{R}^{6 \times T \times m}$$

Decoder

- Take as input the encoder output, the CI (node weights), and THE ORIGINAL TIME SERIES
- After reweighting encoder output with CI, the combination of the updated context vector (e*)t with the stock price series.
- The updated version of the stock price with context vector is used as input for decoder LSTM

DARNN (2)

Decoder and Node Weights

Each node is a time step in each graph

For each node we have:

- Embedding Vector
- Collective influence

Context vector: The nodes that obtain critical positions in the time series (high CI) contribute more to the final sample representations (the hidden state of theencoder that is passed to the decoder)

[1] ref

CAAN

Cross Asset Attention Network (between stocks)

- WORKS WITHIN a batch!!!!
- Based on the stock representation(R_i) of stock i obtained from the Encoder-Decoder Module
- calculate three vectors (based on (R_i)) as the **query vector, key vector and value vector,** with learnable parameters.
- The interrelationships between stock i and other stocks within a batch are computed
- qi (query) by using the query vector of stock i to query the key vectors of other stocks.

[1] ref

Algorithm 1 Optimization of our prediction module within a sample

Input: input sample $\mathbb{X} \in \mathbb{R}^{6 \times T \times E}$, corresponding $CI \in \mathbb{R}^{6 \times T}$, historical price series $P \in \mathbb{R}^{6 \times T}$, labeled target y

Output: predicted label $\tilde{y}$ of input

Denote all parameters of the prediction model as $\mathbb{W}$;
Initialize the parameters $\mathbb{W}$;
for $epoch \in [1, \ldots, maxIteration]$ **do**
    $\mathbb{H} = \text{DARNN-Encoders}(\mathbb{X})$, where $\mathbb{H} \in \mathbb{R}^{6 \times T \times m}$;
    $\mathbb{S} = \text{DARNN-Decoders}(\mathbb{H}, CI, P)$, where $\mathbb{S} \in \mathbb{R}^{6 \times m}$;
    $r = \sum_{i=0}^{6} \mathbb{S}_i$, where $r \in \mathbb{R}^m$;
    $a = \text{CAAN}(r)$, where $a \in \mathbb{R}$;
    $\tilde{y} = f_{prediction}(a)$, where $\tilde{y} \in \mathbb{R}$; f = Linear + sigmoid
    // computing loss on a batch sample;
    $\mathcal{L} = \text{LossFunction}(y, \tilde{y})$;
    Update hidden parameters $\mathbb{W}$ with gradient decent $(\mathcal{L}|\mathbb{W})$;
end for

Description of the training process

05 Experiments

Data

- Feature of each stock : OCHLV: price are adjusted
- Daily quote data from the China A-share market from January, 2010, to December 2019 to cover comprehensive patterns in price trends and avoid the external shock from the COVID-19 on model validation.
- To prevent data snooping, experiment data sets are strictly split according to the sample dates

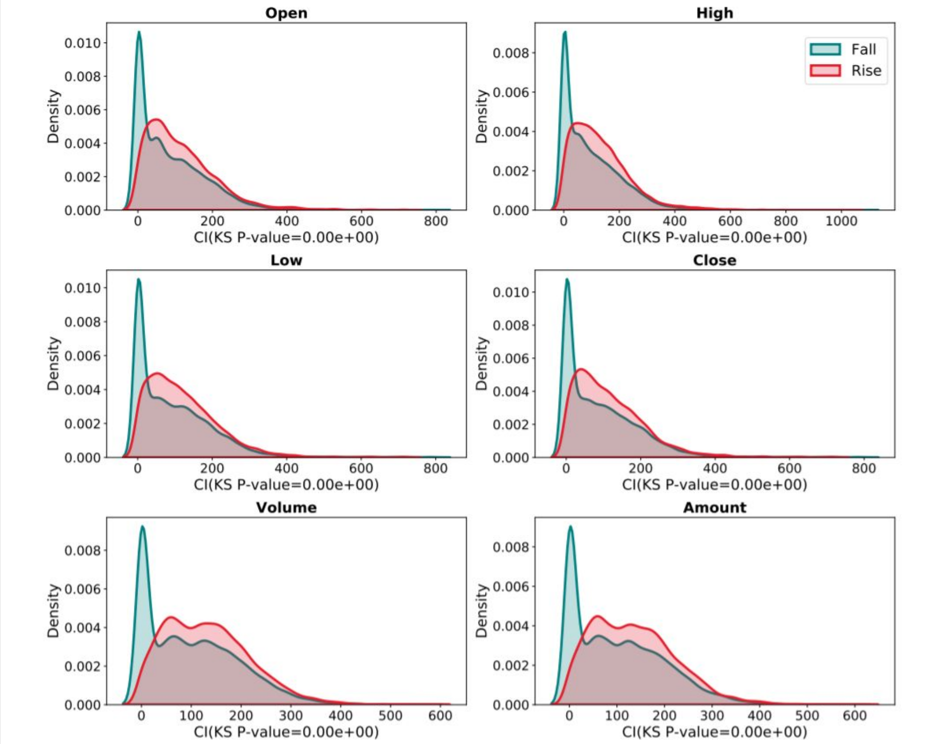
Baselines

- ARMA
- LSTM
- LSTM + CNN
- Fully Connected
- DARNN

$$y = \begin{cases} 1\ (\uparrow)\,, & p_{t+1}^c > p_t^c \\ 0\ (\downarrow)\,, & \text{otherwise,} \end{cases}$$

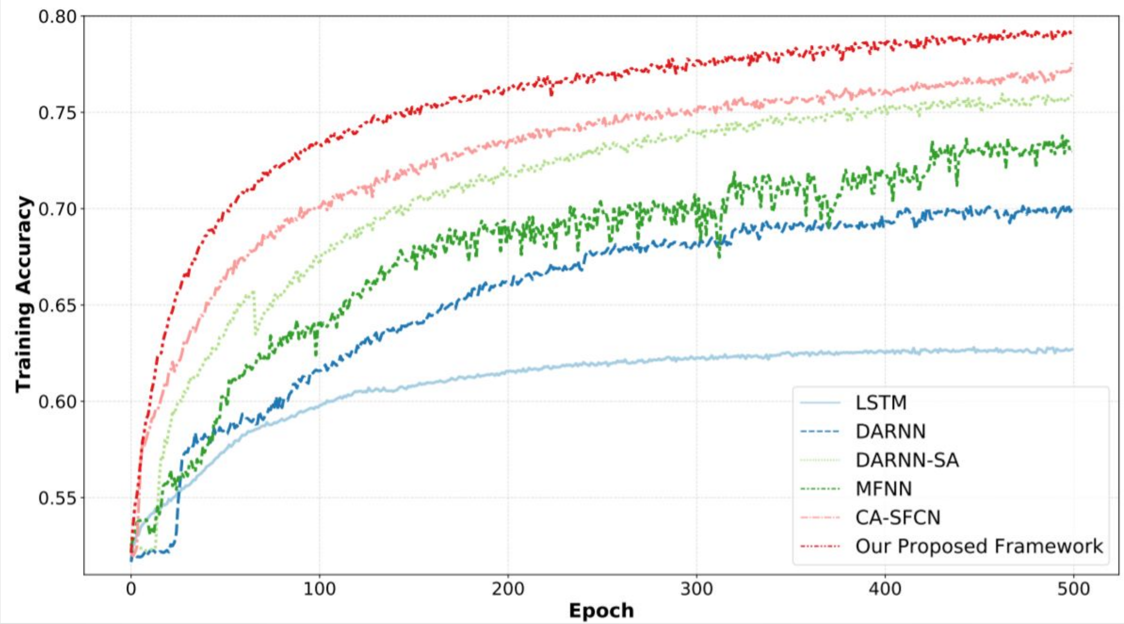
[1] ref

Data Exploration



Distribution of labels for each feature based on CI

It's TRAINING!



Test set performances

Table 2: **Results (%) of our proposed framework and the baselines.** All models predict price trend labels at the next time step. The best-performing results are highlighted with boldface. Our proposed framework outperforms all the state-of-the-art baselines on the test accuracies.

	2019(S1)				2019(S2)				2019(S3)				2019(S4)			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
ARMA	50.15	54.96	42.81	48.13	50.75	46.61	42.51	44.46	49.89	44.26	39.91	41.97	50.07	49.40	41.77	45.26
GARCH	50.28	54.90	44.68	49.26	50.66	46.74	45.74	46.23	50.43	45.05	41.76	43.34	50.36	49.75	42.24	45.69
LSTM	57.94	64.88	52.26	57.89	59.88	58.98	44.04	50.43	56.71	52.23	35.33	42.16	54.75	55.97	44.92	49.84
DARNN	60.87	68.27	54.69	60.73	62.03	61.48	48.29	54.09	60.62	58.10	61.33	59.67	61.54	68.34	63.31	65.73
DARNN-SA	64.32	71.72	58.63	64.52	66.23	66.60	54.40	59.89	65.47	65.00	59.09	61.9	65.63	72.34	68.92	70.59
MFNN	61.21	68.28	60.81	64.33	63.00	65.30	51.40	57.52	62.74	67.68	58.75	62.9	64.69	67.19	57.52	61.98
CA-SFCN	65.51	72.82	60.10	65.85	67.21	67.61	**73.52**	70.44	66.10	**77.81**	68.32	**72.76**	67.30	70.24	73.38	71.78
Our framework	**67.48**	**75.24**	**61.45**	**67.65**	**68.46**	**69.81**	71.67	**70.73**	**68.34**	67.86	**73.77**	68.09	**67.91**	**77.51**	**73.78**	**75.60**

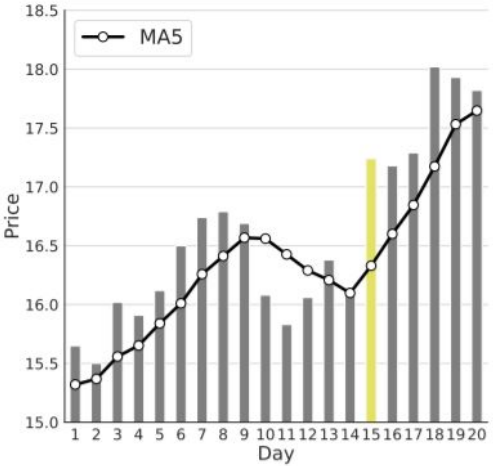
Notes. Precision, recall and the F1 measure are metrics calculated in the upward direction.

05 Experiments - params

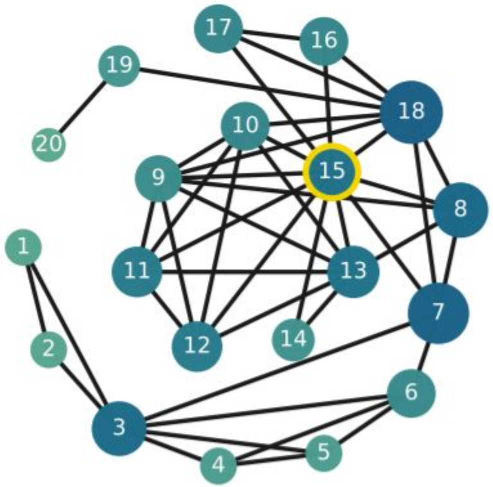
- Sizes of the hidden representations within {32, 64, 128, 256}
- Sizes of the mini batches within {32, 128, 256}
- Adam optimizer with an initial learning rate of 1e-3

[1] ref

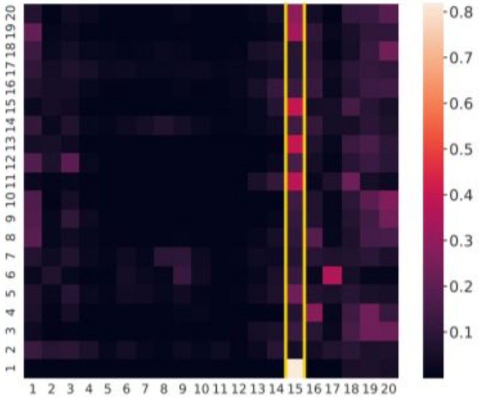
Ablation Study



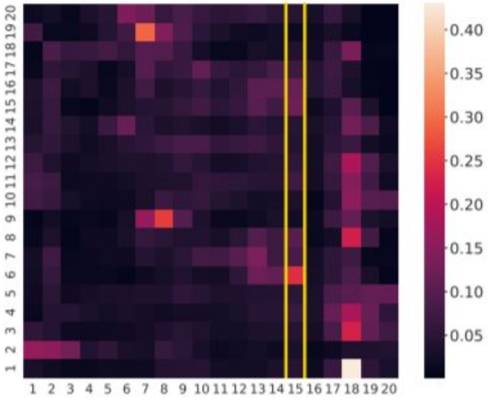
(a) Stock prices



(b) Price graph with CI

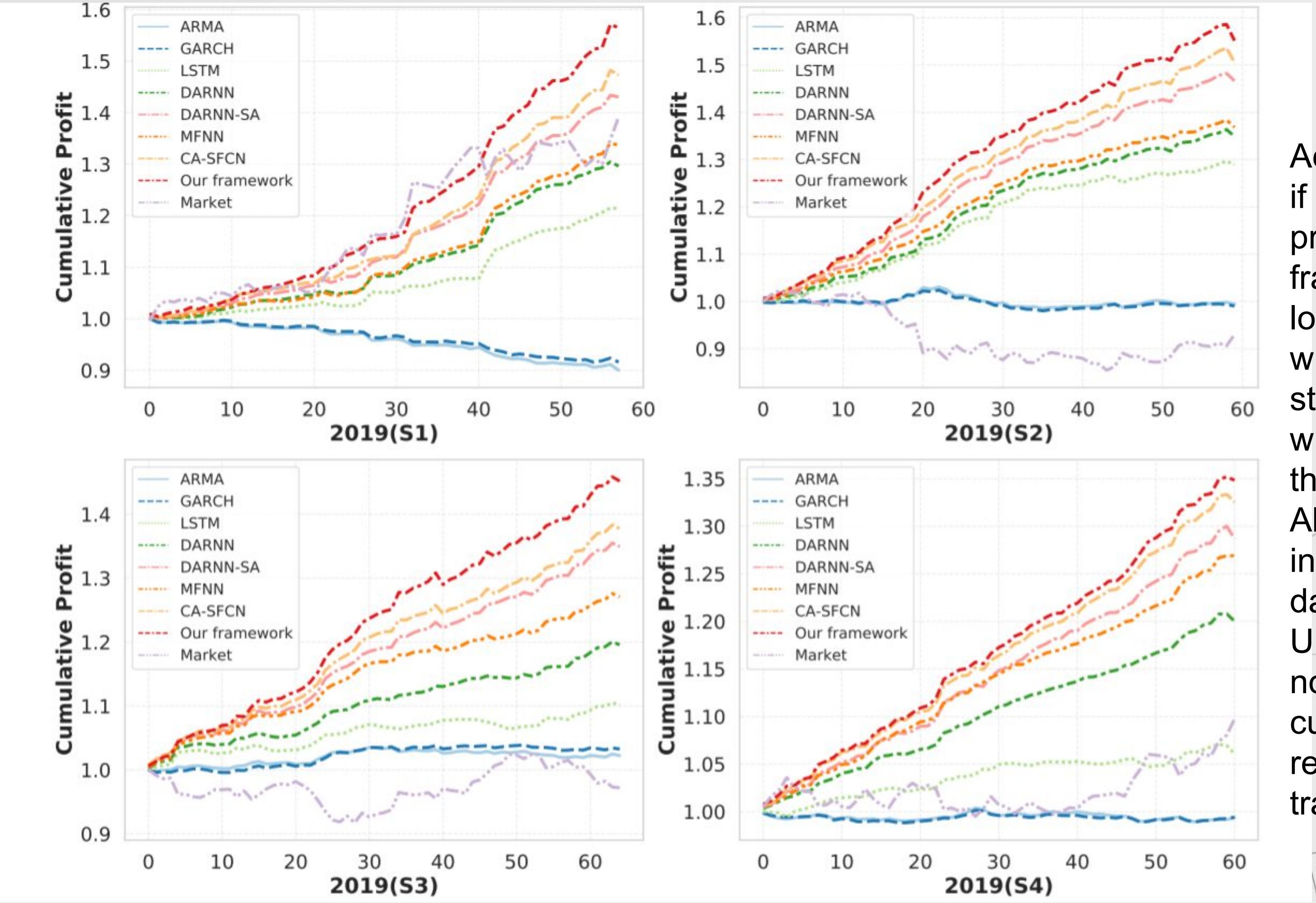


(c) Attention weights without CI



(d) Attention weights with CI

Trading Simulation



According to the prediction: if a rising trend of a stock price is given by our framework, we will take a long position on that stock; while a falling trend of a stock price is predicted, we will take a short position for that stock.

All stocks are evenly invested in and held for one day.

Under the circumstance of no transaction cost, the cumulative profit are reinvested on the next trading da

Conclusion

- Deep learning models collaborating with structural information achieve competent performance
- Practical capabilities in stock prediction and trading
- Employing models with attention mechanisms, the long-term dependencies can be captured via structural information.
- Employing graph node weights as additional knowledge for temporal attention can tackle the chaotic property of financial time series and achieve better stock prediction performance

THANKS

Does anyone have any questions?