DTML-Self-Attention (Proposed)

Tae-Won Lee,
leetee95@gmail.com
2022. 02. 08.

TO DO list

- 지난주

- 논문 구현

- 이번주

- 제안된 모델 구현

Related works

- Prediction paper survey

학회 명	SCI	IF	논문 명	발간 년도	인용 수	내용
International Joint Conference on Artificial Intelligence (IJCAI-19)	conf		CLVSA: A Convolutional LSTM Based Variational Sequence-to-Sequence Modelwith Attention for Predicting Trends of Financial Markets	2021	23	Convolusion 계층과, self attention lstm을 사용하여 모델만의 encoder decoder를 구축하여 예측 진행. 데이터는 open close high low volume 을 이용함.
ACM International Conference on Information and Knowledge Management (CIKM '18),	conf		Incorporating Corporation Relationship via Graph Convolutional Neural Networks for Stock Price Prediction	2018	45	특정기업의 주가를 lstm을 이용하여 node2vector, Deep work, LINE 등을 사용하여 임베딩 한 뒤, 임베딩된 각각의 기업들의 정보의 연관성을 활용하여 주가의 방향을 예측하기 위해 GCN 사용
Expert Systems With Applications	SCIE	6.954	CNNpred: CNN-base d stock market prediction using a diverse set of variables	2018	13	2D, 3D CNN을 활용하여 주식 Market data로부터 예측 진행
International Joint Conference on Artificial Intelligence (IJCAI-20)	conf		Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction	2020	16	Transformer model의 locality를 강화시키기 위해 Multi-Scale Gaussian Prior 를 사용했고, Orthogonal Regularization를 사용하여 redundant heads를 학습하는 것을 피함.
International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI	conf		Stock Market Prediction Based on Generative Adversarial Network	2018	94	Generative Adversarial Network을 이용한 주식 시장 예측

Related works

- Prediction paper survey

학회 명	SCI	IF	논문 명	발간 년도	인용 수	내용
Neural Computing and Applications	SCIE	5.606	A CNN-BiLSTM-AM method for stock price prediction	2020	12	CNN을 활용한 feature extraction과 BiLSTM, attention mechanism을 활용한 주가예측, 지수 데이터 이용 (regression)
Neural Computing and Applications	SCIE	5.606	A CNN-LSTM model for gold price time-series forecasting	2020	102	CNN을 활용한 feature extraction과 LSTM을 활용한 주가 예측 (Classification)

Related works

- Technical indicator

학회 명	SCI	IF	논문 명	발간 년도	인용 수	내용
Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining	conf		Individualized Indicator for All Stock-wise Technical Indicator Optimization with Stock Embedding	2019	21	EMAm, MACD, Klength, KUpper Length, KLower Length, Biasm, ROCm, MeanAmplitudem
Quantitative Finance	SCIE	2.222	Exploring the attention mechanism in LSTM based Hong Kong stock price movement prediction	2019	31	ROCP, OROCP, HROCP, LROCP, MA, MACP, MACD, DIFROCP, DEAROCP, MACDROCP, RSI, RSIROCP, VROCP, BOLL, VMAROCP, VMACP, PRICE, VOLUME
Expert Systems With Applications	SCIE	6.954	Integrating metaheuristics and Artificial Neural Networks for improved stock price prediction	2016	220	여러 이동평균들과, technical indicator, historical data를 이용
International Conference on Advanced Informatics: Concept Theory and Applications	conf		Deep Learning for Stock Market Prediction Using Event Embedding and Technical Indicators	2018	33	Stochastic %K, Stochastic %D, Momentum, Rate of Change, William's %R, A/D Oscillator, Disparity 5
International Journal on Emerging Technologies	X	X	Stock Indices Price Prediction Based on Technical Indicators using Deep learning model	2019	7	RSI, MA, Stochastic Oscillator (%K), William (%R), Exponential Moving Average (EMA), Moving Average Convergence Divergence (MACD):

Related works

- Feature engineering in DTML and AdvLSTM

Features	Calculation
z_{open}	$z_{\text{open}} = \text{open}_t / \text{close}_t - 1$
z_{high}	$z_{\text{high}} = \text{high}_t / \text{close}_t - 1$
z_{low}	$z_{\text{low}} = \text{low}_t / \text{close}_t - 1$
z_{close}	$z_{\text{close}} = \text{close}_t / \text{close}_{t-1} - 1$
$z_{\text{adj_close}}$	$z_{\text{adj_close}} = \text{adj_close}_t / \text{adj_close}_{t-1} - 1$
z_{d5}, z_{d10} z_{d15}, z_{d20} z_{d25}, z_{d30}	e.g., $z_{dk} = \frac{\sum_{i=0}^{k} \text{adj_close}_{t-i}}{k \cdot \text{adj_close}_t} - 1$

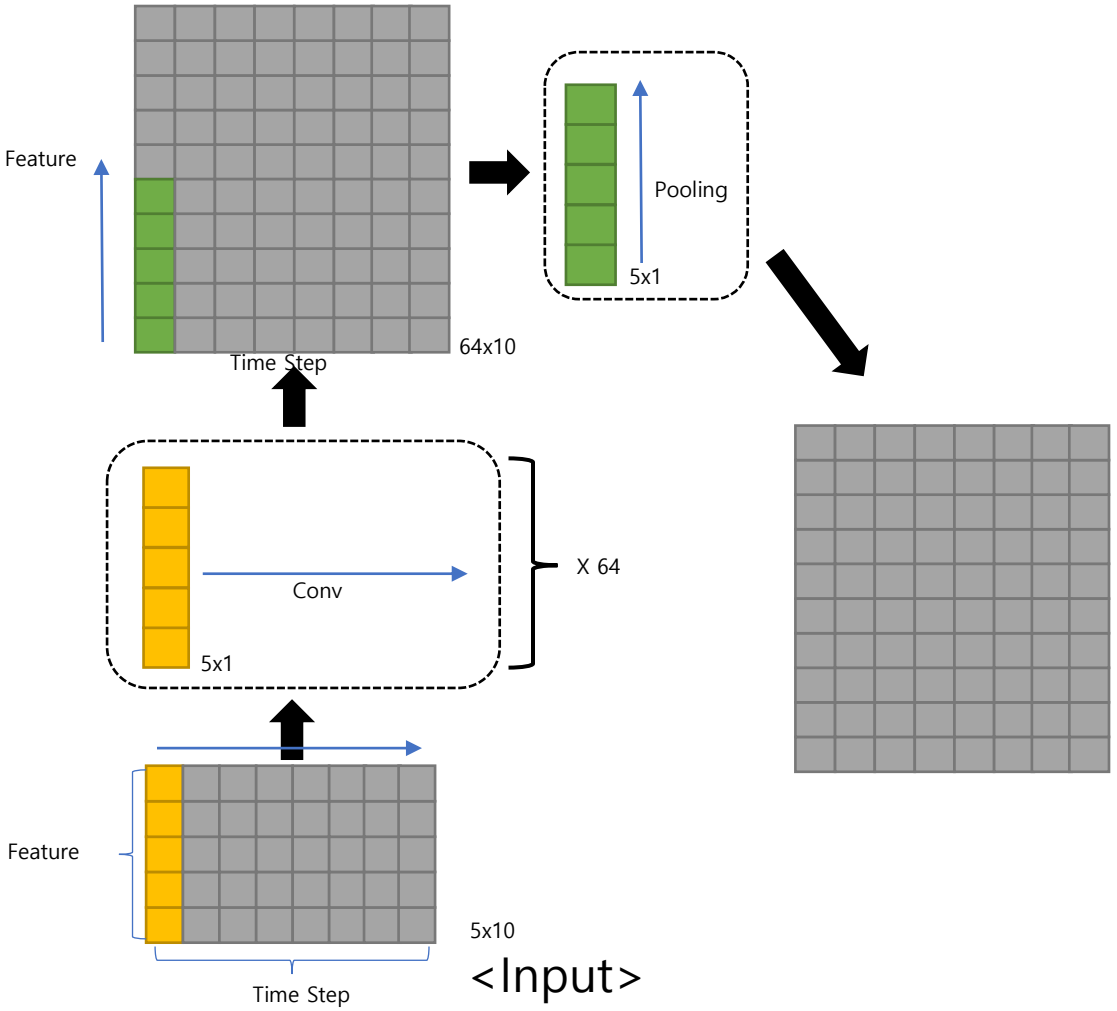
- 개선방안

  - open, close, high, low, volume 등을 이용해 만들어낼 수 있는 technical indicator를 추가로 입력 해주는 것.

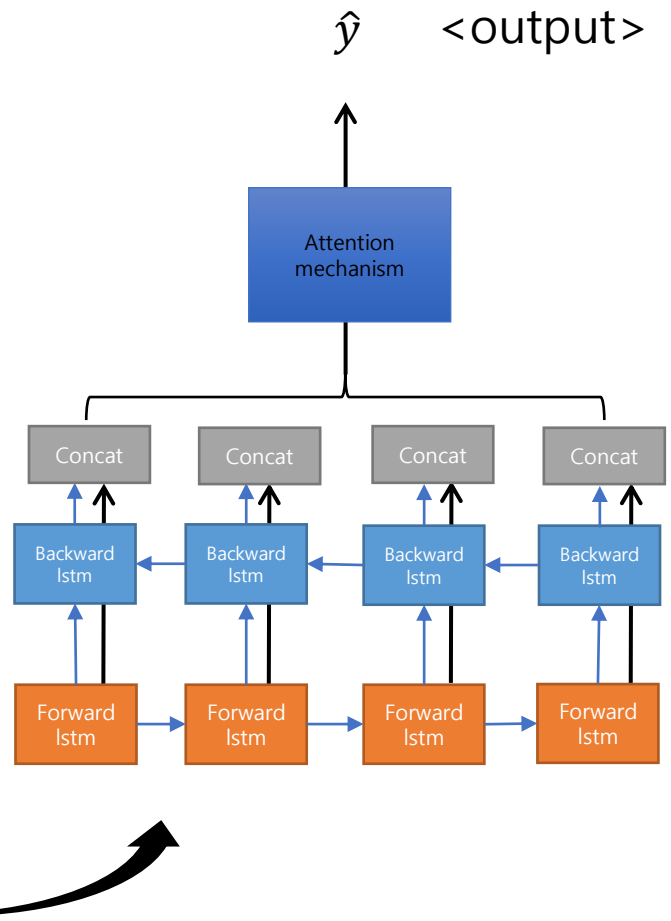
- Issue
  - DTML 구조 중 하나인 attention lstm에서 attention mechanism의 구조를 논문에 나와있는 대로가 아닌 self-attention을 적용한 실험에서 높은 성능을 보임.
  - DTML 논문의 핵심 contribution 중 하나가 transformer 구조가 correlation을 이용하는 것인데, 현재 구현에서 제외됨.

Experiment Setting

- CNN



- Bidirectional attention lstm



Experiment Setting

- CNN-BiLSTM-AM [8]
  - 구현 시 수정사항
    - 본 논문에서는 Regression 문제를 다루고 있지만 비교 실험을 위해 Classification으로 변형
    - 본 논문에서 Input의 형태를 일반적인 normalization을 이용하여 변형했지만, 동일한 input에 대한 model의 성능 비교를 위해 사전에 구현한 DTML, AdvLSTM과 동일한 Feature 사용
- Blocked Cross validation [9]
  - 다수의 실험을 위해 오른쪽과 같은 방식으로 10개의 분할로 실험.
  - KDD 17 datasets에서 10개의 분할로 인해 test data의 개수가 131개로 한정됨.

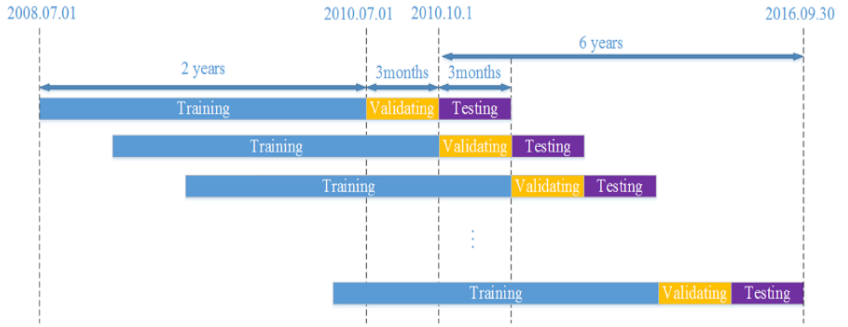


Fig 7. Continuous dataset arrangement for training, validating and testing during the whole sample period.

<PloS one>

Experiment Setting

- Cross validation

학회 명	SCI	IF	논문 명	발간 년도	인용 수	내용
PloS one	SCIE	3.24	A deep learning framework for financial time series using stacked autoencoders and long-short term memory	2017	625	분할의 개수 : 6 분할 비율 : 8:1:1
Applied Sciences-Basel	SCIE	2.679	A Novel Approach to Short-Term Stock Price Movement Prediction using Transfer Learning	2019	22	분할의 개수 : 3 분할 비율 : 6:1:1

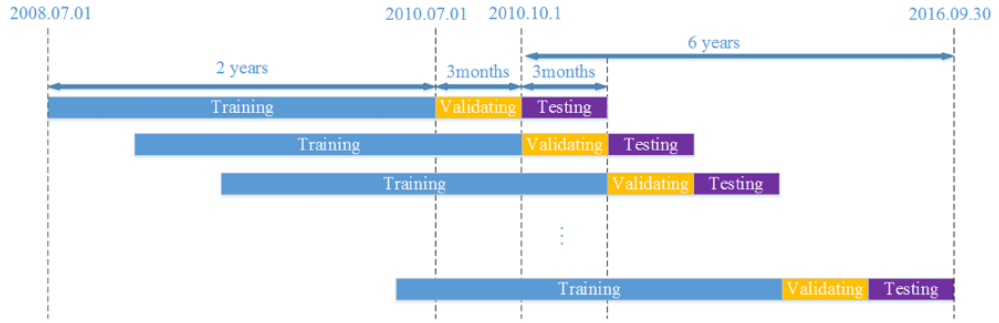


Fig 7. Continuous dataset arrangement for training, validating and testing during the whole sample period.

<PloS one>

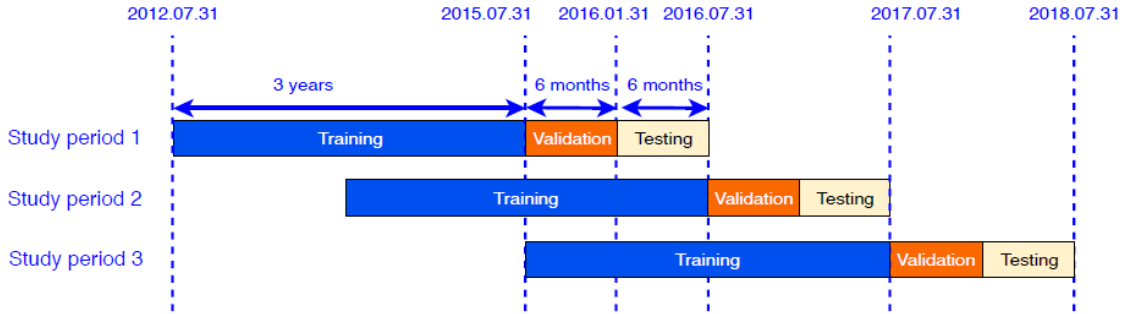


Figure 2. Study periods.

<Applied Sciences-Basel>

- DTML 논문 -> Random seed를 설정하여 10번 반복

Experiment Setting

- KDD 17 (2007-01-01~2016-12-31)

  - **주식 데이터 코드 목록** : 'AAPL', 'AMZN', 'BA', 'BAC', 'BHP', 'BRK-B', 'CHL', 'CMCSA', 'CVX', 'D', 'DCM', 'DIS', 'DOW', 'DUK', 'EXC', 'GE', 'GOOGL', 'HD', 'INTC', 'JNJ', 'JPM', 'KO', 'MA', 'MMM', 'MO', 'MRK', 'MSFT', 'NGG', 'NTT', 'NVS', 'ORCL', 'PEP', 'PFE', 'PG', 'PTR', 'RDS-B', 'RIO', 'SO', 'SPY', 'SYT', 'T', 'TM', 'TOT', 'UNH', 'UPS', 'VALE', 'VZ', 'WFC', 'WMT', 'XOM', **총 50개.**

- ACL 18 (2014-01-01~2015-12-31)

  - **주식 데이터 코드 목록** : 'AAPL', 'ABB', 'ABBV', 'AEP', 'AGFS', 'AMGN', 'AMZN', 'BA', 'BABA', 'BAC', 'BBL', 'BCH', 'BHP', 'BP', 'BRK-A', 'BSAC', 'BUD', 'C', 'CAT', 'CELG', 'CHL', 'CHTR', 'CMCSA', 'CODI', 'CSCO', 'CVX', 'D', 'DHR', 'DIS', 'DUK', 'EXC', 'FB', 'GD', 'GE', 'GOOG', 'HD', 'HON', 'HRG', 'HSBC', 'IEP', 'INTC', 'JNJ', 'JPM', 'KO', 'LMT', 'MA', 'MCD', 'MDT', 'MMM', 'MO', 'MRK', 'MSFT', 'NEE', 'NGG', 'NVS', 'ORCL', 'PCG', 'PCLN', 'PEP', 'PFE', 'PG', 'PICO', 'PM', 'PPL', 'PTR', 'RDS-B', 'REX', 'SLB', 'SNP', 'SNY', 'SO', 'SPLP', 'SRE', 'T', 'TM', 'TOT', 'TSM', 'UL', 'UN', 'UNH', 'UPS', 'UTX', 'V', 'VZ', 'WFC', 'WMT', 'XOM', **총 87개.**

-> 데이터 저장 완료 + 추후에 위의 코드로 데이터 불러오기 가능

Experiment Setting

- KDD 17 (2007-01-01~2016-12-31)
  Train data = 2007-01-03 ~ 2015-01-01,
  validataion data = 2015-01-02 ~ 2016-01-03
  Test data = 2016-01-04 ~ 2016-12-31

- ACL 18 (2014-01-01~2015-12-31)
  Train data = 2014-01-02 ~ 2015-08-02,
  validataion data = 2015-08-03 ~ 2015-09-30
  Test data = 2015-10-01 ~ 2015-12-31

Experiment Setting

- CNN-BiLSTM-AM
  - **Turnover amount** refers to the total amount of shares of all stocks traded that day

- Turnover는 총 주식의 거래대금을 나타내는데, 본 논문에서는 시장 index의 예측을 목적으로 하고 있기 때문에, 전체 시장의 거래대금을 나타내고 있음.

- Turnover의 개념이 시장의 유동성을 나타내는 지표로써, 개별 주식을 예측할 때 개별주식의 거래대금을 추가 input으로 넣어야 할지, 시장의 거래대금을 input 으로 넣어야 할지 모호함이 존재

- opening price, highest price, lowest price, closing price, volume, turnover, ups and downs, and change
  총 8개의 변수중 turnover를 제외한 7개의 변수로 구현

Comparison results

- Accuracy (Index : NASDAQ)

Data	Proposed-dtml-selfatt	Standard Deviation	DTML [2]	Standard Deviation	AdvLSTM [7]	Standard Deviation	CNN BiLSTM AM [8]	Standard Deviation
AAPL	0.5297	0.0587	0.5203	0.0766	0.4766	0.0403	0.4813	0.0684
AMZN	0.4922	0.0403	0.5125	0.0563	0.5313	0.0349	0.5188	0.0588
BA	0.4813	0.0473	0.5063	0.0459	0.5156	0.0407	0.5031	0.0473
BAC	0.5172	0.0366	0.5156	0.0376	0.5078	0.0710	0.4750	0.0856
BHP	0.5328	0.0548	0.4781	0.0637	0.5000	0.0469	0.4516	0.0491
BRK-B	0.5016	0.0373	0.4719	0.0608	0.5000	0.0576	0.5031	0.0394
CHL	0.5266	0.0442	0.5234	0.0573	0.5078	0.0744	0.4750	0.0585
CMCSA	0.5000	0.0532	0.4906	0.0378	0.4875	0.0493	0.5219	0.0337
CVX	0.5188	0.0745	0.4813	0.0666	0.4969	0.0531	0.4891	0.0688
D	0.5047	0.0670	0.5109	0.0625	0.5031	0.0513	0.4984	0.0665
DCM	0.4844	0.0568	0.5266	0.0827	0.4844	0.0513	0.4953	0.0442
DIS	0.5172	0.0392	0.4875	0.0695	0.5109	0.0442	0.5016	0.0570
DOW	0.4844	0.0546	0.5000	0.0431	0.4828	0.0591	0.4875	0.0691
DUK	0.5406	0.0272	0.4828	0.0642	0.4734	0.0637	0.5344	0.0647
EXC	0.5156	0.0563	0.4953	0.0750	0.4688	0.0629	0.5203	0.0641
GE	0.5172	0.0574	0.5156	0.0504	0.5219	0.0706	0.4875	0.0462
GOOGL	0.5156	0.0363	0.5297	0.0461	0.5094	0.0713	0.5188	0.0596
HD	0.4703	0.0466	0.5047	0.1105	0.5281	0.0604	0.4875	0.0781
INTC	0.5031	0.0640	0.5109	0.0370	0.4859	0.0866	0.5188	0.0531
JNJ	0.5063	0.0824	0.5250	0.0622	0.4828	0.0520	0.5109	0.0555

Comparison results

- Accuracy (Index : NASDAQ)

Data	Proposed-dtml-selfatt	Standard Deviation	DTML [2]	Standard Deviation	AdvLSTM [7]	Standard Deviation	CNN BiLSTM AM [8]	Standard Deviation
JPM	0.5109	0.0484	0.5016	0.0439	0.5031	0.0303	0.5328	0.0353
KO	0.5063	0.0776	0.4734	0.0437	0.4797	0.0644	0.5109	0.0408
MA	0.5094	0.0459	0.5125	0.0666	0.4922	0.0757	0.5141	0.0693
MMM	0.4875	0.0889	0.4797	0.0681	0.5125	0.0616	0.4719	0.0517
MO	0.5125	0.0726	0.4891	0.0759	0.5125	0.0348	0.5281	0.0503
MRK	0.5016	0.0386	*0.4938	0.0649	*0.4938	0.0490	0.4859	0.0386
MSFT	0.5031	0.0462	0.5344	0.0636	0.4953	0.0564	0.4984	0.0718
NGG	0.5188	0.0368	0.5375	0.0390	0.5156	0.0576	0.5250	0.0649
NTT	0.5063	0.0564	0.5672	0.0730	0.5156	0.0593	0.5188	0.0536
NVS	0.5141	0.0481	0.4938	0.0514	0.5078	0.0586	0.4844	0.0537
ORCL	0.5000	0.0585	0.5094	0.0390	0.5047	0.0762	0.4938	0.0538
PEP	0.4719	0.0604	0.4875	0.0488	0.5031	0.0684	0.5000	0.0546
PFE	0.5063	0.0538	0.4859	0.0379	0.4781	0.0649	0.5188	0.0531
PG	0.5203	0.0609	0.4844	0.0474	0.5250	0.0656	0.5344	0.0418
PTR	0.4922	0.0598	0.5219	0.0480	0.4844	0.0242	0.5109	0.0637
RDS-B	0.4969	0.0628	0.4875	0.0446	0.4969	0.0643	0.5016	0.0761
RIO	0.5016	0.0611	0.5234	0.0564	0.5172	0.0603	0.4969	0.0355
SO	0.5109	0.0609	0.5000	0.0305	0.5172	0.0525	0.5156	0.0349
SPY	0.4953	0.0489	0.4781	0.0364	0.5031	0.0858	0.5266	0.0699
SYT	0.5000	0.0791	0.5125	0.0368	0.5047	0.0688	0.5313	0.0494

Comparison results

- Accuracy (Index : NASDAQ)

Data	Proposed-dtml-selfatt	Standard Deviation	DTML [2]	Standard Deviation	AdvLSTM [7]	Standard Deviation	CNN BiLSTM AM [8]	Standard Deviation
T	0.4953	0.0585	0.5375	0.0797	0.5094	0.0560	0.4703	0.0672
TM	0.5250	0.0893	0.4813	0.0575	0.5000	0.0772	0.4906	0.0671
TOT	0.5156	0.0652	0.4844	0.0576	0.5203	0.0637	0.5297	0.0386
UNH	0.4891	0.0709	0.4641	0.0850	0.5156	0.0464	0.4875	0.0702
UPS	0.4969	0.0483	0.5109	0.0617	0.4984	0.0525	0.4875	0.0260
VALE	0.4969	0.0698	0.4969	0.0531	0.4922	0.0887	0.4953	0.0644
VZ	0.5172	0.0603	0.4641	0.0469	0.5141	0.0603	0.5234	0.0144
WFC	0.4938	0.0560	0.5219	0.0585	0.4766	0.0776	0.5281	0.0658
WMT	0.5109	0.0420	0.4766	0.0397	0.5156	0.0447	0.4969	0.0666
XOM	0.4906	0.0637	0.4484	0.0453	0.4891	0.0572	0.5375	0.0689

계	Proposed	Standard Deviation	DTML [2]	Standard Deviation	AdvLSTM [7]	Standard Deviation	CNN BiLSTM AM [8]	Standard Deviation
NASDAQ	0.505125	0.05649	0.5010	0.056139	0.5014	0.058897	0.5045	0.0556

Comparison results

- 각 데이터 별 정확성이 높게 나온 Model

model	높은 정확성이 나온 데이터 개수			
Proposed vs DTML vs AdvLSTM vs CNN-BiLSTM-AM	11	14	8	17

- 순위 평균

model	높은 정확성이 나온 데이터 개수
DTML-self-att (proposed)	2.36
DTML	2.56
AdvLSTM	2.66
CNN-BiLSTM-AM	2.36

Comparison results

- 현재 제안된 모델(DTML-Self-att)이 accuracy에서 근소하게 CNN-BiLSTM-AM 모델을 앞섬
- 현재 experiment setting 내에서 전체 데이터를 10분할 하므로 데이터수가 작기때문에 모델의 Parameter 수가 많은 DTML을 기반으로 한 model이 overfitting 되는 경향이 있음.
- 기존의 Accuracy에서 높은 수치를 보이는 CNN-BiLSTM-AM에 새로운 input과 self attention 적용
- 많은 technical indicator의 종류 때문에 선정하는 기준의 모호함

Comparison results

- Execution time (Index : NASDAQ)

Data	Proposed	DTML [2]	AdvLSTM [7]	CNN BiLSTM AM [8]
AAPL	277.2576	263.9149	335.3957	161.3731
AMZN	271.8871	253.9057	328.4342	155.5468
BA	272.1072	259.6827	330.7123	154.4373
BAC	271.8674	261.9649	328.6117	154.0428
BHP	272.2275	265.3077	330.0409	154.0612
BRK-B	273.5721	266.6888	332.5733	156.685
CHL	272.5819	258.5403	332.0283	156.9022
CMCSA	273.1618	260.0197	331.7802	154.8558
CVX	272.4924	256.8395	333.7976	159.7776
D	272.7023	258.8118	336.75	161.0854
DCM	272.4823	257.2388	337.9522	151.6179
DIS	272.3872	259.0166	334.1846	153.217
DOW	272.5773	268.6049	332.3935	150.917
DUK	273.4325	264.1363	336.7299	151.8336
EXC	272.4871	258.7369	332.5818	154.4056
GE	272.8473	255.9551	332.274	151.5199
GOOGL	274.0214	257.6987	332.088	151.5865
HD	273.2945	258.7146	332.761	151.2423
INTC	272.7075	259.5852	330.7832	151.3641
JNJ	272.7015	256.9317	335.7759	151.2671

Comparison results

- Execution time (Index : NASDAQ)

Data	Proposed	DTML [2]	AdvLSTM [7]	CNN BiLSTM AM [8]
JPM	272.4973	253.8562	330.5911	151.1496
KO	273.2475	256.2077	333.9058	152.251
MA	273.4073	253.8022	330.5462	152.4183
MMM	273.7122	256.4322	333.4852	151.9352
MO	274.4323	255.4431	334.212	151.5931
MRK	274.5024	254.1493	331.877	151.7722
MSFT	273.8872	253.6562	330.5377	151.2965
NGG	274.1622	254.4988	333.7752	151.6819
NTT	274.1336	269.4161	331.5776	151.7267
NVS	274.8093	282.8205	333.1482	151.7732
ORCL	274.5472	285.2583	330.9961	151.7742
PEP	273.9875	286.9107	334.1579	152.1638
PFE	274.5314	276.5648	332.2469	151.719
PG	275.0316	276.1022	334.5486	151.5581
PTR	274.5576	273.6902	328.7153	151.6311
RDS-B	276.6224	276.4224	332.1698	151.7431
RIO	275.9973	277.6166	327.683	152.4093
SO	274.4875	274.19	334.8768	151.7444
SPY	274.7325	282.8317	335.1571	151.4708
SYT	274.7375	283.6031	331.131	151.5391

Comparison results

- Execution time (Index : NASDAQ)

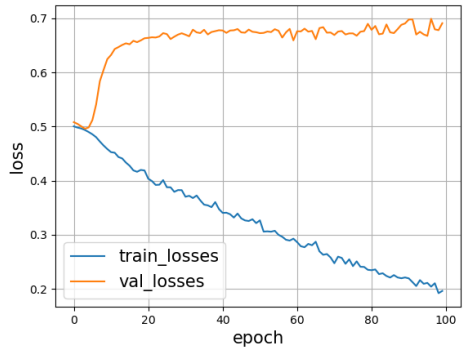
Data	Proposed	DTML [2]	AdvLSTM [7]	CNN BiLSTM -AM [8]
T	274.7372	278.2768	332.5667	151.8402
TM	275.3128	258.4921	329.7004	151.7582
TOT	275.2173	257.8805	330.9925	152.1022
UNH	275.182	255.8676	331.0006	151.5599
UPS	275.342	257.7907	332.7059	151.8602
VALE	275.6073	262.0775	328.3782	151.7032
VZ	274.0373	258.6767	332.8664	152.5361
WFC	276.6782	256.6366	329.7513	152.2303
WMT	275.0723	261.3478	333.4359	151.9162
XOM	275.4223	258.9515	332.3697	151.7532

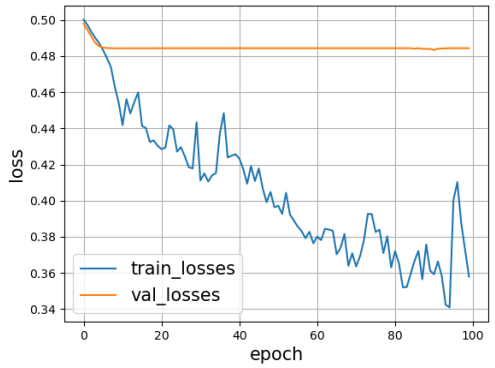
계	Proposed	DTML [2]	AdvLSTM [7]	CNN BiLSTM AM [8]
NASDAQ	273.9886	263.8353	332.3351	152.8470

CNN을 활용한 모델의 실행시간이 가장 적음

Comparison results

- Learning curve(loss)



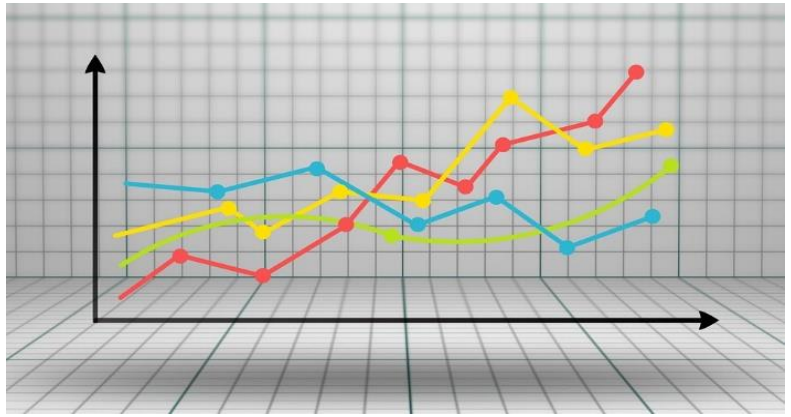


<CNN-BiLSTM-AM>

- 본 논문에서는 MAE Loss(L1 Loss)를 사용.
- 특정 분할 데이터 부분에서 반복적인 실험 epoch 이 진행될수록 예측값이 1로 수렴하여 좌측과 같은 결과가 발생.

Introduction

- Univariate time series는 **하나의 time-dependent variable**를 가지고 있는 데 비해 Multivariate time series 은 최소한 **두개 이상의 time-dependent variable** 을 가지고 있음.



〈 Multivariate time series 의 예시〉

- Time series prediction은 **전력량 예측, 주가예측, 수요 예측** 등의 문제에서 이용되는 중요한 문제임.

- Univariate time series data를 이용한 예측은 예측 시점 **이전의 정보(Seasonality, Tendency, cycle)** 를 이용해 다음 시점의 예측을 진행하지만, Multivariate time series는 연관성이 있다고 판단되는 time series data간의 **Correlation**이용해 다음 값들을 예측함 .

Introduction

- 대부분의 주식 종목들은 각각의 sector에 속함.

- Previous paper

  - sector의 **고정된 list**를 가지고 있고, sector의 **pre-trained list에 의존**함.
  - 시간에 따라 자연적으로 변하는 주식 상관의 **동적인 속성을 잃음**.
  - sector에 없는 주식이나, sector가 모호한 주식에 대해 예측 모형이 적용되지 않음.
  - 예측의 품질이 **sector 정보의 품질에 크게 의존**됨.

- Univariate time series data를 이용한 예측은 예측 시점 **이전의 정보(Seasonality, Tendency, cycle)**를 이용해 다음 시점의 예측을 진행하지만, Multivariate time series는 연관성이 있다고 판단되는 time series data간의 **Correlation**이용해 다음 값들을 예측함 .

Related works

- **Correlated Stock prediction**

Statistical method

Vector Autoregression (VAR) [1]

$$y_t = c + A_1 y_{t-1} + A_2 y_{t-2} + \cdots + A_p y_{t-p} + e_t$$

- VAR은 확률적 프로세스 모델의 한 유형으로 Multivariate time series 를 이용하여 단일 변수 Autoregression 모델을 일반화 함.

- 통계적인 모델의 단점은 선형적인 정보만을 선택적으로 학습한다는 단점이 있음.

Others

[5]

- 주가의 패턴을 포착하고, Markov random field를 사용하여 연관성 정보로 이용함.

[6]

- 주식 사이의 상관을 Graph 형태로 따로 학습함.

- 위 방법의 주된 제한점은 pre-defined correlation에 의존한다는 점임.

Neural network method

DA-RNN [2]

- Multivariate time series의 Correlation을 Attention 구조를 활용하여 학습하는 초기의 모델
- 금융 시계열에서 Close 값 만을 이용하는 단점이 있음.

Spectral temporal GNN [4]

- Graph neural network로 Multivariate time series의 Correlation을 학습.
- 금융 도메인에 적용이 되지 않음

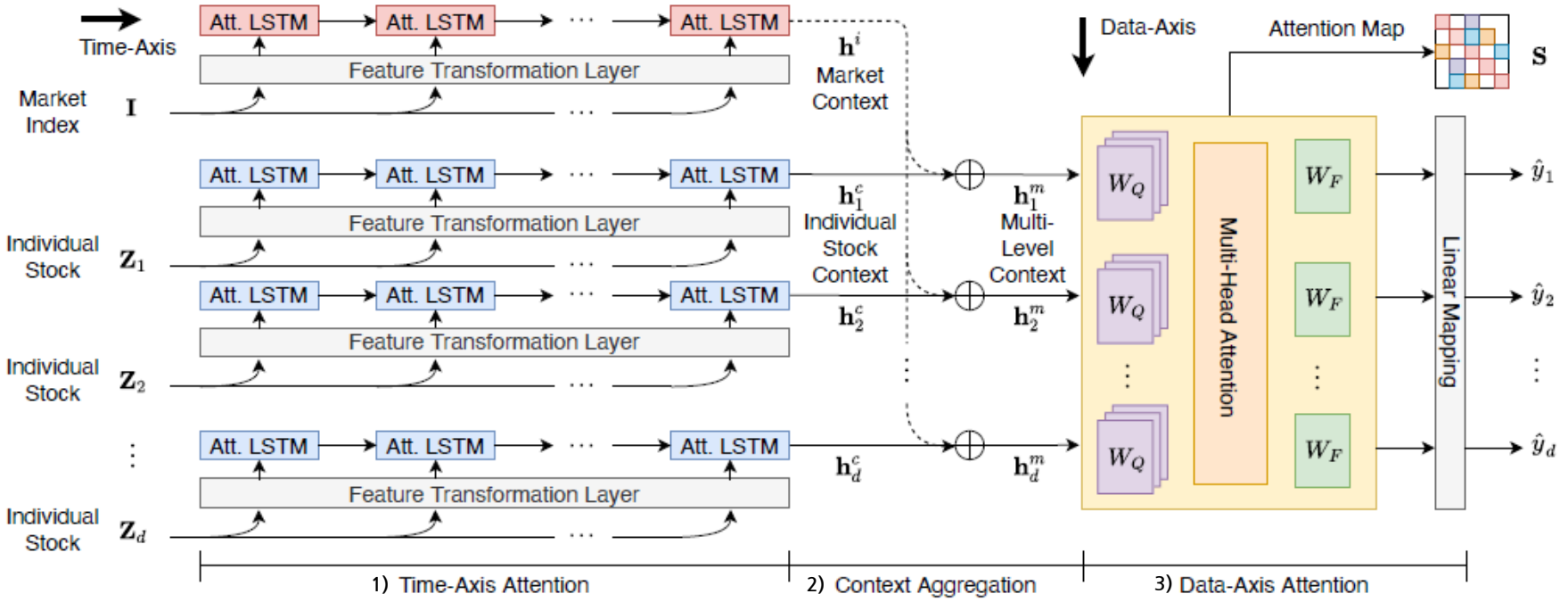
Transforemr encoder [7] [8] [9]

- 전력 소비량 , 감염병 전파, 주가 예측 등의 업무를 Transforemr encoder를 활용하여 상관성을 학습한 예측 진행.

- 신경망 기반 방식은 비선형적 정보를 함께 학습할 수 있음.
- Transformer 방식은 기존 RNN의 Gradient vanishing 문제를 피할 수 있음.

Experiment Setting

- **Model Structure**



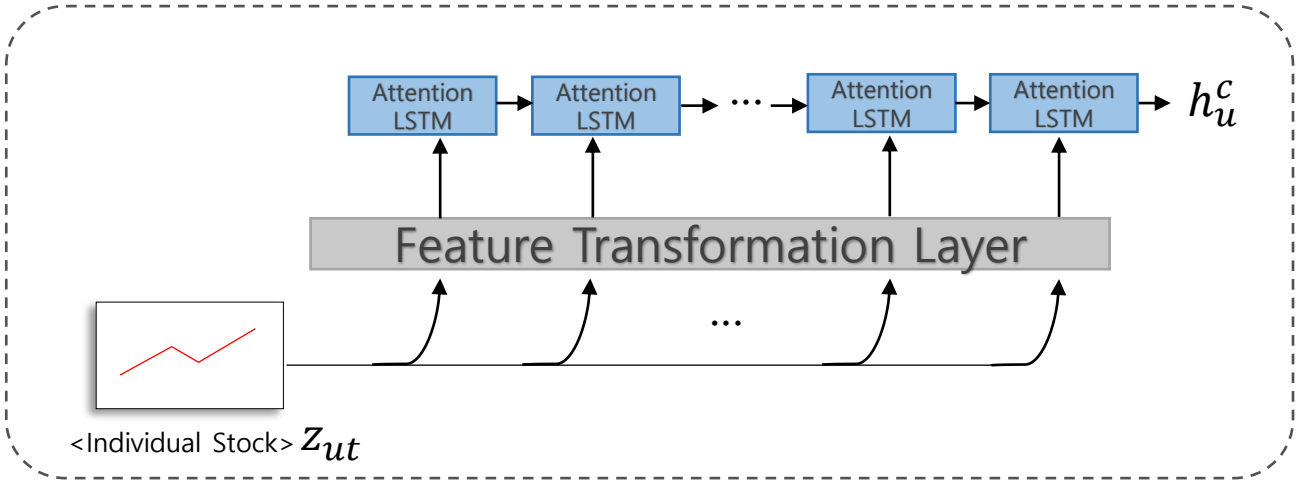
- 전체 모델 구조는 크게 3가지로 나뉨
  1) Time-Axis Attention
  2) Context Aggregation
  3) Data-Axis Attention

Experiment Setting

- **Model Structure - 1) Attentive Context Generation**

- $\{z_{ut}\}_{t \le T}$ of length l, where u and t are stock and time indices respectively

- 위의 z_{ut} 에서 포괄적인 context vector인 h_u^c 를 얻는것이 Time-Axis Attention단계의 목표



<Time-Axis Attention>

- Feature Transformation

$$\tilde{\mathbf{z}}_{ut} = \tanh(\mathbf{W}_s \mathbf{z}_{ut} + \mathbf{b}_s), where \ \ \mathbf{W}_s \in \mathbb{R}^{h \times l} \ and \ \ \mathbf{b}_s \in \mathbb{R}^h$$

- 위 계산은 Attention LSTM 구조에 학습시키기 전 계산복잡도의 증가 없이 복수의 Time step을 aggregation하기 위한 과정

Experiment Setting

- **Model Structure - 1) Attentive Context Generation**

  - Attention LSTM

$$\alpha_i = \frac{\exp(\mathbf{h}_i^\top \mathbf{h}_T)}{\sum_{j=1}^{T} \exp(\mathbf{h}_j^\top \mathbf{h}_T)}$$

<Attention score>

$$\tilde{\mathbf{h}}^{\mathrm{c}} = \sum_i \alpha_i \mathbf{h}_i$$

<Context vector>

  - Context vector는 attentionLSTM을 통과하여 최종적으로 계산됨
  - AttentionLSTM은 Vanila LSTM과 비교해 효과적으로 정보를 전달함.

  - Context Normalization

$$h_{ui}^{\mathrm{c}} = \gamma_{ui} \frac{\tilde{h}_{ui}^{\mathrm{c}} - \mathrm{mean}(\tilde{h}_{ui}^{\mathrm{c}})}{\mathrm{std}(\tilde{h}_{ui}^{\mathrm{c}})} + \beta_{ui}$$

  - 각각의 주식의 정보를 요약한 Context vector는 서로 다른 값의 범위를 가지고 있기 때문에 Aggregation 단계 전에 Normalization을 진행.
  - 감마와 베타는 학습가능한 파라미터이다.

Experiment Setting

- **Model Structure - 2) Multi-Level Context Aggregation**

  - Multi-Level Contexts

$$\mathbf{h}_u^m = \mathbf{h}_u^c + \beta \mathbf{h}^i$$

  - h_u^m은 개별 주식의 context vector이고, h^i는 Market Index의 context vector임.
  - β는 Index context vector가 어느정도 영향을 미칠지 결정할 수 있는 hyper parameter이다.

  - The Effect of Global Contexts

$$\mathbf{h}_u^{m\top} \mathbf{h}_v^m = \mathbf{h}_u^{c\top} \mathbf{h}_v^c + \beta \mathbf{h}^{i\top} (\mathbf{h}_u^c + \mathbf{h}_v^c) + \beta^2 \mathbf{h}^{i\top} \mathbf{h}^i$$

  - Global Contexts의 효과를 알아보기 위해 가장 단순한 Attention인 dot product를 시행.
  - 오른쪽 항에서 두번째 term을 보면, 주식들과 시장 움직임의 상관성에 더 큰 가중치를 부여함.
  - 오른쪽 항에서 세번째 term을 보면, 시장 움직임의 강도를 나타낼 수 있음.

Experiment Setting

- **Model Structure - 3) Data-Axis Self-Attention**

- Self-Attention

$$\mathbf{H} \in \mathbb{R}^{d \times h}, \qquad W \in R^{hXh}$$

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_q \qquad \mathbf{K} = \mathbf{H}\mathbf{W}_k \qquad \mathbf{V} = \mathbf{H}\mathbf{W}_v$$

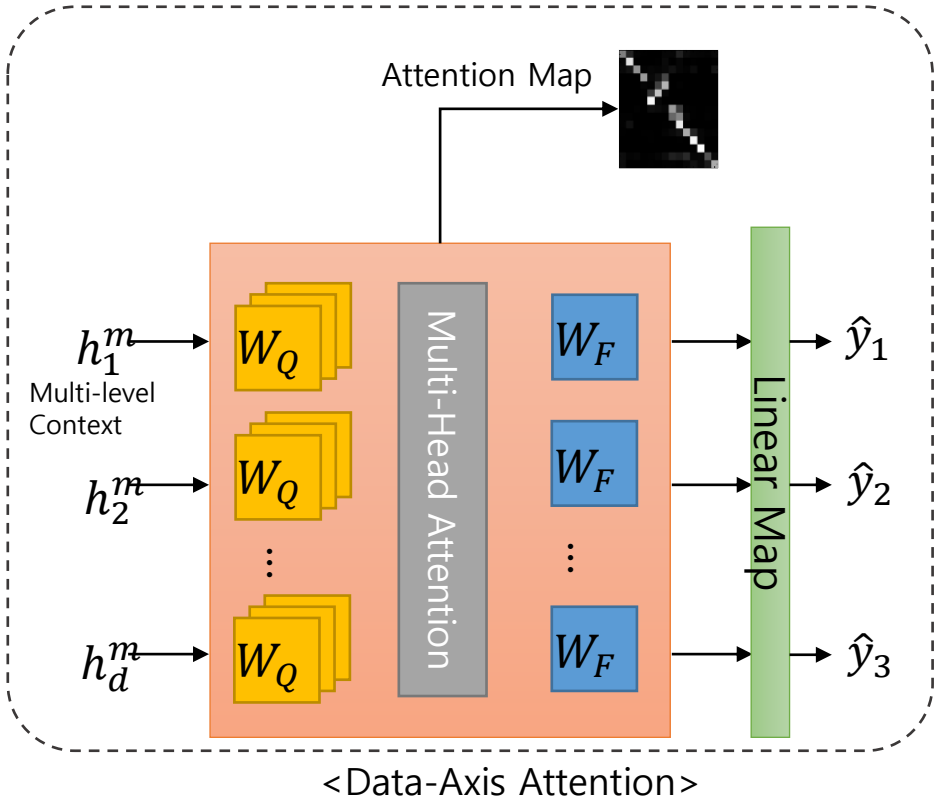
$$\tilde{\mathbf{H}} = \mathbf{S}\mathbf{V} \quad \text{where} \quad \mathbf{S} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{h}}\right)$$

- d is the number of stocks and h is the length of context vectors
- W is learnable weight

- Nonlinear Transformation

$$\mathbf{H}_p = \tanh(\mathbf{H} + \tilde{\mathbf{H}} + \text{MLP}(\mathbf{H} + \tilde{\mathbf{H}}))$$

- The hidden layer size of MLP is 4h



<Data-Axis Attention>

Experiment Setting

- **Model Structure - 3) Data-Axis Self-Attention**

- Transformer encoder를 사용하는 장점.

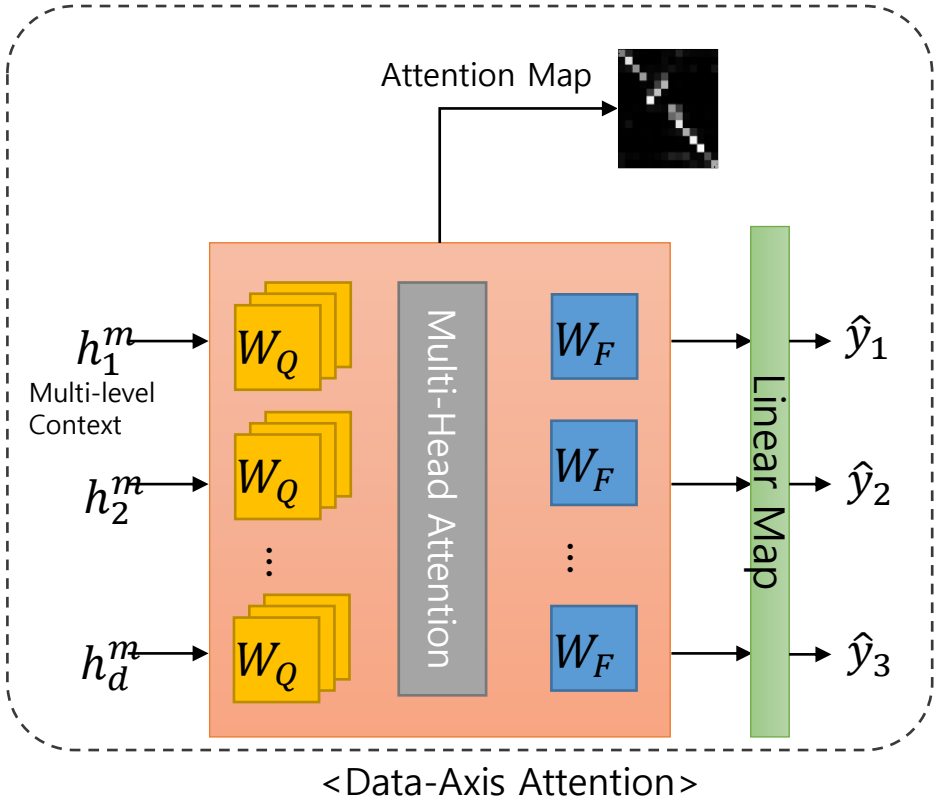
- 주식시장의 정보 확산에 대비해 다른 Query와 Key를 학습함으로써 **비대칭적인 Attention score를 학습할 수 있음**.

- AttentionLSTM이 최근의 지역적인 정보에 초점을 두는 것과는 대비되는 것으로, Transforemr encode는 **지역적인 정보에 초점을 두지 않음**

- Final Prediction

$$\hat{y} = \sigma(H_p W_p + b_p).$$

- 각 값들을 확률로 해석하고, 직접적인 방향예측에 적용하기 위해 logistic sigmoid를 사용함.



<Data-Axis Attention>

Experiment Setting

- Training with Selective Regularization

- training

$$\mathcal{L}(\mathcal{X}, \mathrm{y}) = -\frac{1}{d}\sum_{u}\left(y_u \log \hat{y}_u + (1 - y_u)\log(1 - \hat{y}_u)\right)$$

- 본 논문의 제안된 Structure인 DTML은 주식 예측값과 실제 값 사이의 Cross Entropy Loss를 최소하는 방향으로 Training함
- $\mathcal{X} \in \mathbb{R}^{w \times d \times l}$ is an input tensor of the current time step, and y is the true stock movements

- Selective Regularization

$$\mathcal{L}_{\mathrm{reg}}(\mathcal{X}, \mathrm{y}) = \mathcal{L}(\mathcal{X}, \mathrm{y}) + \lambda(\|\mathbf{W}_{\mathrm{p}}\|_{\mathrm{F}}^2 + \|\mathbf{b}_{\mathrm{p}}\|_2^2)$$

- Regularization은 deep learning trainin시 overfitting을 줄이기 위해 많이 사용되지만. λ 의 최적값을 찾는 데 어려움이 있음
- 마지막 Predictor의 parameter에만 페널티를 부여하는 방식을 사용
- 그 결과로 output space는 제한하지만, 표현력은 보존함.

Experiment Setting

- Experiment Setting

  - Datasets

Dataset	Country	Stocks	Days	From	To
ACL18[1]	US	87	504	2014-01-01	2015-12-31
KDD17[1]	US	50	2,518	2007-01-01	2016-12-31
NDX100	US	95	1,259	2013-01-01	2017-12-31
CSI300	China	219	1,119	2015-06-01	2019-12-31
NI225	Japan	51	856	2016-07-01	2019-12-31
FTSE100	UK	24	1,134	2014-01-01	2018-06-30

  - ACL18 and KDD17 are public datasets

  - 다음날의 움직임을 예측하는 것을 목표로 했고

  - 상승(y=1) or 하락(y=0) 으로 분류됨.

Experiment Setting

- Experiment Setting
  - Feature Vectors

Features	Calculation
z_{open}	$z_{\text{open}} = \text{open}_t/\text{close}_t - 1$
z_{high}	$z_{\text{high}} = \text{high}_t/\text{close}_t - 1$
z_{low}	$z_{\text{low}} = \text{low}_t/\text{close}_t - 1$
z_{close}	$z_{\text{close}} = \text{close}_t/\text{close}_{t-1} - 1$
$z_{\text{adj_close}}$	$z_{\text{adj_close}} = \text{adj_close}_t/\text{adj_close}_{t-1} - 1$
z_{d5}, z_{d10} z_{d15}, z_{d20} z_{d25}, z_{d30}	e.g., $z_{dk} = \frac{\sum_{i=0}^{k} \text{adj_close}_{t-i}}{k \cdot \text{adj_close}_t} - 1$

- z_{ut}는 주식 u의 t 시점(날짜)의 트렌드를 묘사함.

- 위 값들은 종가에 대한 open, high, low 등의 상대적인 값들을 의미함

- zdk represents a long-term trend of the adjusted closing prices during the previous k days.

- Evaluation
  - Accuracy (ACC)
  - Matthews correlation coefficient (MCC)

- Hyperparameters

Window size (w)	10, 15
market context weight	0.01, 0.1, 1
number of epochs	100, 200
learning rate	0.001, 0.0001
strength λ of selective regularization	1
dropout rate to	0.15
optimizer	Adam

Experiment Result

- Prediction Accuracy

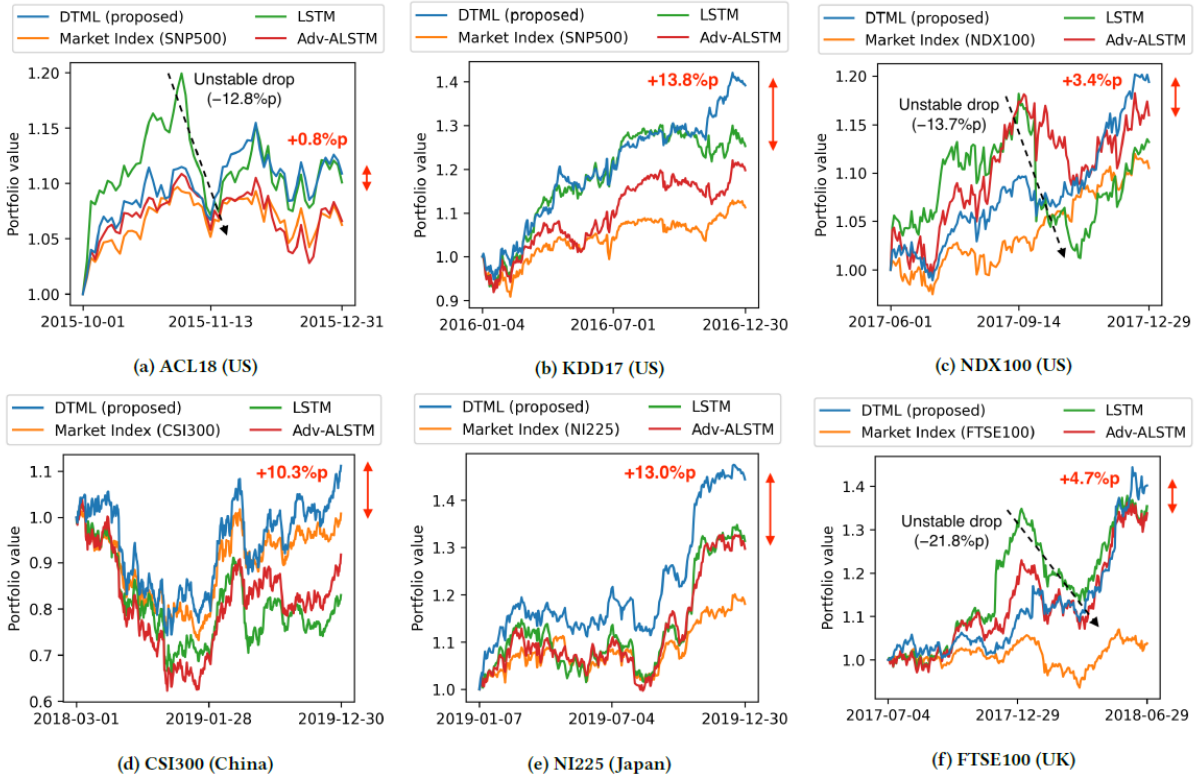
Model	ACL18 (US)		KDD17 (US)		NDX100 (US)	
	ACC	MCC	ACC	MCC	ACC	MCC
LSTM [24]	0.4987 ± 0.0127	0.0337 ± 0.0398	0.5118 ± 0.0066	0.0187 ± 0.0110	0.5263 ± 0.0003	0.0037 ± 0.0049
ALSTM [31]	0.4919 ± 0.0142	0.0142 ± 0.0275	0.5166 ± 0.0041	0.0316 ± 0.0119	0.5260 ± 0.0007	0.0028 ± 0.0084
StockNet [31]	0.5285 ± 0.0020	0.0187 ± 0.0011	0.5193 ± 0.0001	0.0335 ± 0.0050	0.5392 ± 0.0016	0.0253 ± 0.0102
Adv-ALSTM [9]	0.5380 ± 0.0177	0.0830 ± 0.0353	0.5169 ± 0.0058	0.0333 ± 0.0137	0.5404 ± 0.0003	0.0046 ± 0.0090
DTML (proposed)	**0.5744 ± 0.0194**	**0.1910 ± 0.0315**	**0.5353 ± 0.0075**	**0.0733 ± 0.0195**	**0.5406 ± 0.0037**	**0.0310 ± 0.0193**

Model	CSI300 (China)		NI225 (Japan)		FTSE100 (UK)	
	ACC	MCC	ACC	MCC	ACC	MCC
LSTM [24]	0.5367 ± 0.0038	0.0722 ± 0.0050	0.5079 ± 0.0079	0.0148 ± 0.0162	0.5096 ± 0.0065	0.0187 ± 0.0129
ALSTM [31]	0.5315 ± 0.0036	0.0625 ± 0.0076	0.5060 ± 0.0066	0.0125 ± 0.0139	0.5106 ± 0.0038	0.0231 ± 0.0077
StockNet [31]	0.5254 ± 0.0029	0.0445 ± 0.0117	0.5015 ± 0.0054	0.0050 ± 0.0118	0.5036 ± 0.0095	0.0134 ± 0.0135
Adv-ALSTM [9]	0.5337 ± 0.0050	0.0668 ± 0.0084	0.5160 ± 0.0103	0.0340 ± 0.0201	0.5066 ± 0.0067	0.0155 ± 0.0140
DTML (proposed)	**0.5442 ± 0.0035**	**0.0826 ± 0.0074**	**0.5276 ± 0.0103**	**0.0626 ± 0.0230**	**0.5208 ± 0.0121**	**0.0502 ± 0.0214**

- 본 논문에서 제안된 DTML 알고리즘이 이전에 높은 정확성을 보였던 Adv-ALSTM에 비해 더 나은 일반화가 가능했고, 정확성에서 좋은 성과를 보임.

Experiment Result

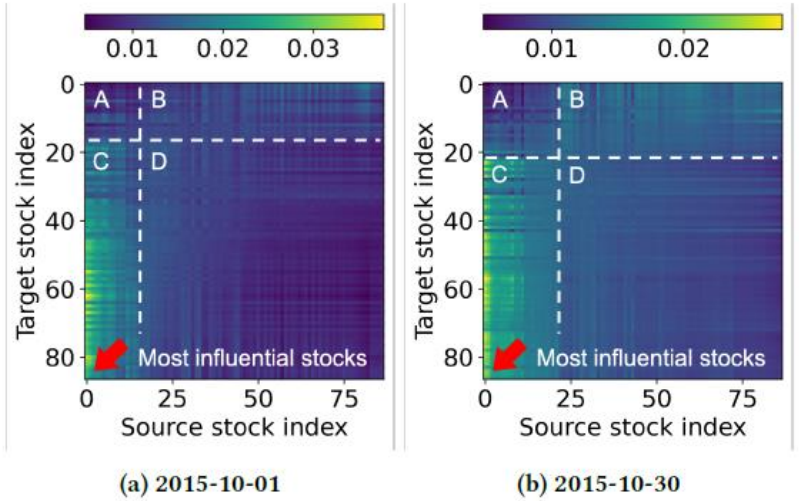
- Invest simulation



- SNP500 is used for the ACL18 and KDD17 datasets 데이터셋을 비교모델 중 최상의 성과와 비교한 결과 13.8%p 더 높은 성과를 보였고
- 나머지 시뮬레이션에서도 좋은 결과를 보임으로써, 비교모델과 비교해 더 일반화된 모델임을 알 수 있음.

Experiment Result

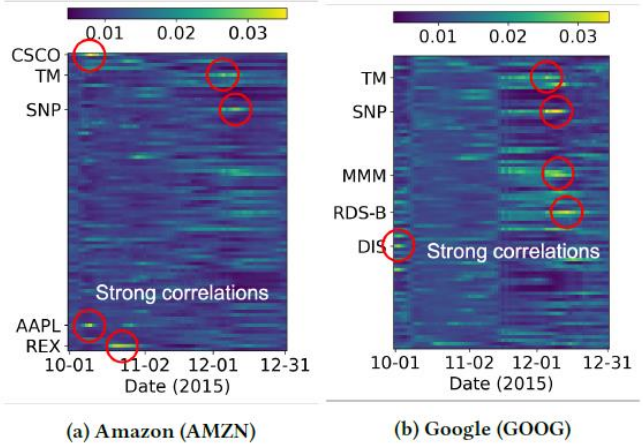
- Interpreting Attention Maps



(a) 2015-10-01 (b) 2015-10-30

- 위의 Attention Map을 분석한 결과, c에 해당하는 소수의 주식이 대부분의 주식 예측 결과에 영향을 미치는 것을 알 수 있음.
- B영역은 대부분의 주식이 다른 주식의 예측에 영향을 주는 것을 알 수 있음.

<전체 주식간의 attention score>



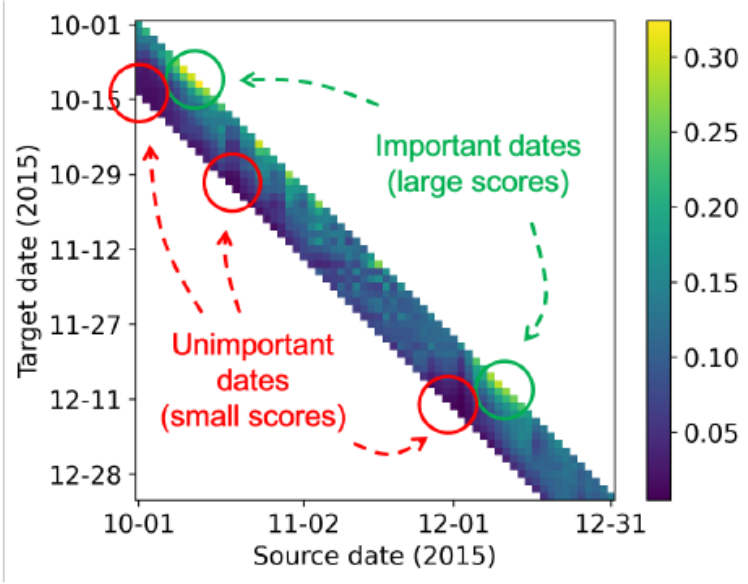
(a) Amazon (AMZN) (b) Google (GOOG)

- 위의 Attention Map을 분석한 결과 시간에 따라 특정 주식에 대한 가중치가 달라짐을 알 수 있음.
- 이전에는 연관 지을 수 없었던 다른 주식과의 상관성을 알 수 있음.

<아마존과 구글 주식의 다른 주식에 대한 attention score>

Experiment Result

- Interpreting Attention Maps



<구글 주식의 target data 와 source data에 관한 attention score>

- 초록색은 Attention 값이 큰 부분을 나타내고, 붉은색 부분은 Attention 값이 적은 부분을 나타냄
- 위의 Attention Map은 본 DTML 모델이 다른 주식 뿐만 아니라 해당 주식 자체의 정보 또한 이용함을 알 수 있음.

Conclusion

- DTML

  - 상술된 다양한 데이터셋과 평가 metric 에서 비교모델들에 비해 전반적으로 좋은 성능을 보임.

  - 비교 모델에 비해 일반적으로 성능이 좋은 모델임을 알 수 있음.

- Issue of DTML

  - 일반적인 시계열 값을 넣지 않고 Context를 만들어 넣은 이유에 대한 의문점.

  - Context Normalization에서 학습 가능한 Parameter를 추가해야 되는지에 대한 의문점.

Reference

1 James H. Stock and Mark W. Watson. 2001. Vector Autoregressions. Journal of Economic Perspectives. 101-115. 10.1257/jep.15.4.101

2 Jaemin Yoo, Yejun Soun, Yong-chan Park and U Kang. 2021. Accurate Multivariate Stock Movement Prediction via Data-Axis Transformer with Multi-Level Contexts. ACM SIGKDD International Conference on Knowledge discovery and data mining. 2037–2045. https://doi.org/10.1145/3447548.3467297

3 Yifeng Zhang, Ka-Ho Chow and S.-H. Gary Chan. 2019. DA-LSTM: A Long Short-Term Memory with Depth Adaptive to Non-uniform Information Flow in Sequential Data. Neural and Evolutionary Computing. arXiv:1903.02082

4 Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. 2020. Spectral Temporal Graph Neural Network for Multivariate Time-series Forecasting. NeurIPS.

5 Chang Li, Dongjin Song, and Dacheng Tao. 2019. Multi-task Recurrent Neural Networks and Higher-order Markov Random Fields for Stock Price Movement Prediction. KDD.

6 Wei Li, Ruihan Bao, Keiko Harimoto, Deli Chen, Jingjing Xu, and Qi Su. 2020. Modeling the Stock Relation with Graph Network for Overnight Stock Movement Prediction. IJCAI.

7 Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, Tat-Seng Chua. 2019. Enhancing Stock Movement Prediction with Adversarial Training. IJCAI.

8 Wenjie Lu, Jiazheng Li, Jingyang Wang & Lele Qin. 2021. A CNN-BiLSTM-AM method for stock price prediction. Neural Computing and Applications

Reference

9. Wei Bao,Jun Yue ,Yulei Rao. 2017. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. Plos one