به نام خدا

دانشگاه تهران

دانشکده مهندسی

برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق

تمرین چهارم

نام و نام خانوادگی	بهراد موسایی شیرمحمد - محمد جواد رنجبر کلهرودی
شماره دانشجویی	۸۱۰۱۰۱۲۷۸ - ۸۱۰۱۰۱۱۷۳
تاریخ ارسال گزارش	۱۴۰۲.۰۹.۱۷

فهرست

پاسخ ۱. پیشبینی سری زمانی

پیش بینی را میتوان به دو دسته regression و classification تقسیم کرد در regression مقدار عددی روزهای بعدی برای یک سری زمانی پیشبینی می.شود در classification صعودی یا نزولی بودن روند سری زمانی برای چند روز آینده پیش بینی می.شود در این سوال شما با حالت regression آشنا خواهید شد و بخشی از این مقاله را پیاده سازی میکنید.

۱-۱. دانلود داده ها

ابتدا با استفاده از ویکی‌پدیا داده‌های S&P ۶۰۰ را دانلود کرده و به فرم یک دیتافریم در می‌آوریم. این اسکریپت فهرست شرکت‌های S&P ۶۰۰ را از ویکی‌پدیا بازیابی می‌کند و آنها را فیلتر می‌کند تا فقط شرکت‌هایی را که قبل از ۱ ژانویه ۲۰۱۰ به فهرست اضافه شده‌اند را شامل شود. نمادهای سهام این شرکت‌ها سپس در فهرستی استخراج می‌شوند. با استفاده از «yfinance»، این اسکریپت داده‌های سهام تاریخی این نمادها را از ۱ ژانویه ۲۰۱۰ به بعد در یک بازه زمانی روزانه دانلود می‌کند.

		Open	High	Low	Close	Adj Close	Volume	Open	High	Low	...	Low	Close	Adj Close	Volume	Open
0	2010-01-04	47.700001	48.740002	47.700001	48.330002	35.464890	654600	60.730000	61.180000	60.230000	...	47.450001	47.500000	45.265305	7182800	19.770000
1	2010-01-05	48.349998	48.500000	47.439999	48.000000	35.222721	892100	60.939999	61.040001	59.259998	...	47.250000	47.660000	45.417786	3221900	19.809999
2	2010-01-06	47.820000	48.740002	47.799999	48.490002	35.582302	776300	59.540001	59.779999	57.709999	...	47.619999	48.110001	45.846619	3065000	19.850000
3	2010-01-07	48.299999	48.770000	47.930000	48.630001	35.685047	814200	58.770000	59.630001	58.680000	...	47.759998	48.110001	45.846619	2638300	20.080000
4	2010-01-08	48.779999	48.799999	48.250000	48.689999	35.729069	529700	59.480000	60.029999	59.119999	...	47.880001	48.919998	46.618507	3162200	20.040001
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3513	2023-12-18	208.270004	210.630005	208.179993	209.800003	209.800003	1963900	135.779999	136.690002	135.039993	...	515.599976	518.619995	518.619995	1607300	42.049999
3514	2023-12-19	209.800003	211.139999	209.339996	210.119995	210.119995	1758000	136.050003	136.850006	134.619995	...	519.809998	528.140015	528.140015	2569400	42.380001
3515	2023-12-20	209.970001	211.990005	206.529999	206.580002	206.580002	920300	136.690002	136.690002	134.690002	...	519.369995	519.429993	519.429993	1497300	41.290001
3516	2023-12-21	207.779999	209.309998	206.399994	207.539993	207.539993	811500	135.190002	136.119995	134.550003	...	518.070007	526.559998	526.559998	1016600	40.410000
3517	2023-12-22	208.559998	209.679993	207.850006	208.490005	208.490005	741800	136.699997	137.210007	136.110001	...	526.900024	529.049988	529.049988	1081500	40.419998

شکل ۱: داده ها

۱-۲. کاوش در داده های سری زمانی و آشنایی با تئوریها و کتابخانه های معروف

در این داده‌ها تعداد زیادی مقادیر null موجود است:

```
null_values = stock_data.isnull().sum()
# Display columns with null values and their coun
print(null_values[null_values > 0])
```

```
HWM   Open         1720
      High         1720
      Low          1720
      Close        1720
      Adj Close    1720
      Volume       1720
WRK   Open         1377
      High         1377
      Low          1377
      Close        1377
      Adj Close    1377
      Volume       1377
BF.B  Open         3518
      High         3518
      Low          3518
      Close        3518
      Adj Close    3518
      Volume       3518
dtype: int64
```

شکل **۲** مقادیر **null** در داده‌ها

وجود مقادیر نال (null values) در وسط یا انتهای سری زمانی می‌تواند باعث مشکلاتی در تحلیل و پردازش داده‌ها شود. برای حل این مشکلات و استفاده صحیح از داده‌های سری زمانی با مقادیر نال، می‌توان از روش‌های زیر استفاده کرد:

حذف مقادیر نال: در صورتی که تعداد مقادیر نال محدود باشد و تأثیر آن بر تحلیل نادیده گرفتنی باشد، می‌توان مقادیر نال را حذف کرد. این روش در مواردی که تغییرات زمانی کمتری در داده‌ها وجود داشته باشد، مطمئن‌تر است.

پر کردن مقادیر نال با مقادیر قبلی یا بعدی: در صورتی که تعداد مقادیر نال محدود باشد و تأثیر آن بر تحلیل اهمیتی نداشته باشد، می‌توان مقادیر نال را با مقدار قبلی یا بعدی غیر نال جایگزین کرد. این روش، برای تحلیل داده‌های متغیر زمانی با تغییرات نسبتاً کمتر مفید است.

تخمین مقادیر نال با استفاده از روش‌های آماری: در صورتی که تعداد مقادیر نال بیشتر باشد یا روش‌های قبلی مناسب نباشند، می‌توان از روش‌های آماری مانند میانگین، مد، روش‌های تقسیم بندی مقادیر (binning) و یا مدل‌های پیش‌بینی استفاده کرد تا مقادیر نال را تخمین بزنیم و جایگزین کنیم.

استفاده از روش‌های پیش‌بینی و مدلسازی: در صورتی که مقادیر نال بسیار زیاد باشند و روش‌های قبلی قابل استفاده نباشند، می‌توان از روش‌های پیش‌بینی و مدلسازی مانند مدل‌های خطی، مدل‌های زمانی (time series models)، شبکه‌های عصبی و یا روش‌های ماشینی استفاده کرد تا مقادیر نال را پیش‌بینی و پر کنیم.

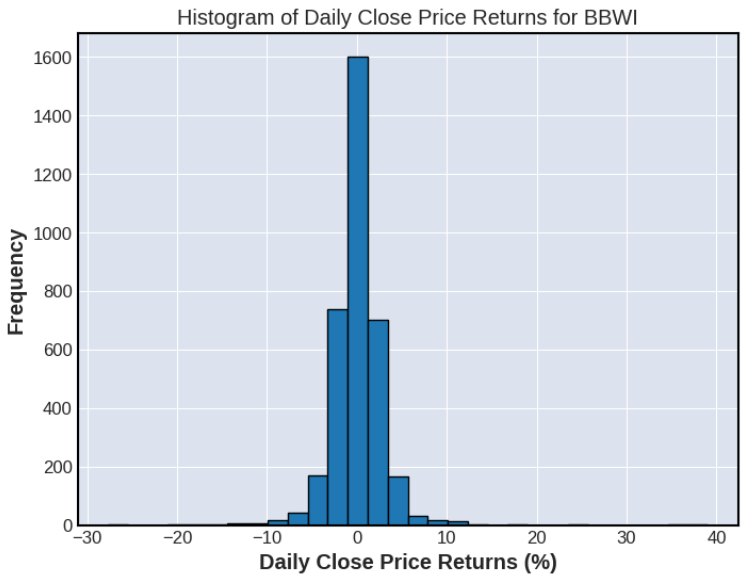
با توجه به ماهیت داده‌های بورسی برای تخمین داده‌های null، استفاده از دو راه حل منطقی به نظر می‌رسید، ۱- جایگزینی با میانگین چند روز اخیر، ۲- جایگزنی به استفاده از forward filling و backward filling ما در این پروژه از راه حل دوم استفاده کردیم.

همچنین مشخص است که برای داده‌هایی که در ابتدا و انتهای سری زمانی هستن این روش همچنان جواب خواهد داد.

مقدار close price return به صورت زیر تعریف می‌شود:

$$Close\ Price\ return = \frac{Close\ price_{Current} - Close\ price_{previous}}{Close\ price_{Current}} * 100$$

حال برای یکی از سهام‌ها به صورت تصادفی این مقادر را محاسبه می‌کنیم و نمودارهای مربوطه را رسم می‌کنیم.

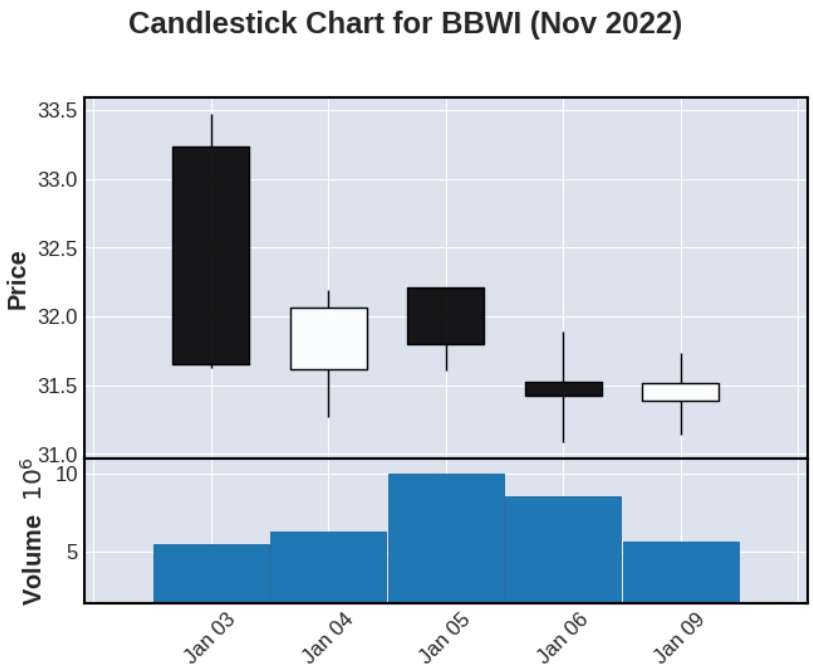


شکل ۳: هیستوگرام **BBWI**

نظریه حرکت تصادفی بیان می کند که قیمت سهام تصادفی است، به طوری که حرکت یا روند گذشته قیمت سهام یا بازار نمی تواند برای پیش بینی حرکت آینده آن استفاده شود. با این وجود با بررسی نمودار

نسبتا نورمال می‌توان نتیجه گرفت که مقادیر روزهای قبل در مقدار روز قبل تقریبا اثر دارند و اگر توزیع این نمودار متمرکز نبود صحت برای random walk theory بود و این نمودار این نظریه را رد می‌کند.

همچنین برای چند روز تصادفی candlestick chart به صورت زیر می‌باشد.



شکل۴ : کندل استیک مربوطه

۱-۳. TimeSeriesSplit

از آنجا که نحوه‌ی جداسازی مقادیر برای دیتاست‌های معمولی یادگیری ماشین فرق می‌کند. از کتابخانه‌ی مربوط به این کار استفاده می‌کنیم.

از آنجا که قصد cross validation با k=۵ را داریم. داده‌ها را برای این کار آماده می‌کنیم.

۱-۴. آماده سازی ورودی و خروجی مدل گذشته

حال برای سادگی close price روز پیشین را در پنجره‌هایی در نظر می‌گیریم و قصد پیش‌بینی قیمت روز بعدی یا horizon را داریم. حال گروه منتخبی از سهام‌ها شامل ""AAPL"، "MSFT"، "AMZN"، "META"، ""GOOGL") را انتخاب می‌کنیم. حال به صورت پنجره‌ای داده‌های ده روز قبل را به عنوان ویژگی و داده‌ی روز قبل را به عنوان lable نگهداری می‌کنیم.

همچنین از آنجا که قصد cross validation داریم هربار ۴ بخش از داده‌ها را به عنوان داده‌ی آموزش و یک بخش را به عنوان داده‌ی تست کنار می‌گذاریم.

داده‌ها را یک بار به صورت نورمال شده و یک بار به صورت نورمال نشده استفاده می‌کنیم.

نحوه‌ی نورمالیزیشن داده به این صورت است که با استفاده داده‌های آموزش پارامترهای نورمالزیشن را به دست آورده و با داده‌های تست را نورمال می‌کنیم.

۱-۵. مدلهای شبکه عصبی حافظه دار

LSTM (Long Short-Term Memory), GRU (Gated Recurrent Unit), و Bi-LSTM (Bidirectional LSTM) همگی مدل‌های شبکه عصبی بازگشتی (Recurrent Neural Networks - RNN) هستند که برای پردازش داده‌های دنباله‌ای مانند سری زمانی و متن استفاده می‌شوند. این مدل‌ها با استفاده از ساختارهای خاصی به نام "Gate"، قابلیت یادگیری و حفظ اطلاعات بلندمدت را دارند.

LSTM (Long Short-Term Memory):

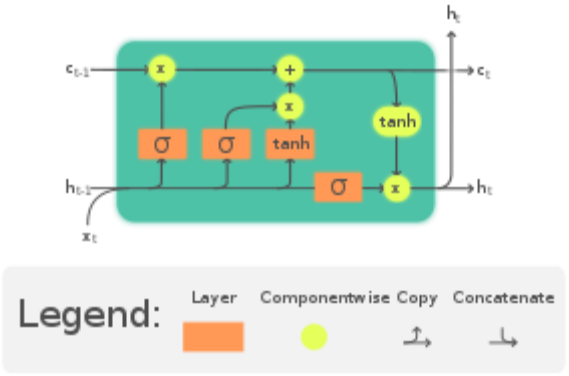
LSTM با استفاده از سه Gate اصلی، یعنی ""Forget Gate"، "Input Gate"" و " Output Gate" عملکرد خود را کنترل می‌کند. این Gateها به صورت مجزا عمل می‌کنند:

Forget Gate: این Gate تصمیم می‌گیرد که کدام اطلاعات باید از حافظه قبلی حذف شوند.

Input Gate: این Gate تصمیم می‌گیرد کدام اطلاعات جدید باید به حافظه اضافه شوند.

Output Gate: این Gate تصمیم می‌گیرد که چه قدر از حافظه باید در خروجی نهایی استفاده شود.

با استفاده از این سه Gate و حالت سلول (Cell State)، LSTM قادر است به خوبی بازدهی طولانی‌مدت را یاد بگیرد و مشکل "گرادیان محو شونده" (vanishing gradient) در شبکه‌های عصبی بازگشتی را کاهش دهد.



شکل ۵ **LSTM**

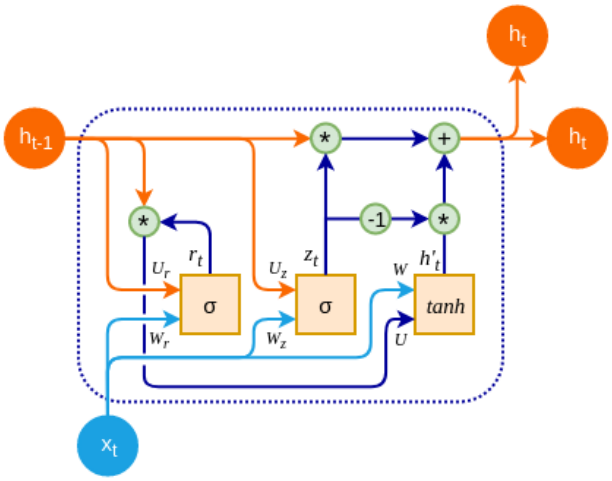
GRU (Gated Recurrent Unit):

GRU مانند LSTM نیز از Gateها برای کنترل جریان اطلاعات استفاده می‌کند. اما در GRU، تعداد Gateها کمتر است و دارای دو Gate اصلی می‌باشد:

Update Gate: این Gate تصمیم می‌گیرد که کدام اطلاعات باید به روزرسانی شوند و در حافظه نگهداری شوند.

Reset Gate: این Gate تصمیم می‌گیرد کدام اطلاعات باید حذف شوند و فراموش شوند.

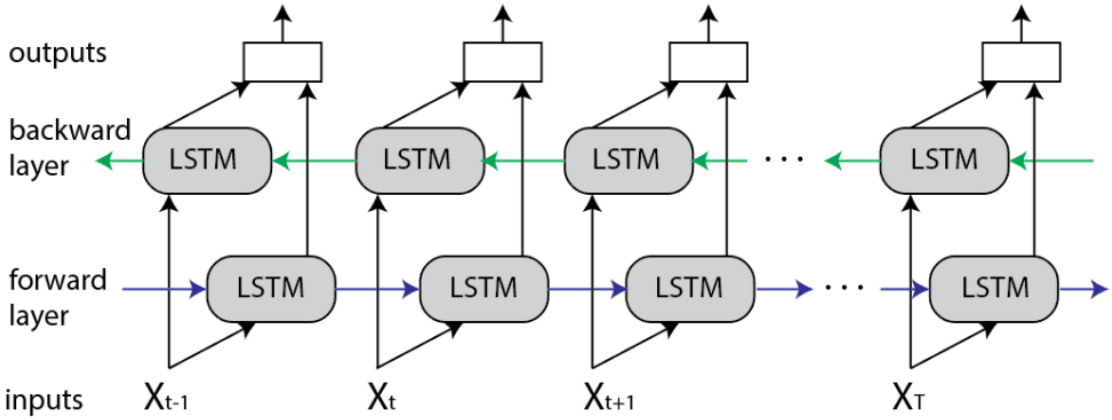
GRU به عنوان یک نسخه ساده‌تر از LSTM محسوب می‌شود و در برخی موارد می‌تواند سرعت بالاتری در آموزش و استنتاج داشته باشد.



شکل ۶ **GRU**

Bi-LSTM (Bidirectional LSTM):

Bi-LSTM دو لایه LSTM را به صورت موازی در نظر می‌گیرد. یک لایه LSTM به صورت معمول جریان اطلاعات را از چپ به راست (از ابتدای دنباله تا انتها) می‌برد و لایه دیگر آن را از راست به چپ (از انتها تا ابتدا) جریان می‌دهد. این روش به مدل امکان می‌دهد از اطلاعات زمانی هر دو جهت برای پیش‌بینی و تحلیل استفاده کند و اطلاعات دنباله‌ای را به طور جامع‌تر درک کند.



شکل ۷ **BILSTM**

تفاوت‌ها و شباهت‌ها:

LSTM و GRU هر دو از ساختارهای Gate برای کنترل جریان اطلاعات در شبکه‌های عصبی بازگشتی استفاده می‌کنند. این ساختارها شامل Gateهای مختلفی هستند که هر کدام وظیفه خاصی در جریان اطلاعات دنباله‌ای دارند.

LSTM با سه Gate Forget Gate، Input Gate، Output Gate و حالت سلول (Cell State)، قابلیت حفظ اطلاعات بلندمدت را دارد و مشکل گرادیان محو شونده را به حداقل می‌رساند.

GRU با دو (Update Gate، Reset Gate) ساده‌تر از LSTM است و در برخی موارد می‌تواند در آموزش و استنتاج سریع‌تر عمل کند.

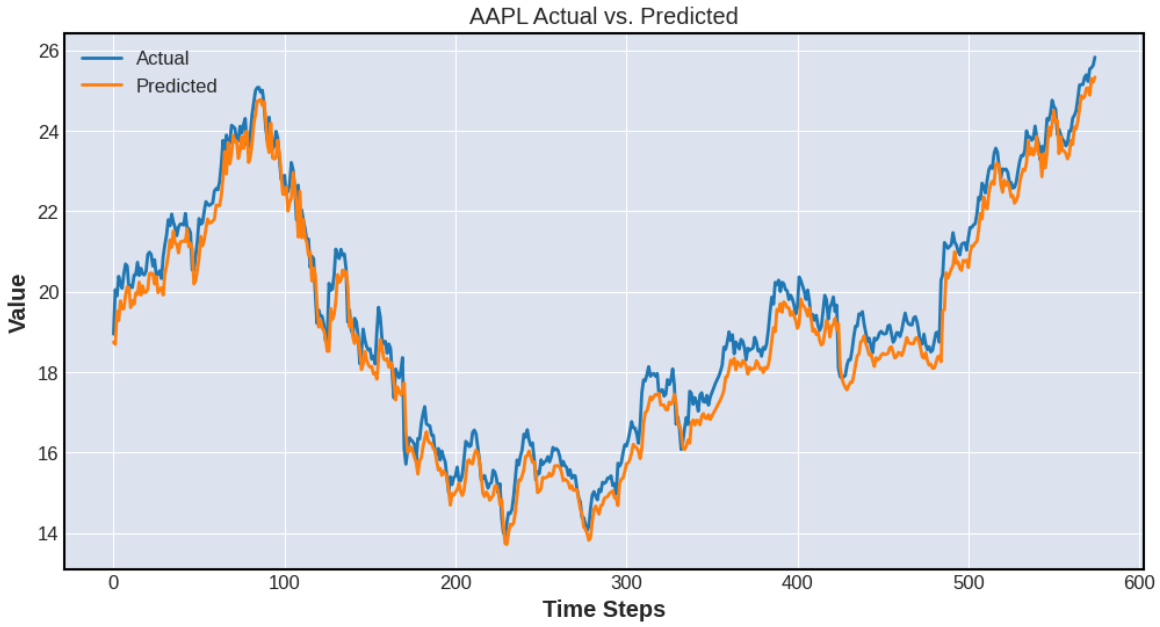
Bi-LSTM با استفاده از دو لایه LSTM، اطلاعات زمانی را به صورت همزمان از جهت چپ به راست و راست به چپ جریان می‌دهد، که به مدل امکان می‌دهد از اطلاعات هر دو جهت برای پیش‌بینی و تحلیل استفاده کند.

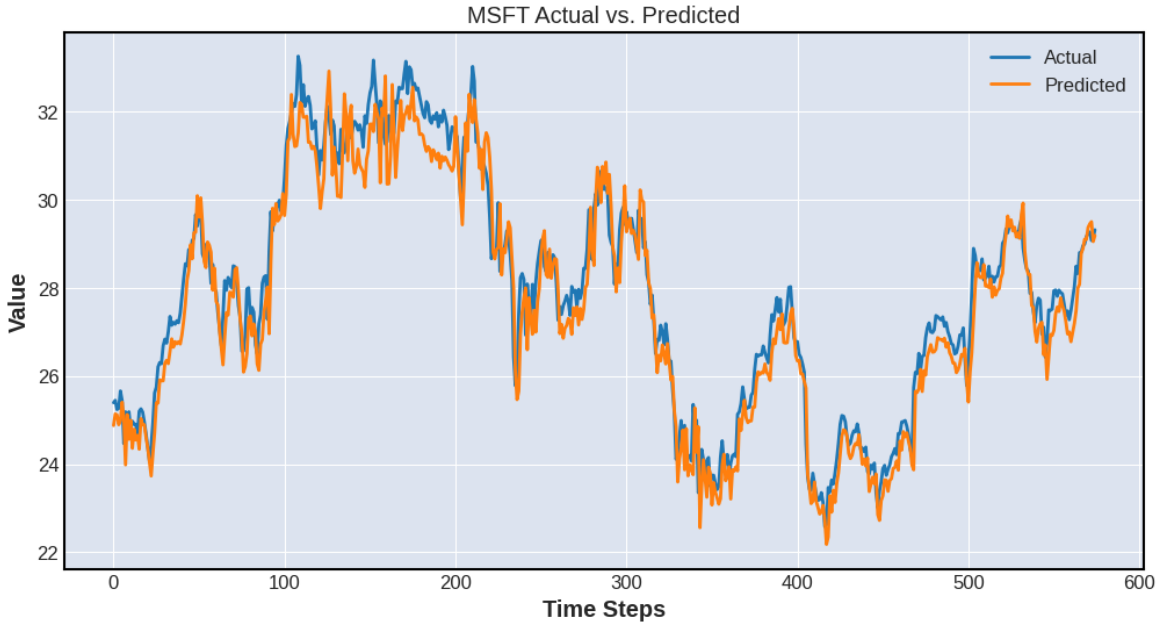
در کل، LSTM و GRU و Bi-LSTM همگی به عنوان مدل‌های قوی و پرکاربرد در حوزه پردازش داده‌های دنباله‌ای شناخته می‌شوند و هر کدام ویژگی‌ها و قابلیت‌های خاص خود را دارند. انتخاب بین این مدل‌ها بسته به مسئله مورد نظر، حجم داده، زمان اجرا، و دیگر عوامل ممکن است.

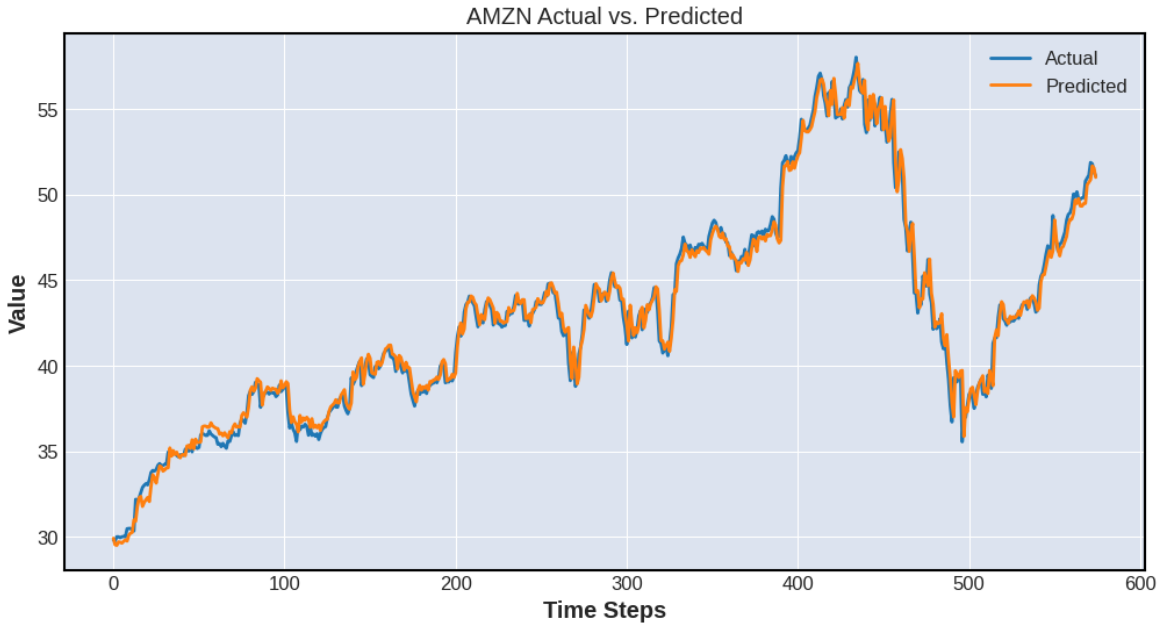
حال تمام این شبکه‌ها را برای ۵ بار داده‌ها مختلف آموزش و تست، آموزش می‌دهیم. و نتایج هرکدام از این مدل‌ها به صورت زیر می‌باشد.

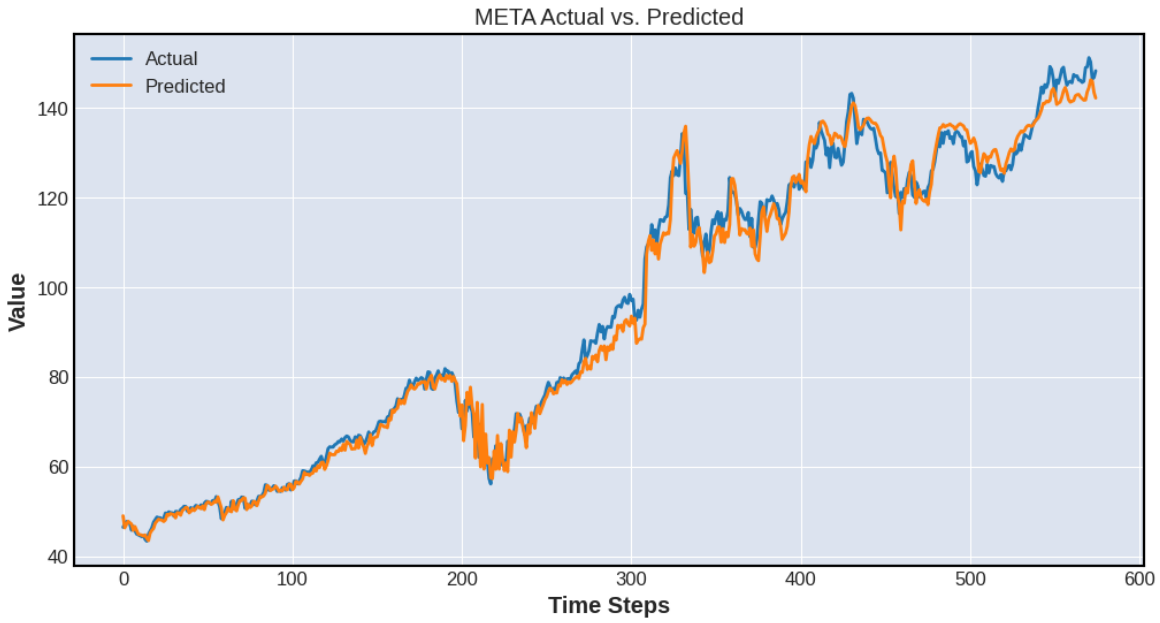
LSTM

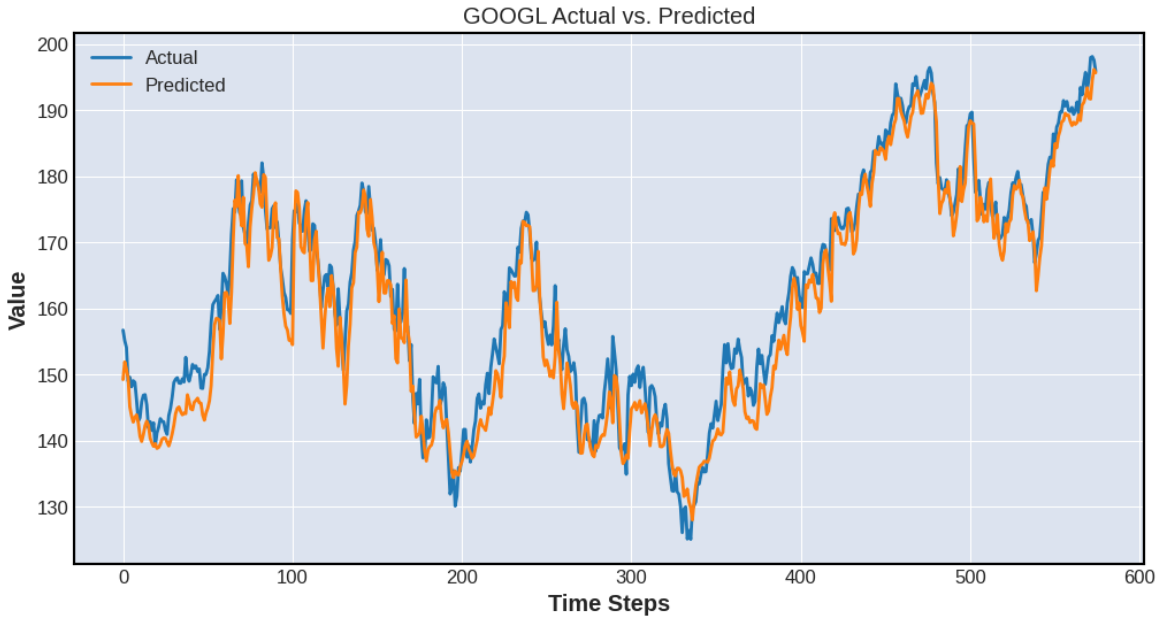
Mean of elements (element-wise): [۲۳۶.۲۲۴۳۲۰۲۲ ۲۳۶.۲۲۴۳۲۰۲۲ ۵.۵۱۹۰۷۶۴۲ ۳.۳۰۸۴۴۷۶۹]






AMZN Actual vs. Predicted
Actual
Predicted
Value
Time Steps
30
35
40
45
50
55
0
100
200
300
400
500
600


META Actual vs. Predicted
Actual
Predicted
Value
Time Steps
40
60
80
100
120
140
0
100
200
300
400
500
600

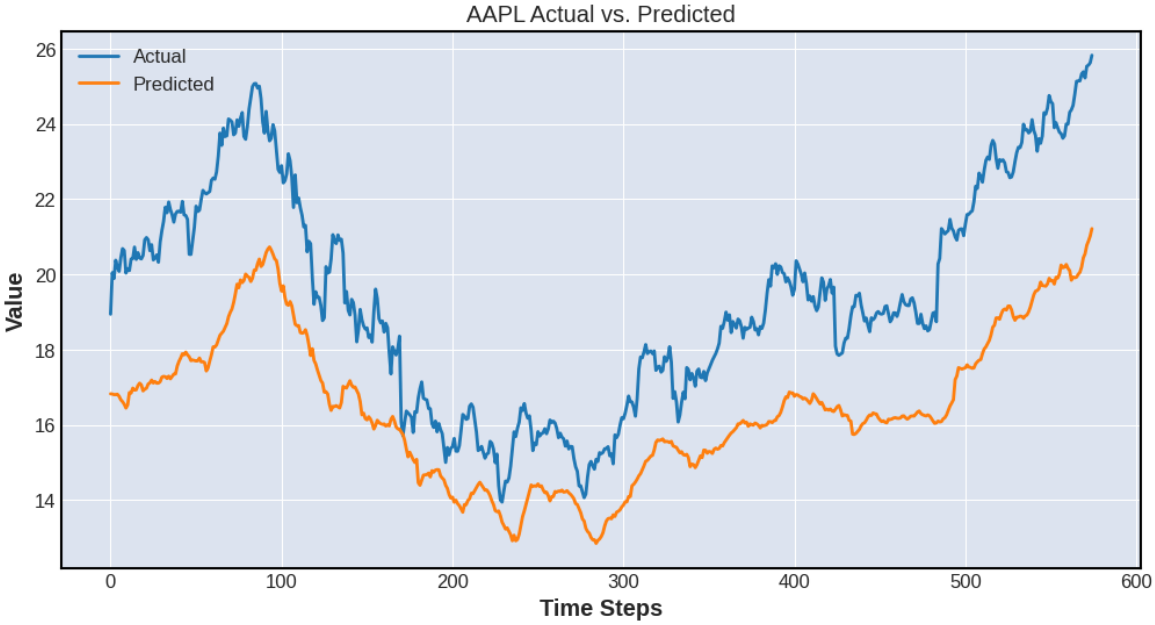


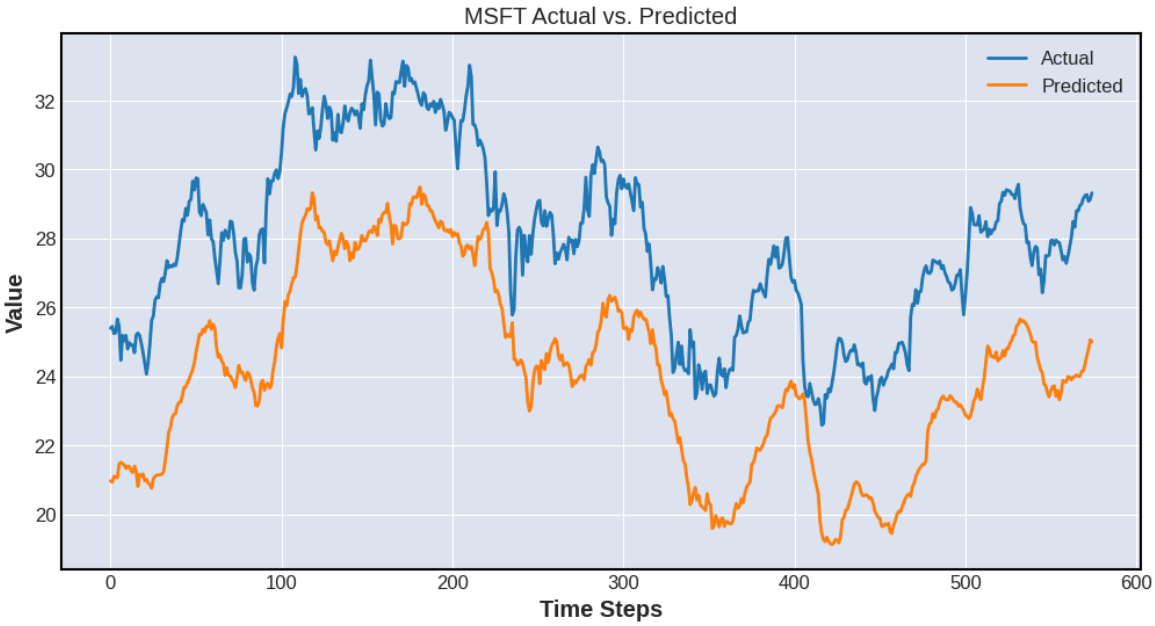
شکل ۸: نتایج **LSTM**

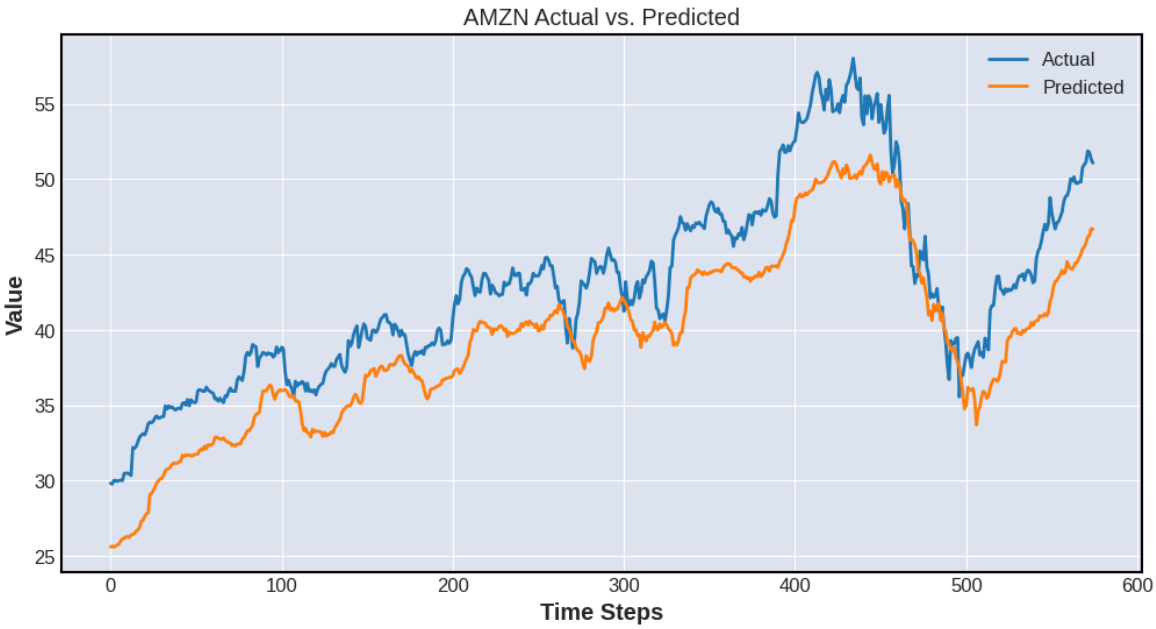
GRU

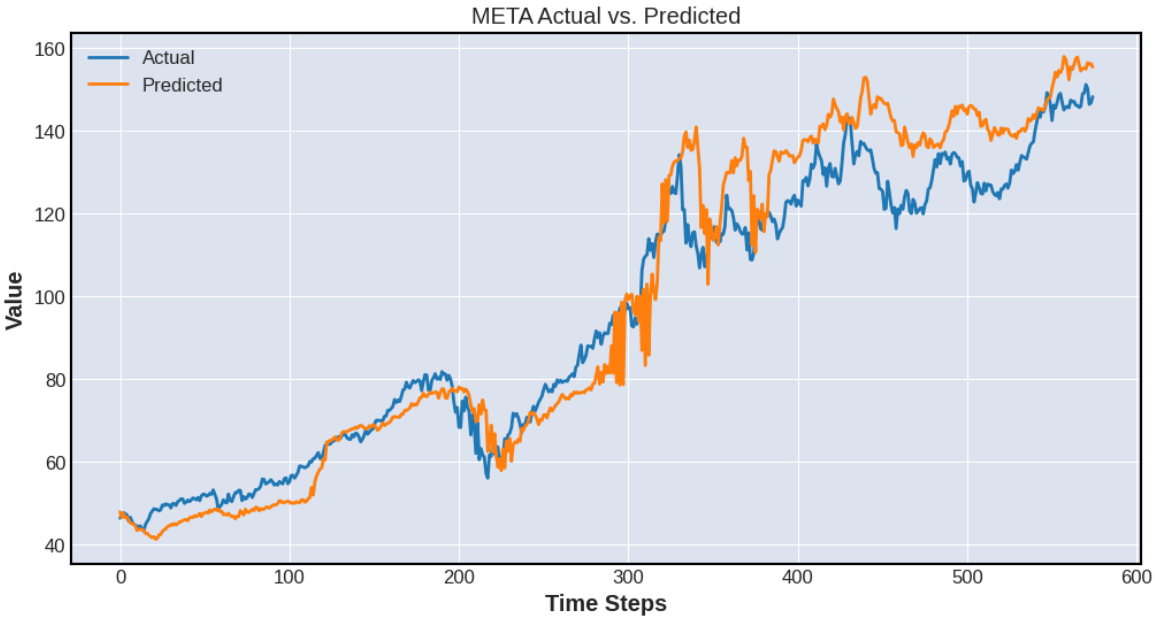
نتایج به صورت زیر خواهد بود:

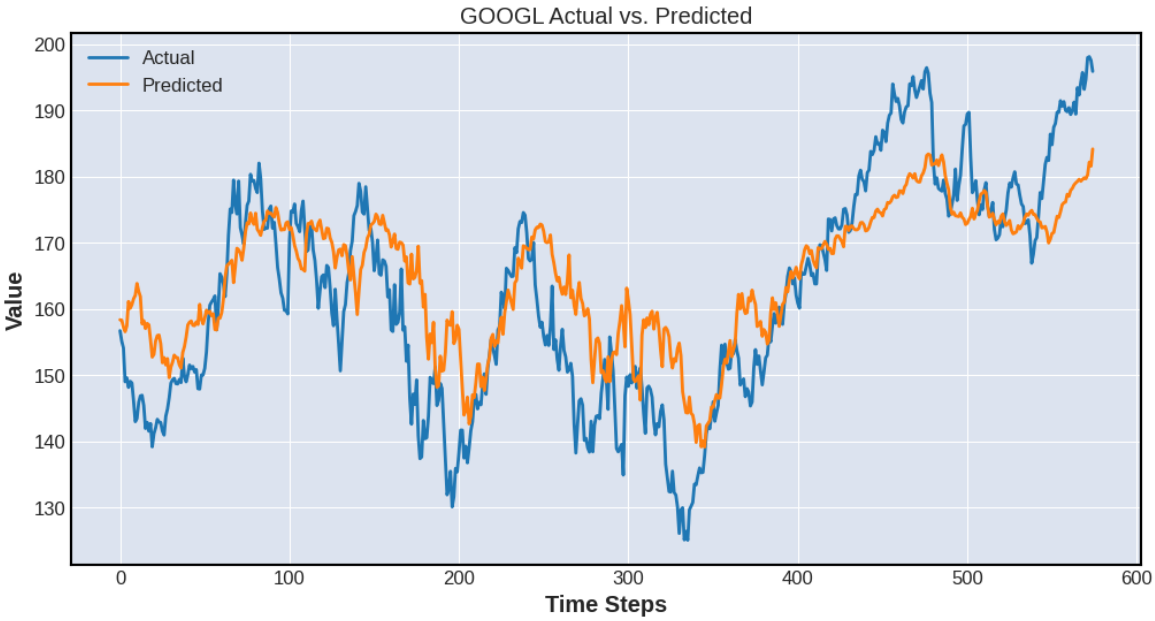
Mean of elements (element-wise): [۲۹۹.۶۶۴۳۱۰۸۴ ۲۹۹.۶۶۴۳۱۰۸۴ ۸.۲۳۵۶۱۵۲۱ ۶.۱۷۱۵۰۰۱۱]




MSFT Actual vs. Predicted
Actual
Predicted
Value
Time Steps
32
30
28
26
24
22
20
0
100
200
300
400
500
600


AMZN Actual vs. Predicted
Actual
Predicted
Value
Time Steps
55
50
45
40
35
30
25
0
100
200
300
400
500
600



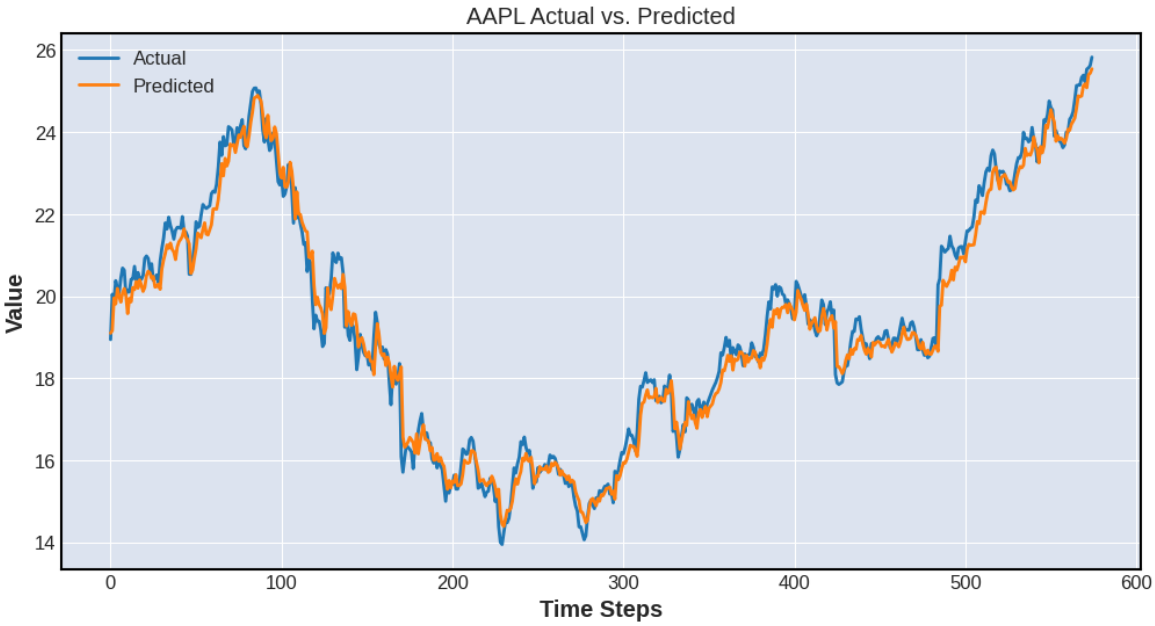


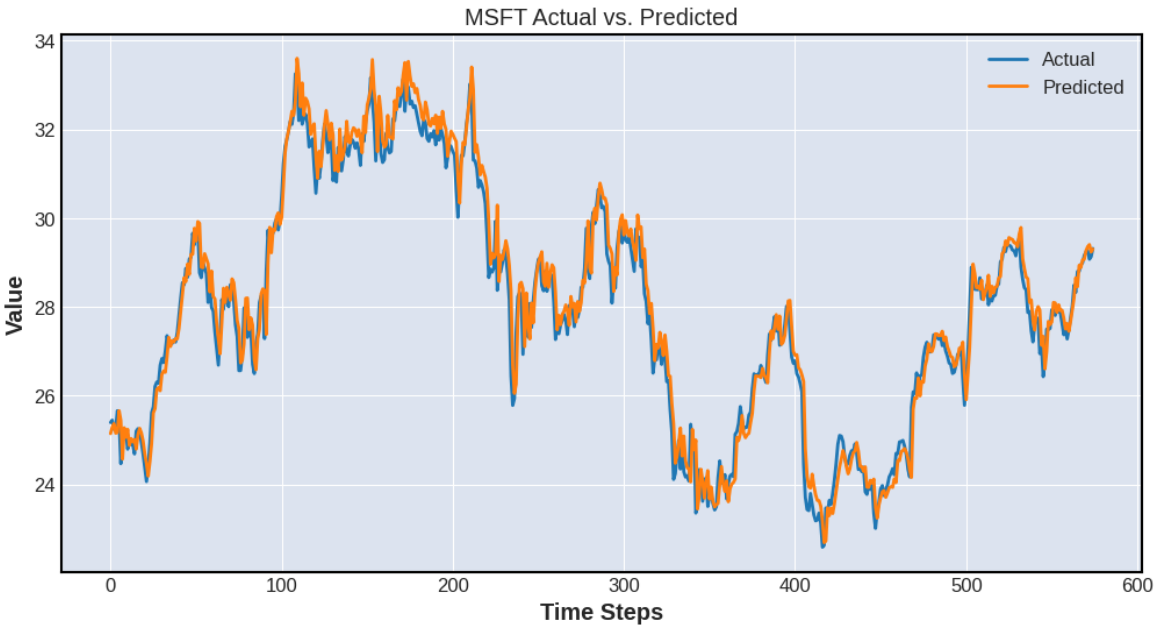
شکل ۹: نتایج **GRU**

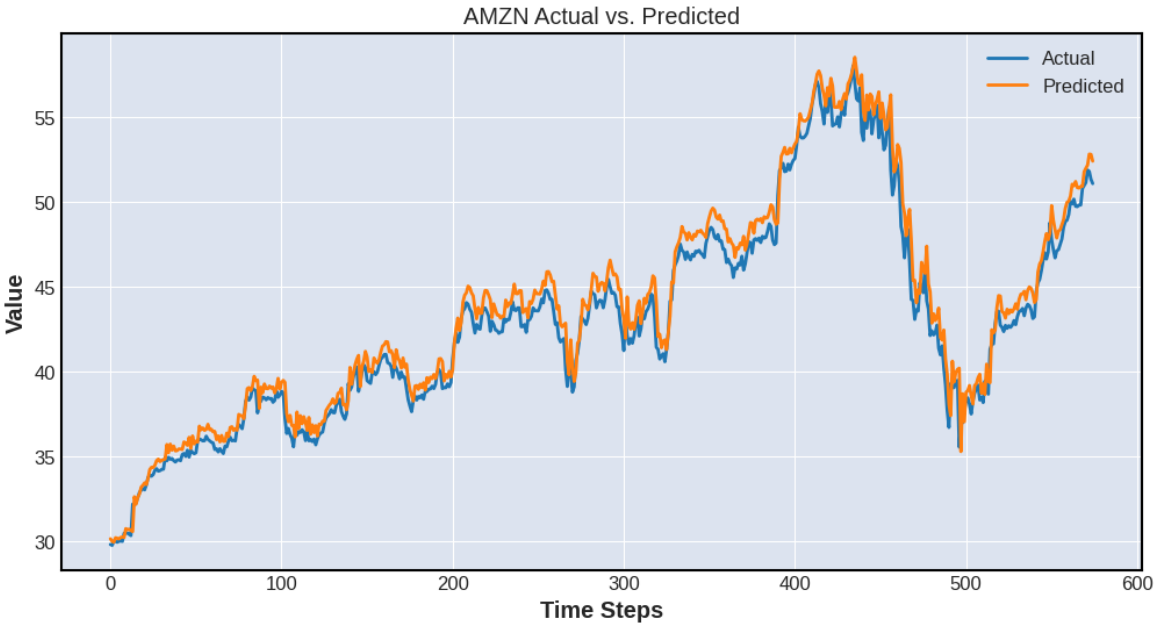
Bi-LSTM

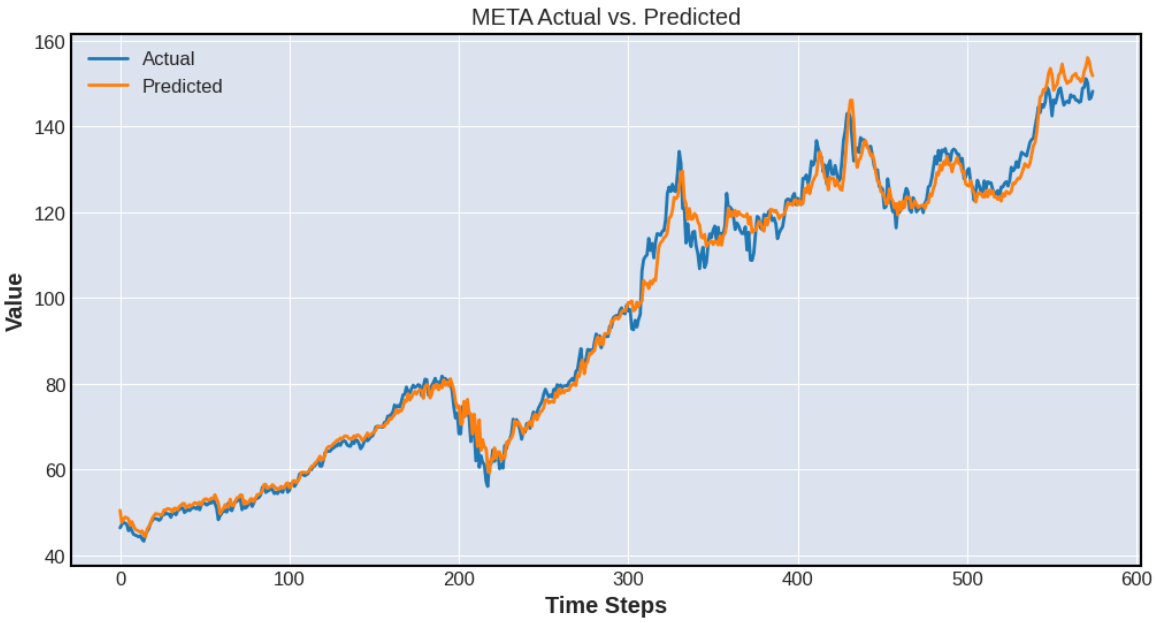
نتایج به صورت زیر خواهد بود:

Mean of elements (element-wise): [۱۳۸.۵۰۹۴۲۱۱۶ ۱۳۸.۵۰۹۴۲۱۱۶ ۵.۶۳۹۸۳۲۷۸ ۴.۱۳۲۰۳۱۴۹]

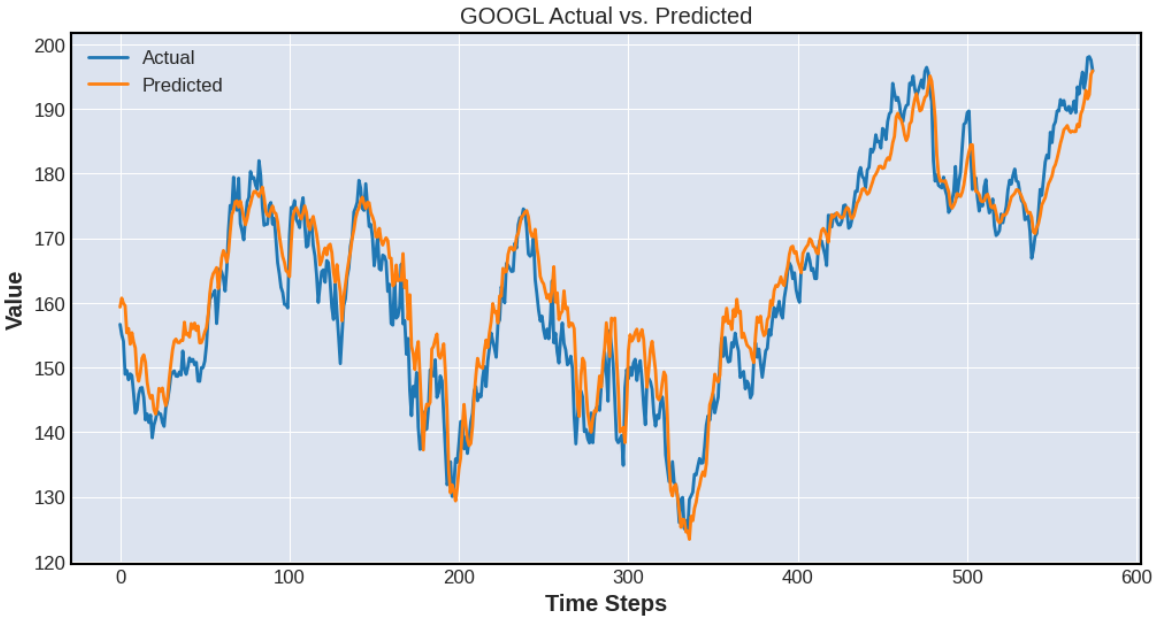

AAPL Actual vs. Predicted
Actual
Predicted
Value
Time Steps
26
24
22
20
18
16
14
0
100
200
300
400
500
600


MSFT Actual vs. Predicted
Actual
Predicted
Value
Time Steps
34
32
30
28
26
24
0
100
200
300
400
500
600





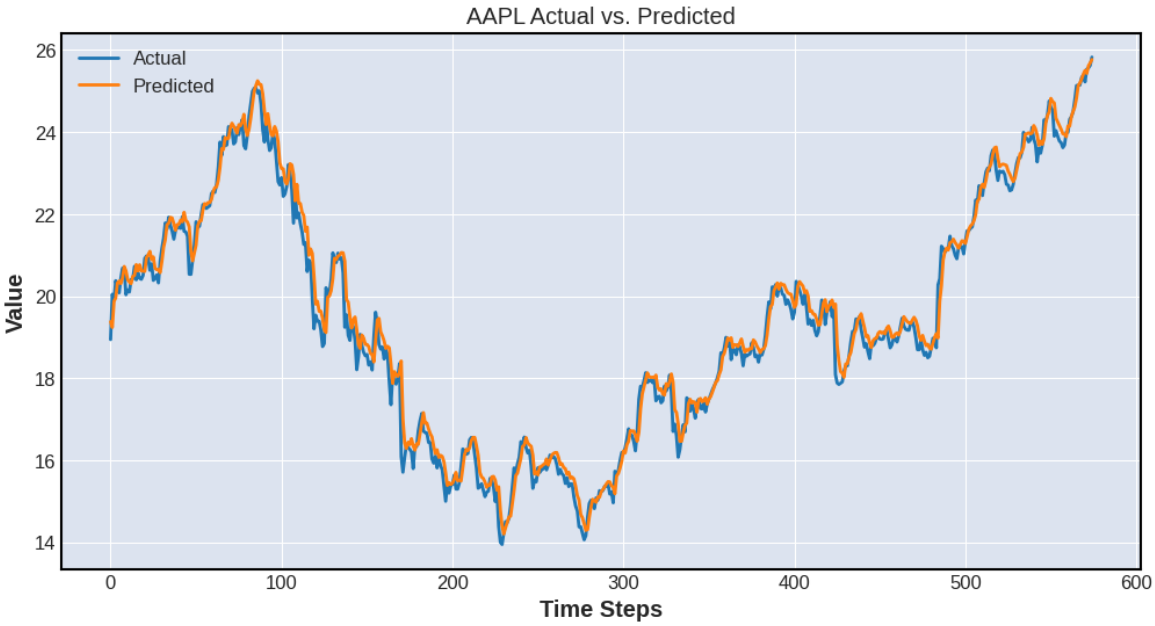
شکل ۱۰: نتایج مربوط به **Bi-LSTM**

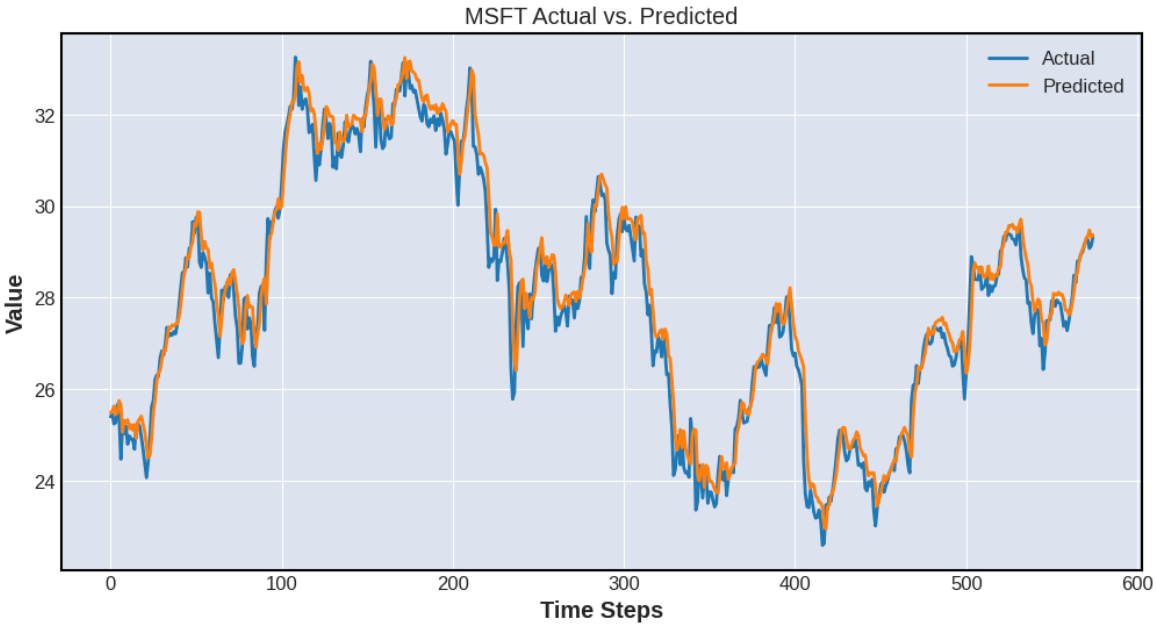


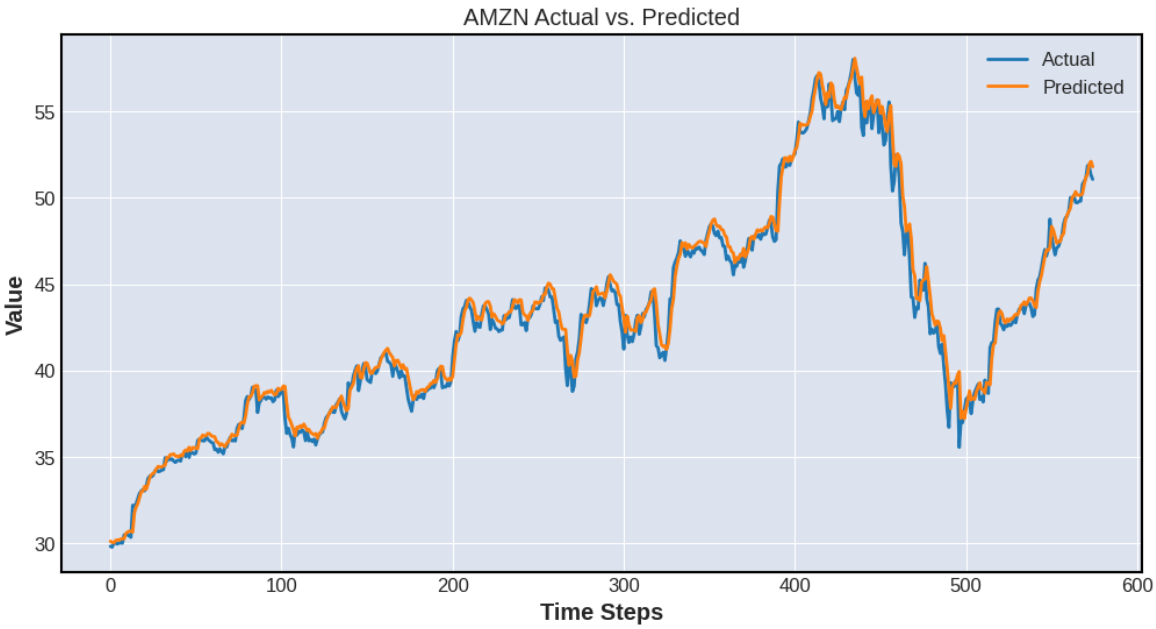
MPL

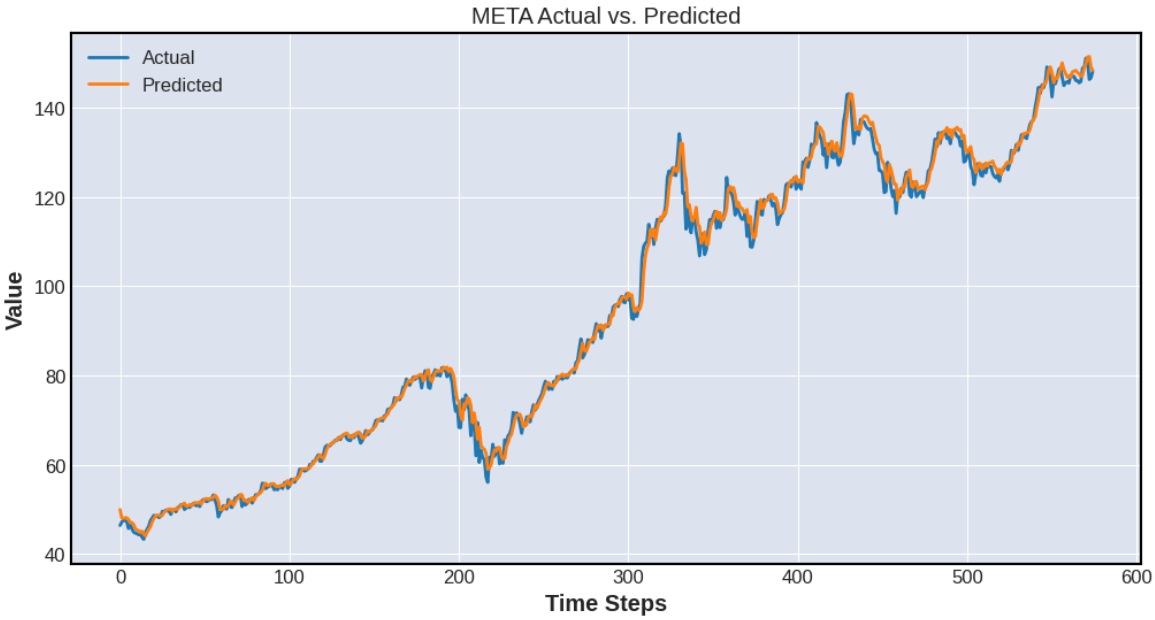
نتایج به صورت زیر خواهد بود:

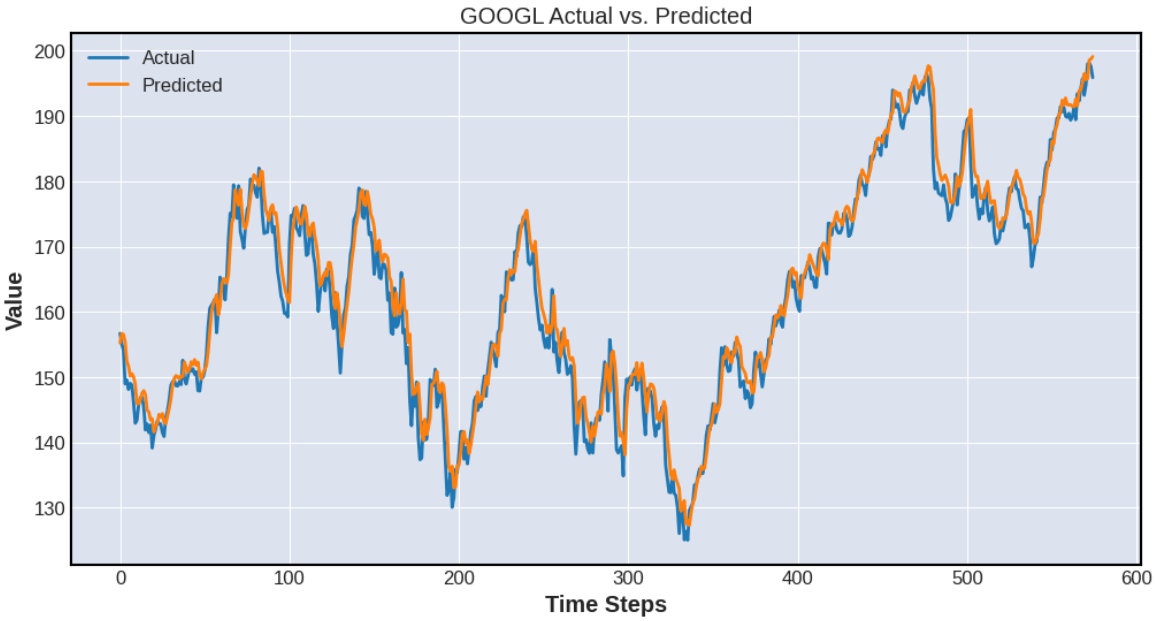
Mean of elements (element-wise): [۱۲.۹۹۲۸۸۰۰۶ ۱۲.۹۹۲۸۸۰۰۶ ۲.۱۴۹۸۰۴۱۶ ۲.۱۵۵۲۵۱۶۵]




MSFT Actual vs. Predicted
Actual
Predicted
Value
32
30
28
26
24
0
100
200
300
400
500
600
Time Steps


AMZN Actual vs. Predicted
Actual
Predicted
Value
55
50
45
40
35
30
0
100
200
300
400
500
600
Time Steps



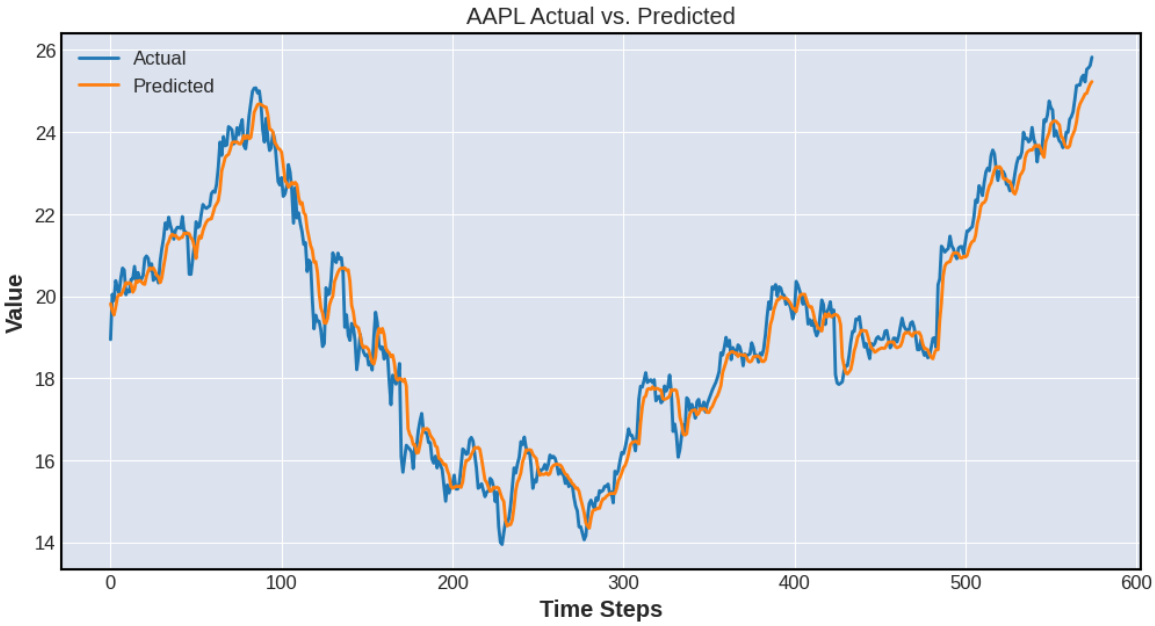


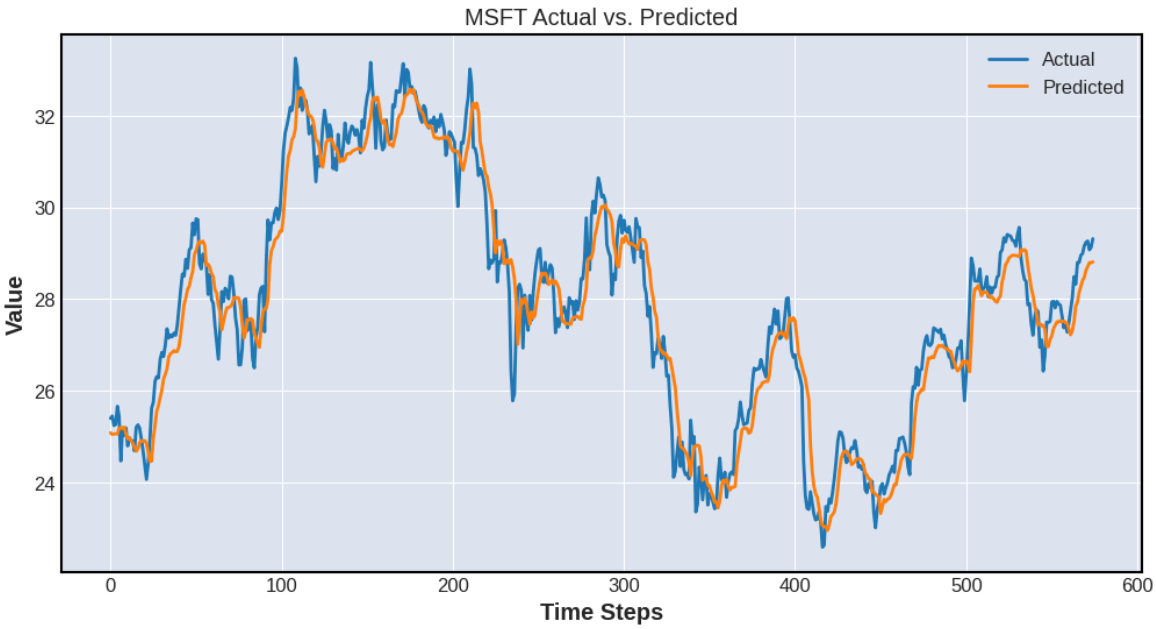
شکل ۱۱: نتایج **MLP**

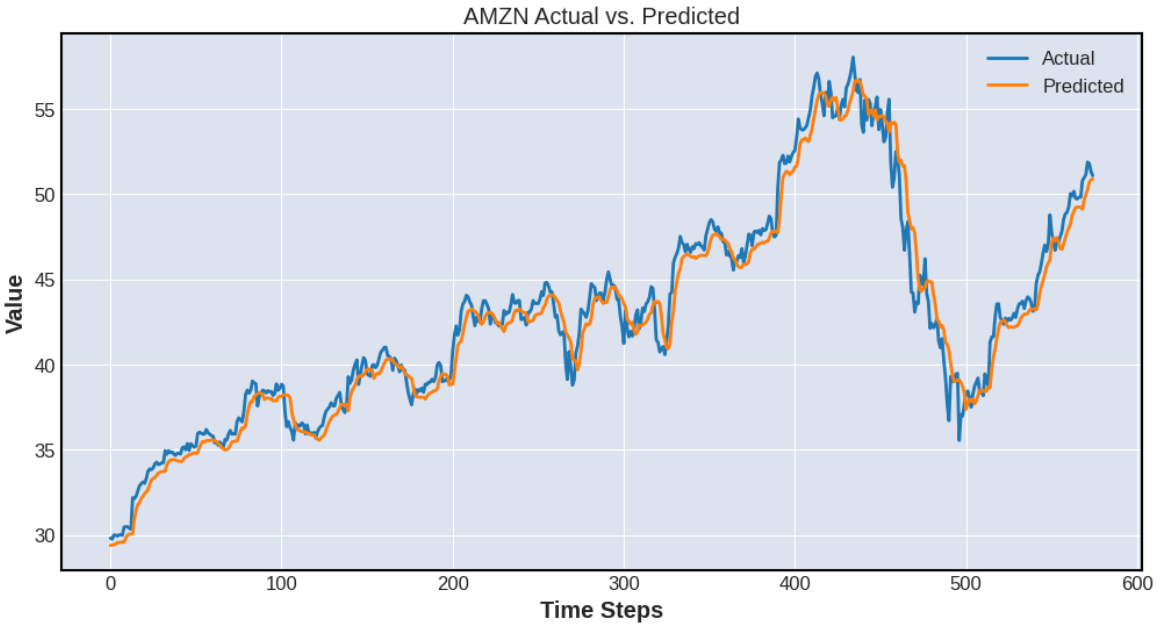
CNN

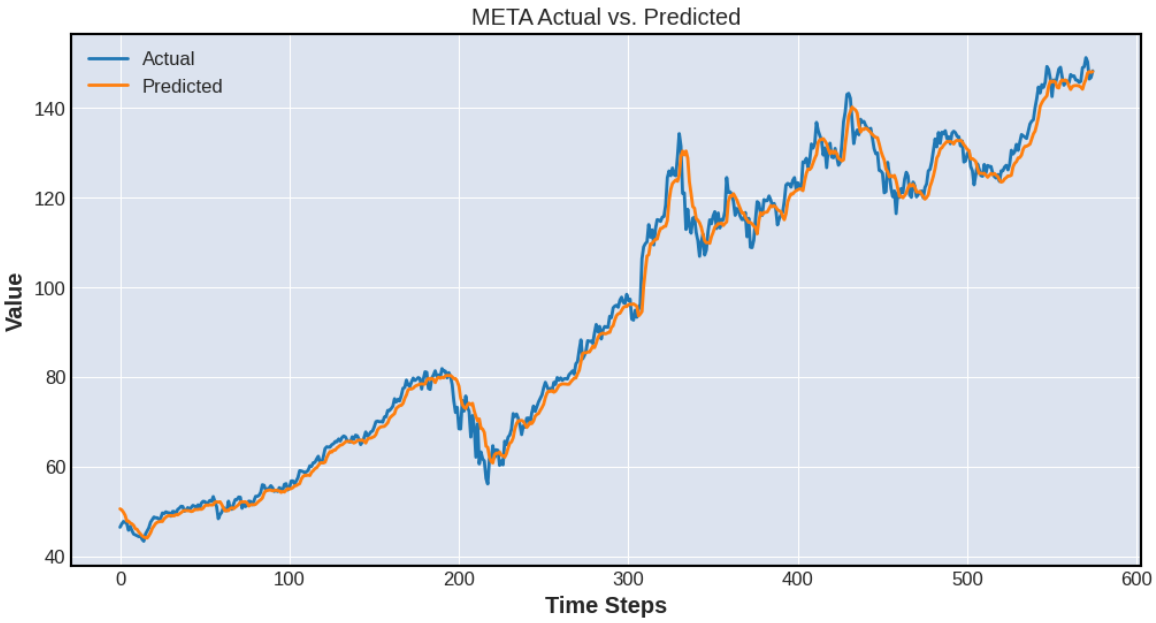
نتایج به صورت زیر خواهد بود:

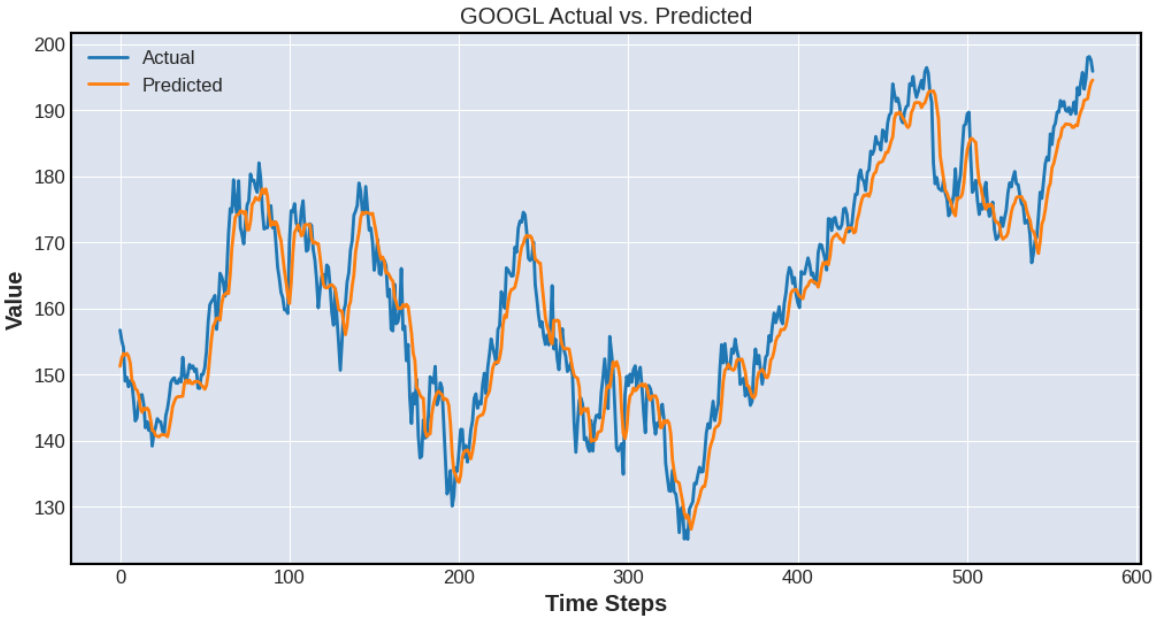
Mean of elements (element-wise): [۲۰.۷۰۴۷۱۹۶۴ ۲۰.۷۰۴۷۱۹۶۴ ۲.۵۶۷۹۲۹۰۵ ۲.۳۹۵۱۶۲۳۹]


AAPL Actual vs. Predicted
Actual
Predicted
Value
Time Steps
26
24
22
20
18
16
14
0
100
200
300
400
500
600


MSFT Actual vs. Predicted
Actual
Predicted
Value
Time Steps
32
30
28
26
24
0
100
200
300
400
500
600


AMZN Actual vs. Predicted
Actual
Predicted
Value
Time Steps
55
50
45
40
35
30
0
100
200
300
400
500
600


META Actual vs. Predicted
Actual
Predicted
Value
Time Steps
140
120
100
80
60
40
0
100
200
300
400
500
600

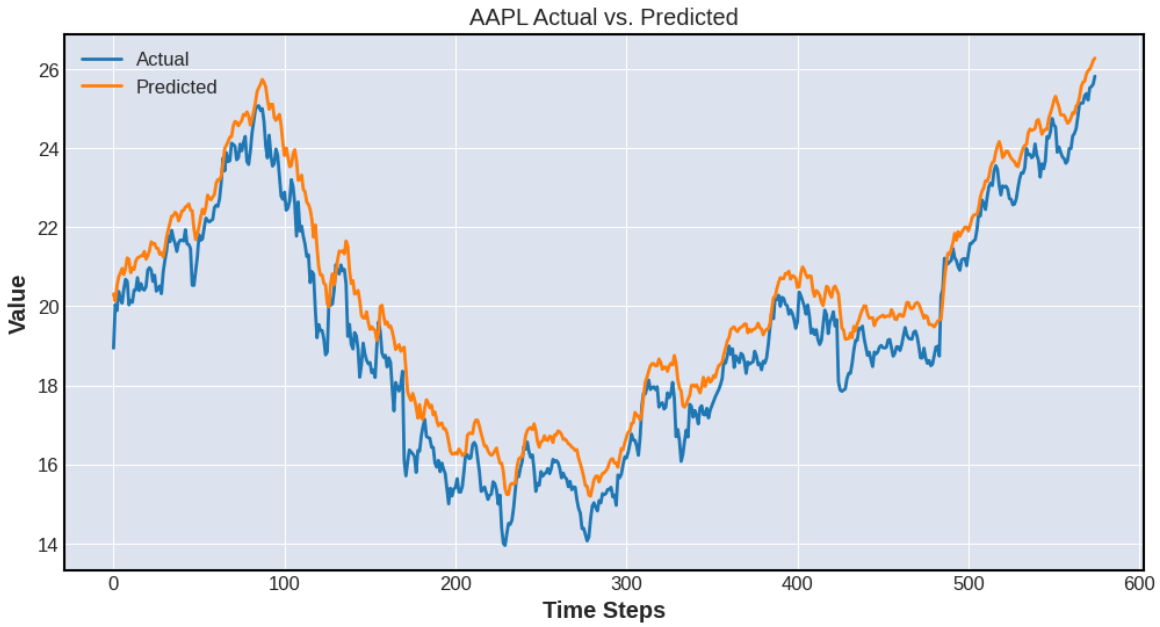


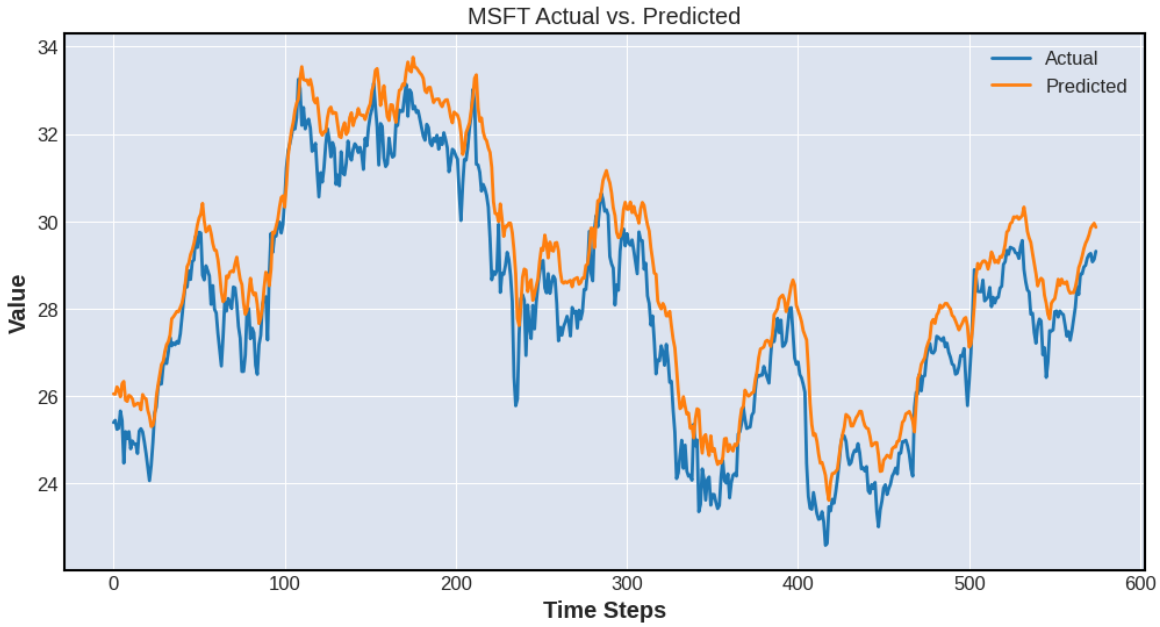
شکل ۱۲: نتایج **CNN**

CONV-LSTM

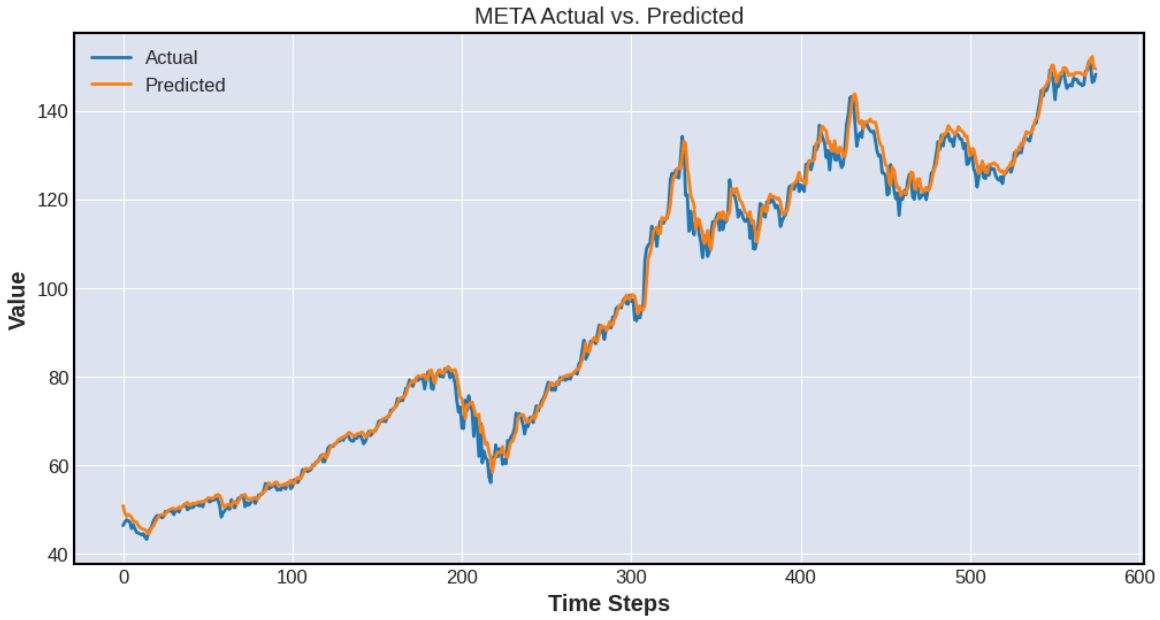
نتایج به صورت زیر خواهد بود:

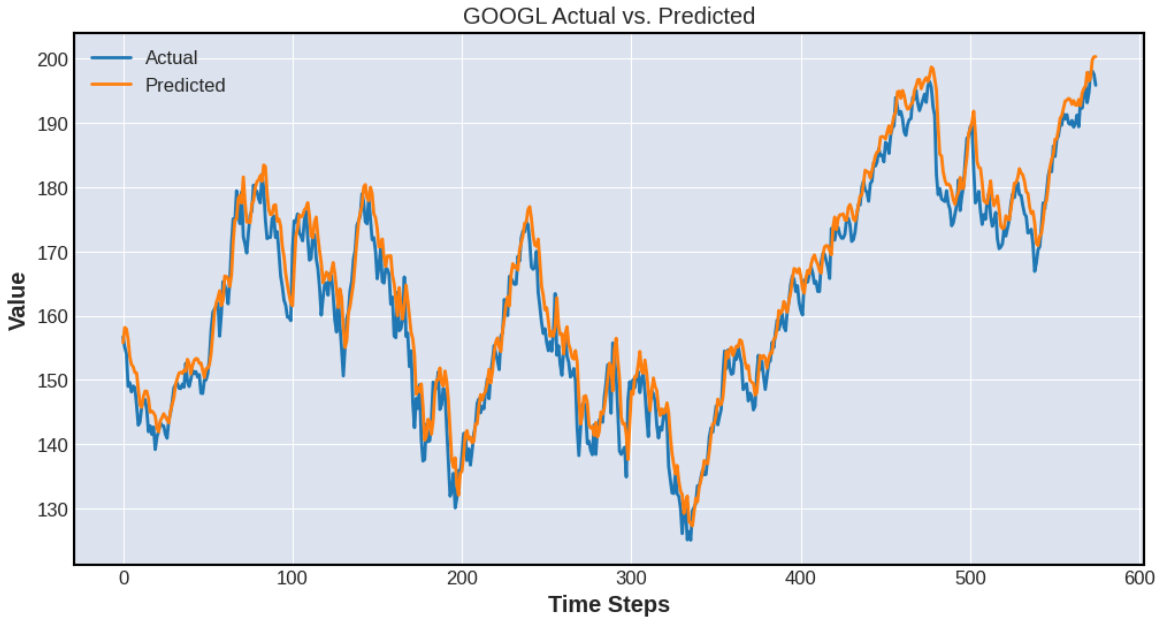
Mean of elements (element-wise): [۴۱.۰۸۴۴۰۳۶۶ ۴۱.۰۸۴۴۰۳۶۶ ۳.۲۹۵۴۲۸۱۱ ۳.۲۴۴۲۳۱۰۶]




MSFT Actual vs. Predicted
Actual
Predicted
Value
Time Steps
34
32
30
28
26
24
0
100
200
300
400
500
600


AMZN Actual vs. Predicted
Actual
Predicted
Value
Time Steps
55
50
45
40
35
30
0
100
200
300
400
500
600





شکل ۱۳: نتایج **CONV-LSTM**

حال تمام مراحل بالا را برای داده‌های نورمال شده نیز انجام می‌دهیم، از آنجا که شکل‌ها تغییرات زیادی نداشتند فقط مقادیر خطا را قرار می‌دهیم:

Mean of elements (element-wise): [۰.۰۱۲۶۶۹۵ ۰.۰۱۲۶۶۹۵ ۰.۰۵۹۳۰۷۵۵ ۳.۶۷۰۲۳۰۰۵]

Mean of elements (element-wise): [۰.۰۳۹۶۷۶۲۲ ۰.۰۳۹۶۷۶۲۲ ۰.۰۹۹۴۷۹۶ ۵.۷۶۰۱۱۵۸۱]

Mean of elements (element-wise): [۰.۰۲۷۲۰۱۹۳ ۰.۰۲۷۲۰۱۹۳ ۰.۰۸۴۷۱۱۴۷ ۴.۹۸۹۱۹۵۲۵]

Mean of elements (element-wise): [۹.۸۳۱۴۶۷۲۹e-۰۴ ۹.۸۳۱۴۶۷۲۹e-۰۴ ۲.۰۹۵۲۱۱۳۳e-۰۲ ۱.۸۱۹۲۴۳۸۴e+۰۰]

Mean of elements (element-wise): [۲.۸۶۴۳۰۷۹۶e-۰۳ ۲.۸۶۴۳۰۷۹۶e-۰۳ ۳.۶۸۹۵۲۹۱۹e-۰۲ ۲.۹۳۸۳۴۵۳۸e+۰۰]

Mean of elements (element-wise): [۰.۰۰۷۱۱۷۹۷ ۰.۰۰۷۱۱۷۹۷ ۰.۰۴۹۵۳۰۴۹ ۳.۲۴۳۴۳۹۱۳]

مشخص است که مدل‌ها سریعتر همگرا شده‌اند و نتایج به صورت کلی بهتر است.

میانگین مربعات خطا (MSE) معیاری است که مجذور خطاها را میانگین می‌گیرد، که در آن خطا تفاوت بین مقادیر واقعی و پیش‌بینی‌شده است. فرمول آن MSE = (۱/n) Σ (Yi - Ŷi)^۲ است که در آن Yi مقدار واقعی، Ŷi مقدار پیش بینی شده و n تعداد مشاهدات است. MSE بر خطاهای بزرگتر به دلیل تربیع خطاها قبل از میانگین گیری تأکید می کند، و آن را نسبت به مقادیر پرت حساس می کند. MSE پایین تر نشان دهنده تناسب مدل بهتر است.

میانگین خطای مطلق (MAE) میانگین خطاهای مطلق است، مشابه MSE در اندازه گیری تفاوت بین مقادیر واقعی و پیش بینی شده، اما بدون مربع کردن خطاها. فرمول MAE = (۱/n) Σ |Yi - Ŷi|، که در آن |Yi - Ŷi| خطای مطلق برای هر مشاهده است. MAE شهودی تر است و کمتر تحت تأثیر عوامل پرت نسبت به MSE است، و MAE پایین تر نشان دهنده تناسب مدل بهتر است.

میانگین درصد خطای مطلق (MAPE) اندازه خطا را بر حسب درصد کمیت می کند. این میانگین خطاهای مطلق تقسیم بر مقادیر واقعی است که به صورت درصد بیان می شود. فرمول MAPE = (۱/n)

Yi / (Yi - Ŷi)| Σ | است × ۱۰۰ درصد MAPE بینشی را در مورد دقت مدل در شرایط نسبی ارائه می دهد که برای درک بزرگی خطا در رابطه با مقیاس داده مفید است. با این حال، اگر مقادیر واقعی صفر یا نزدیک به صفر باشند، MAPE می تواند گمراه کننده یا تعریف نشده باشد.

داده‌های ارائه‌شده معیارهای عملکرد شش معماری شبکه عصبی مختلف را در یک پیش‌بینی رگرسیون یا سری زمانی نشان می‌دهند که احتمالاً مربوط به پیش‌بینی قیمت سهام است. سه معیار گزارش شده عبارتند از: میانگین مربعات خطا (MSE)، میانگین خطای مطلق (MAE) و میانگین درصد خطای مطلق (MAPE).

بر اساس MSE، مدل MLP (پرسپترون چند لایه) با کمترین امتیاز (۰.۱۴۰۶) بهترین عملکرد را دارد که نشان می‌دهد به طور میانگین مربعات خطاهای پیش‌بینی آن کوچک‌ترین هستند. Bi-LSTM (حافظه کوتاه مدت دو جهته) نیز عملکرد قوی با ۰.۱۷۶۵ MSE را نشان می دهد. از سوی دیگر، مدل GRU (واحد بازگشتی دروازه‌ای) دارای بالاترین ۹.۹۵۴۱) MSE) است که نشان می‌دهد پیش‌بینی‌های آن دقت کمتری دارند.

زمانی که MAE را در نظر می گیریم که میانگین مستقیم بزرگی خطا را ارائه می دهد، مدل MLP مجدداً از سایرین با کمترین ۰.۲۷۰۰ MAE بهتر عمل می کند و به دنبال آن مدل Bi-LSTM با MAE ۰.۳۲۵۳ قرار دارد. MAE بالاتر مدل GRU یعنی ۲.۹۲۸۶ نشان می دهد که پیش بینی های آن به طور متوسط دارای خطاهای مطلق بزرگ تری هستند.

برای MAPE، که خطاهای پیش‌بینی را به عنوان درصدی از مقادیر واقعی درک می‌کند، مدل MLP با کمترین خطا (۱.۴۳۲۶٪) پیشتاز است و از نزدیک با مدل ۱.۶۷۹۹٪) Bi-LSTM) دنبال می‌شود. مدل Conv-LSTM دارای بالاترین ۴.۳۴۰۲٪) MAPE) است، که نشان می دهد خطاهای پیش بینی آن نسبت به مقادیر واقعی بزرگترین هستند.

به طور کلی توقع داریم مدل‌هایی همچون conv-lstm بهتر عمل کند.

از آنجا که این مدل در بخش conv سعی می‌کند ویژگی‌های local را یادبگیرد و lstm سعی می‌کند ویژگی‌های global را یاد بگیرد. بنابراین این مدل توقع داریم بهتر عمل کند.

مدل های MLP و CNN از آنجا که اصولا برای این وظیفه نیستند باید بدترین عملکرد را داشته باشند.

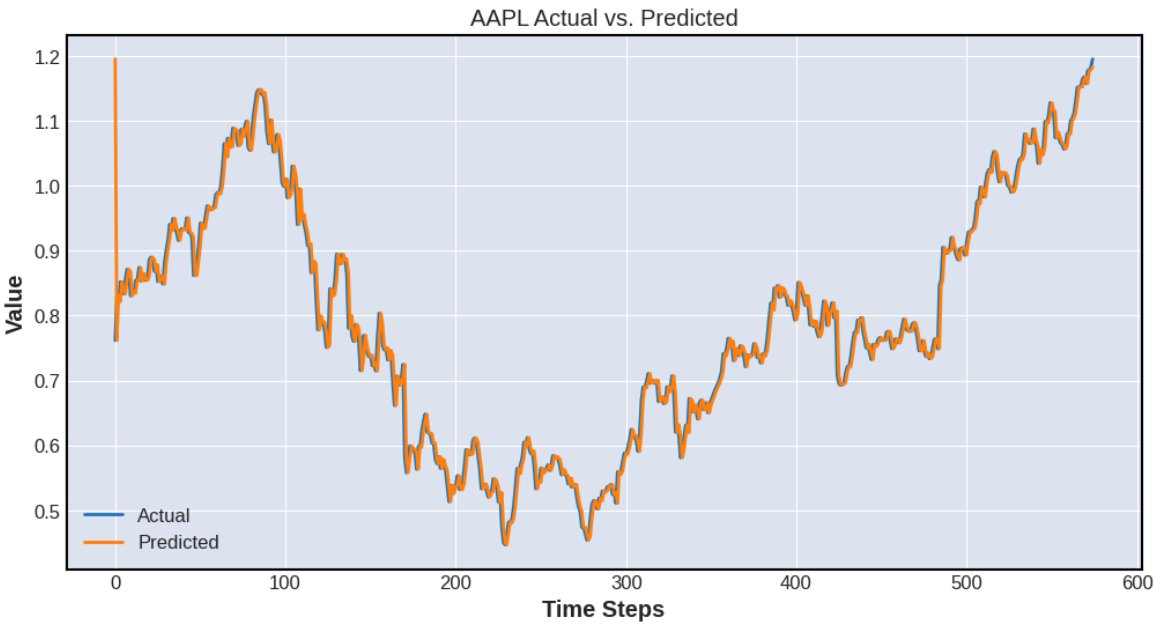
مدل BIlstm از انجا که به دو طرف اهمیت میدهد نسبت به مدل lstm بهتر عمل می‌کند زیرا چند روز آخر را بیشتر مورد توجه قرار می‌دهد.

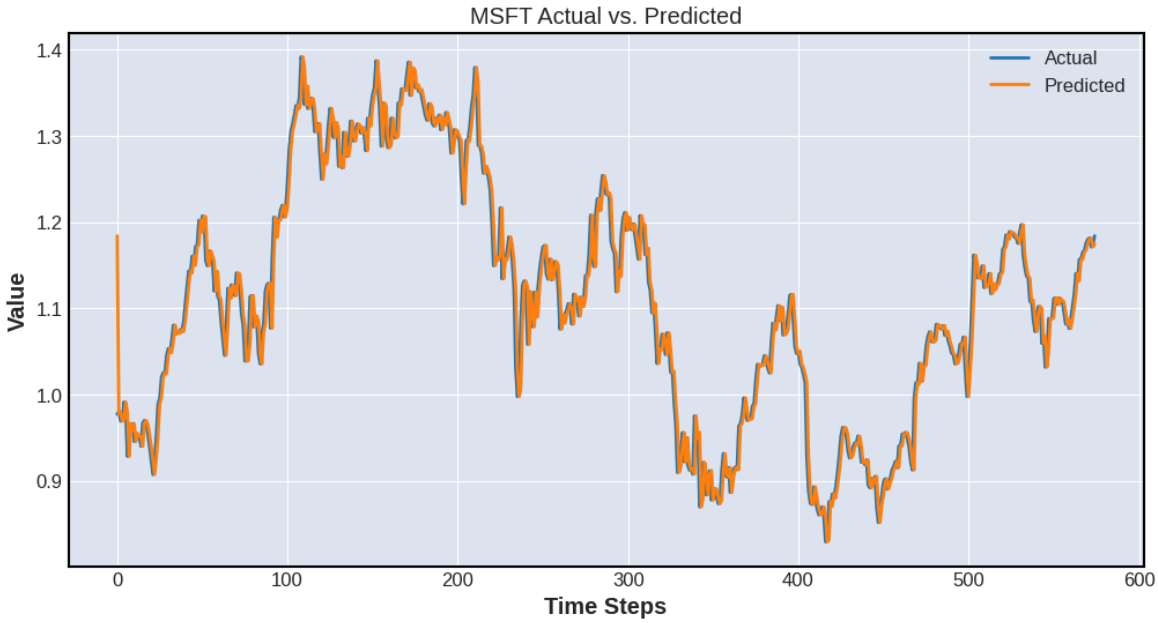
مدل gru کمی بهتر از مدل lstm عمل می‌کند

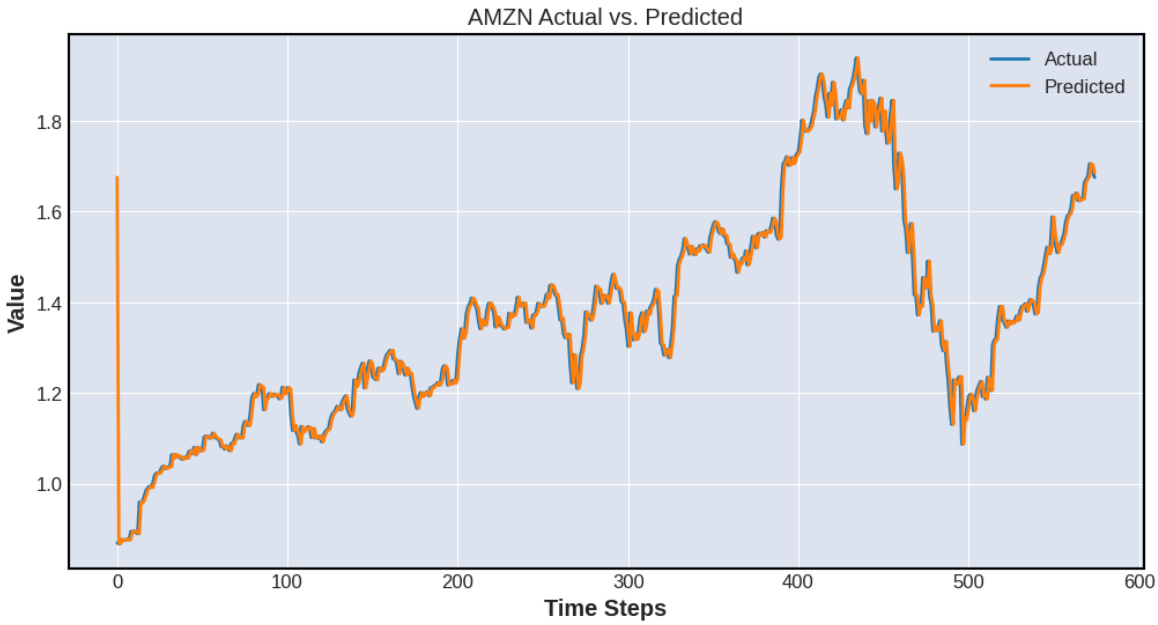
۶-۱. Naive Forecast

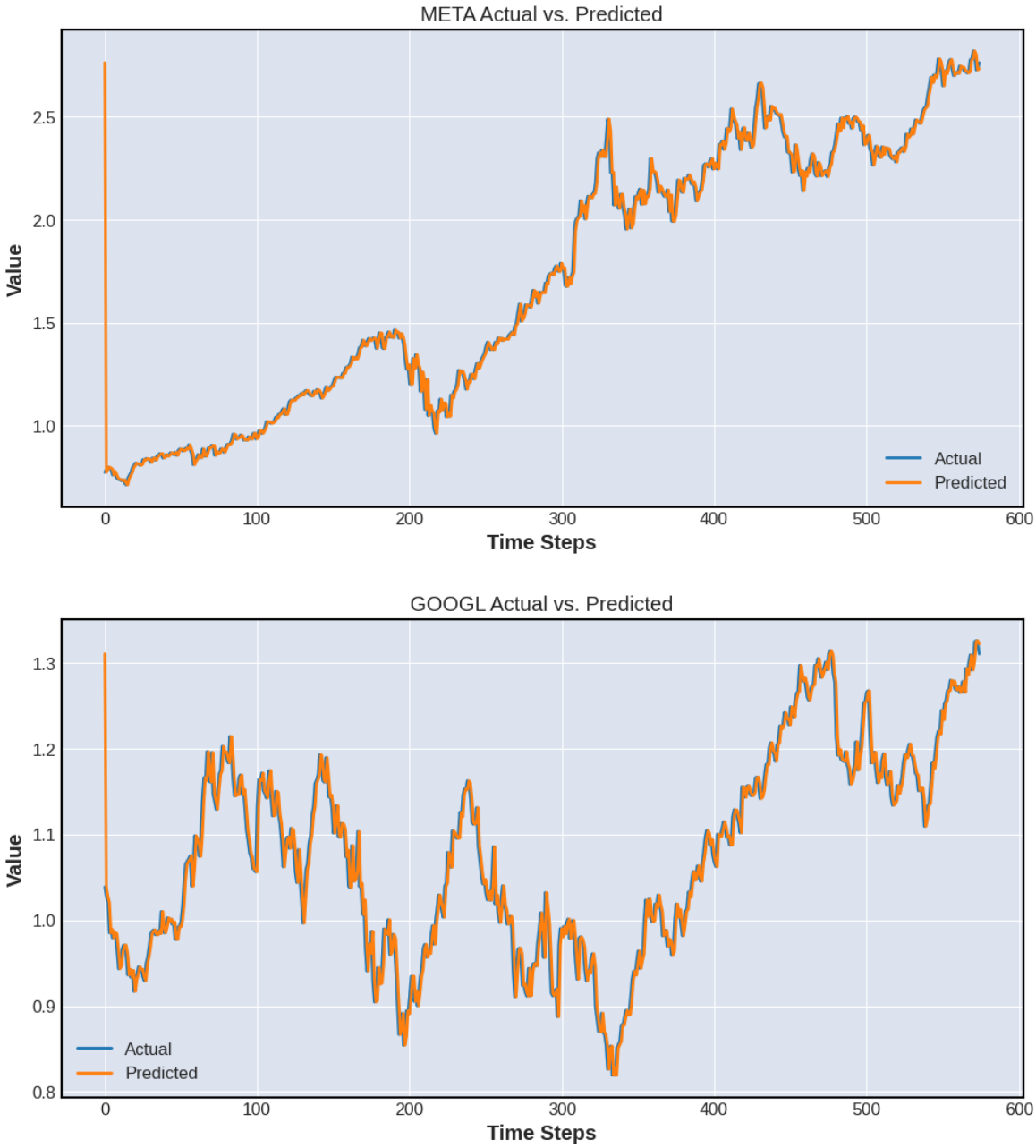
در این بخش آن چه ارائه شده برای ارزیابی عملکرد پیش‌بینی پایه در داده‌های سری زمانی برای فهرست نمادهای سهام («نماد») طراحی شده است. برای هر نماد سهام، به سادگی با جابجایی داده‌های واقعی آزمون («Y_test[i]») یک گام به جلو، پیش‌بینی‌هایی را ایجاد می‌کند. این رویکرد ساده لوحانه معمولاً به عنوان معیاری برای مدل های پیچیده تر استفاده می شود. این کد سه معیار ارزیابی را محاسبه می کند: میانگین مربعات خطا (MSE)، میانگین خطای مطلق (MAE) و میانگین درصد مطلق خطا (MAPE)، که دقت پیش بینی ها را در مقایسه با مقادیر واقعی اندازه گیری می کند. مقادیر پایین تر برای این معیارها نشان دهنده عملکرد پیش بینی بهتر است. پس از محاسبه این معیارها، کد مقادیر واقعی و پیش‌بینی‌شده را در یک نمودار خطی با استفاده از «matplotlib» تجسم می‌کند و یک مقایسه بصری از عملکرد مدل ارائه می‌کند. این نمایش گرافیکی به ارزیابی سریع میزان همسویی پیش‌بینی‌های ساده (مقادیر تغییر یافته) با قیمت‌های واقعی سهام در طی مراحل زمانی معین کمک می‌کند و معیارهای چاپ شده ارزیابی کمی را ارائه می‌دهند.

نتایج به صورت زیر خواهد بود:




MSFT Actual vs. Predicted
Actual
Predicted
Value
1.4
1.3
1.2
1.1
1.0
0.9
0
100
200
300
400
500
600
Time Steps


AMZN Actual vs. Predicted
Actual
Predicted
Value
1.8
1.6
1.4
1.2
1.0
0
100
200
300
400
500
600
Time Steps



شکل ۱۴: نتایج **naive forcast**

mse: ۰.۰۰۰۷۳۰ - mae: ۰.۰۱۴۹۴۷ - mape: ۰.۰۲٪

با توجه به آن چه به دست آوردیم این رویکرد بهترین عملکرد را دارد. البته این دیتاها مربوط به کمپانی‌های معروف است و لزوما به معنی بهترین عملکرد برای این مدل نیست.

پاسخ ۲ – پیشبینی افکار خودکشی در رسانه های اجتماعی

هدف از این تمرین تشخیص افکار خودکشی از مجموعه دادههای توییتر است. در مقاله چندین روش یادگیری ماشین و یادگیری عمیق مورد بررسی قرار گرفته است. با نگاه اجمالی به نتایج این مقاله میتوان دریافت که توانایی مدلهای مبتنی بر یادگیری عمیق در تشخیص افکار خودکشی بیشتر است. بوده به بیان دیگر نتایج این مقاله توانایی مدلهای مبتنی بر یادگیری عمیق در پردازش متن را نشان میدهد. مجموعه داده ای جهت انجام این تمرین پیوست شده است که شامل متن توییت و برچسب

خودکشی میباشد.

از بین مدلهای بررسی شده درمقاله شما میبایست مدلهای LSTM-۲ layer LSTM و layer LSTM+CNN را برای تشخیص افکار خودکشی بررسی کنید از آنجایی که برخی مولفه ها مانند نرخ آموزش، تعداد نورونهای لایه ها و ..... در مقاله مشخص نشده است، شما میتوانید از مقادیر معقول برای این موارد استفاده کنید.

۲-۱ پیش پردازش داده

در ابتدا شما لازم است تمامی پیش پردازشهای گفته شده در مقاله مانند ریشه یابی، حذف کلمات کم ارزش حذف پسوند حذف علائم نگارشی و را روی داده انجام دهید برای مثال متن پس از پیش پردازش داده متن شماره یک به متنی مانند شماره دو تبدیل شود.

متن شماره یک

my life is meaningless i just want to end my life so badly my life is completely empty and i dont want to have to create meaning in it creating meaning is pain how long will i hold back the urge to run my car head first into the next person coming the opposite way when will I stop feeling jealous

of tragic characters like gomer pile for the swift end they were able to bring to their lives

متن شماره دو

life meaningless want end life badly life completely empty dont want create meaning creating meaning pain long hold back urge run car head first next person coming opposite way stop feeling jealous tragic character like gomer pile swift end able bring life

۲-۲. ساخت ماتریس جاسازی

در این بخش همان طور که در مقاله ذکر شده لازم است از مدل از پیش آموزش دیده شده word۲vec ماتریس جاسازی را بسازید. دلیل استفاده و ویژگیهای این ماتریس را در گزارش به صورت مختصر توضیح دهید.

۲-۳ آموزش مدلهای یادگیری عمیق

در این بخش باید سه مدل ۲-layer LSTM LSTM و ۲-layer LSTM + CNN را با استفاده از داده های پیش پردازش شده آموزش دهید. (نکته) ممکن است در طراحی مدل ۲-layer LSTM + CNN با خطایی مواجه شوید که ورودی لایه LSTM باید سه بعدی ،باشد برای رفع این خطا میتوانید از لایه Reshape استفاده کنید. البته راه حلهای دیگری نیز وجود دارد.

۲-۴. مقایسه نتایج

در این بخش نتایج سه مدل آموزش داده شده را مقایسه کنید و استدلال خود را جهت توجیه نتایج درگزارش بنویسید.

۲-۱ پیش پردازش داده

پیش پردازش داده‌ها، فرایندی فنی باهدف بهبود مجموعه‌داده‌ها برای قابل‌استفاده شدن آن‌هاست. این فرایند شامل تغییر و گاهی حذف داده‌های ناکامل(Incomplete) ، دارای فرمت نادرست، نامربوط و

تکراری (Duplicate) است. همچنین گاهی نیاز به تبدیل داده متنی به مقادیر عددی و بازطراحی ویژگی‌ها می‌شود. در یک پروژه تحلیل داده، معمولاً پیش پردازش داده ها زمان‌برترین قسمت است.

حال در این بخش به صورت زیر عمل می کنیم:

۱. کوچک کردن حروف: `Suicide['lower_case'] = Suicide['Tweet'].apply(lambda x: x.lower())`

- این خط تمام حروف موجود در ستون `Tweet` از داده‌های `Suicide` را به حروف کوچک تبدیل می‌کند.

۲. حذف عبارت RT (ریتویت): کد دوم که کامنت شده، هدفش حذف کلمه "rt" از متون است، اما به نظر می‌رسد که اشتباهاً از متغیر `Dataset` استفاده کرده است که در این قطعه کد وجود ندارد.

۳. توکنایز کردن: `Suicide['Special_word'] = Suicide.apply(lambda row: tokenizer.tokenize(row['lower_case']), axis=۱)`

- این خط، متن کوچک شده را به توکن‌ها (کلمات) تقسیم می‌کند.

۴. یافتن کلمات کم‌فراوانی:

- این بخش کلماتی را که کمترین فراوانی را در مجموعه داده دارند، پیدا می‌کند و آن‌ها را از متن حذف می‌کند.

۵. حذف کلمات توقف:

- `Suicide['stop_words'] = Suicide['Special_word'].apply(lambda x: [item for item in x if item not in stop])`

- این خط کلمات توقف (کلماتی که معمولاً در تحلیل متن نادیده گرفته می‌شوند) را از لیست توکن‌ها حذف می‌کند.

۶. انتخاب کلمات با طول حداقل سه حرف:

- `Suicide['short_word'] = Suicide['stop_words'].str.findall('w{۳,}')`

- فقط کلماتی که حداقل سه حرف دارند، انتخاب می‌شوند.

۷. تبدیل به رشته متنی:

- این بخش توکن‌ها را دوباره به یک رشته متنی تبدیل می‌کند و علائم نگارشی را حذف می‌کند.

۸. حذف کلمات غیرانگلیسی:

- `Suicide['NonEnglish'] = Suicide['string'].apply(lambda x: " ".join(x for x in x.split() if x in words))`

- این خط تنها کلماتی که در لغت‌نامه انگلیسی وجود دارند را نگه می‌دارد.

۹. لماتیزیشن:

- `Suicide['tweet'] = Suicide['NonEnglish'].apply(lambda x: " ".join([Word(word).lemmatize() for word in x.split()]))`

- کلمات به شکل اصلی خود (لماتیزیشن) تبدیل می‌شوند.

۱۰. نمایش داده‌ها:

- `Suicide.head()`

- این دستور پنج رکورد اول از دیتاست پردازش شده را نمایش می‌دهد.

این فرآیند در کل به تمیز کردن و تجزیه و تحلیل داده‌های متنی مربوط به توییت‌ها کمک می‌کند.

	Tweet	Suicide	lower_case	Special_word	Contents	stop_words	short_word	string	NonEnglish	tweet
0	my life is meaningless i just want to end my l...	1	my life is meaningless i just want to end my l...	[my, life, is, meaningless, i, just, want, to,...	my life is meaningless i just want to end my l...	['life', 'meaningless', 'want', 'end', 'life',...	[life, meaningless, want, end, life, badly, li...	life meaningless want end life badly life comp...	life meaningless want end life badly life comp...	life meaningless want end life badly life comp...
1	muttering i wanna die to myself daily for a fe...	1	muttering i wanna die to myself daily for a fe...	[muttering, i, wanna, die, to, myself, daily, ...	muttering i wanna die to myself daily for a fe...	['muttering', 'wanna', 'die', 'daily', 'months...	[muttering, wanna, die, daily, months, feel, w...	muttering wanna die daily months feel worthles...	muttering die daily feel worthless cant live h...	muttering die daily feel worthless cant live h...
2	work slave i really feel like my only purpose ...	1	work slave i really feel like my only purpose ...	[work, slave, i, really, feel, like, my, only,...	work slave i really feel like my only purpose ...	['work', 'slave', 'really', 'feel', 'like', 'p...	[work, slave, really, feel, like, purpose, lif...	work slave really feel like purpose life make ...	work slave really feel like purpose life make ...	work slave really feel like purpose life make ...
3	i did something on the 2 of october i overdose...	1	i did something on the 2 of october i overdose...	[i, did, something, on, the, 2, of, october, i...	i did something on the 2 of october i overdose...	['something', '2', 'october', 'overdosed', 'fe...	[something, october, overdosed, felt, alone, h...	something october overdosed felt alone horribl...	something felt alone horrible hospital two day...	something felt alone horrible hospital two day...
4	i feel like no one cares i just want to die ma...	1	i feel like no one cares i just want to die ma...	[i, feel, like, no, one, cares, i, just, want,...	i feel like no one cares i just want to die ma...	['feel', 'like', 'one', 'cares', 'want', 'die'...	[feel, like, one, cares, want, die, maybe, fee...	feel like one cares want die maybe feel less l...	feel like one want die maybe feel less lonely	feel like one want die maybe feel le lonely

شکل ۱۵: نحوه عملکرد پیش پردازش بر روی داده ها

۲-۲. ساخت ماتریس جاسازی

ماتریس جاسازی یک مفهوم کلیدی در پردازش زبان طبیعی (NLP) و یادگیری ماشینی است، به ویژه در کارهای مربوط به داده های متنی. برای تبدیل داده های متنی به شکل عددی استفاده می شود تا بتوان آنها را توسط الگوریتم های یادگیری ماشین پردازش کرد. بیایید هدف آن و نحوه استفاده از آن در زمینه مورد شما را بشکافیم:

ماتریس جاسازی چیست؟

نمایش کلمات به عنوان بردار: در NLP، کلمات یا عبارات از واژگان به بردارهای اعداد واقعی نگاشت می شوند. این بردارها در یک ماتریس جاسازی شده ذخیره می شوند. هر ردیف از ماتریس مربوط به یک بردار است که یک کلمه خاص در واژگان را نشان می دهد.

گرفتن اطلاعات معنایی: هدف این بردارها گرفتن معنای معنایی کلمات است. کلماتی با معانی مشابه معمولاً در این فضای پربعد نزدیک به هم قرار می گیرند. برای مثال، «شاد» و «شاد» ممکن است بردارهای مشابهی داشته باشند.

کاهش ابعاد: کلمات ذاتاً ابعاد بالایی دارند (زیرا واژگان می‌توانند گسترده باشند)، اما جاسازی‌ها آنها را به یک فضای پیوسته با ابعاد پایین‌تر ترسیم می‌کنند، که محاسبات را امکان‌پذیرتر می‌کند و روابط معنایی را به شکل متراکم‌تری به تصویر می‌کشد.

استفاده ماتریس جاسازی

در سناریوی ما، جایی که در حال تجزیه و تحلیل توییت‌های مربوط به «خودکشی» هستید، ماتریس جاسازی چندین هدف را دنبال می‌کند:

تبدیل متن به داده های عددی: استفاده اولیه از ماتریس جاسازی، تبدیل داده های متنی توییت ها به یک قالب عددی است که می تواند توسط مدل های یادگیری ماشین پردازش شود.

استخراج ویژگی برای یادگیری ماشینی: بردارهای موجود در ماتریس تعبیه شده را می توان به عنوان ویژگی برای وظایف مختلف یادگیری ماشین مانند تجزیه و تحلیل احساسات، مدل سازی موضوع یا خوشه بندی استفاده کرد. این وظایف می تواند به درک مضامین یا احساسات اساسی در توییت های مربوط به خودکشی کمک کند.

تحلیل معنایی: با استفاده از embedding ها می توانید محتوای معنایی توییت ها را تحلیل کنید. این می تواند در درک زمینه و تفاوت های ظریف در بحث های مربوط به خودکشی، که اغلب حساس و پیچیده هستند، بسیار مهم باشد.

ورودی برای مدل‌سازی بیشتر: اگر در حال ساخت مدل‌ها یا طبقه‌بندی‌کننده‌های پیش‌بینی‌کننده هستید (مثلاً برای شناسایی توییت‌های پرخطر)، ماتریس تعبیه‌شده یک ورودی ساختار یافته و عددی را ارائه می‌دهد که این مدل‌ها به آن نیاز دارند.

دلیل استفاده از ویژگی های ماتریس جاسازی شده

دلایل استفاده از ویژگی های یک ماتریس تعبیه شده عبارتند از:

غنای معنایی: تعبیه‌ها غنای معنایی کلمات را که برای درک زمینه و احساسات در داده‌های متن ضروری است، نشان می‌دهد.

کاهش ابعاد: آنها ابعاد بالای داده های متنی را کاهش می دهند و وظایف محاسباتی را کارآمدتر و قابل مدیریت می کنند.

عملکرد مدل بهبود یافته: مدل‌هایی که با جاسازی‌ها آموزش داده شده‌اند، اغلب بهتر از مدل‌هایی هستند که با اشکال ابتدایی‌تر بازنمایی متن مانند مجموعه کلمات آموزش داده شده‌اند، زیرا اطلاعات بیشتری در مورد استفاده از کلمه دریافت می‌کنند.

به طور خلاصه، یک ماتریس جاسازی، داده‌های متنی را به شکلی تبدیل می‌کند که می‌تواند به طور موثر توسط مدل‌های مختلف یادگیری ماشین استفاده شود، در نتیجه امکان تجزیه و تحلیل دقیق‌تر و معنادارتر داده‌های متن را فراهم می‌کند، که به‌ویژه در حوزه‌های حساسی مانند تحلیل سلامت روان ارزشمند است.

```
print(embedding_matrix)
```

```
[[-0.11348409  0.29833013 -0.0004761  ... -0.3151035   0.12399767
  -0.10693783]
 [-0.12075401  0.34691364  0.05157819 ... -0.37699366  0.16010079
  -0.12371175]
 [-0.08178905  0.20713386  0.04869472 ... -0.43026084  0.10177354
  -0.0808231 ]
 ...
 [-0.15933561  0.151096    0.02585114 ... -0.18235224  0.07893433
  -0.14239125]
 [-0.09257533  0.20135275  0.0613501  ... -0.43641821  0.1754344
  -0.15923612]
 [-0.05803766  0.18383066 -0.01557013 ... -0.34222192  0.08892867
  -0.13691851]]
```

شکل ۱۶: ماتریس جاسازی

۲-۳ آموزش مدلهای یادگیری عمیق

۱-layer LSTM

این کد یک مدل شبکه عصبی بازگشتی (RNN) با استفاده از کتابخانه‌های TensorFlow و Keras برای کار بر روی داده‌های متنی می‌سازد. این مدل معمولاً برای وظایفی مانند تحلیل احساسات یا طبقه‌بندی متن استفاده می‌شود. در ادامه هر قسمت از کد توضیح داده شده است:

۱. وارد کردن کتابخانه‌ها:

- کتابخانه‌های TensorFlow و Keras برای ساخت مدل‌های یادگیری عمیق وارد می‌شوند.

۲. ایجاد مدل:

- یک مدل `Sequential` ساخته می‌شود که به این معنی است که لایه‌های مدل به ترتیب اضافه خواهند شد.

۳. لایه Embedding:

- این لایه برای تبدیل اندیس‌های کلمات به بردارهای کثیف استفاده می‌شود. `n_most_common_words` تعداد کلمات در واژگان و `emb_dim` بعد بردار کلمات است.

۴. لایه SpatialDropout۱D:

- این لایه از overfitting جلوگیری می‌کند با غیرفعال کردن بخشی از نورون‌ها (در اینجا ۷۰٪).

۵. لایه LSTM:

- LSTM یک نوع شبکه عصبی بازگشتی است که برای داده‌های دنباله‌ای مانند متن مناسب است. `۲۰۰` تعداد نورون‌ها در این لایه است.

۶. لایه Dense:

- این لایه برای خروجی نهایی مدل است. `۲` نشان‌دهنده تعداد خروجی‌ها (برای طبقه‌بندی دو کلاسه) و ``activation='softmax'` احتمالات خروجی را محاسبه می‌کند.

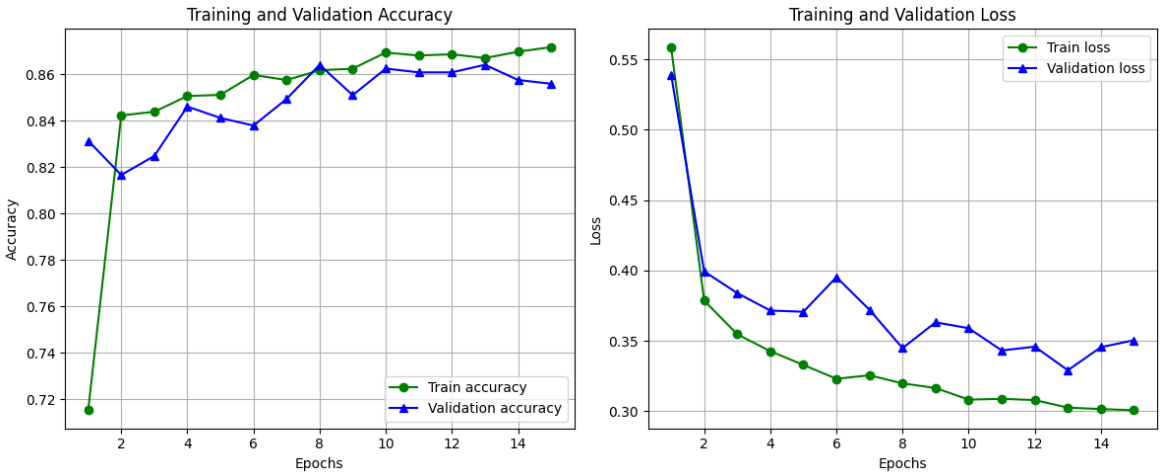
۷. کامپایل کردن مدل:

- مدل با بهینه‌ساز `Adam` و تابع هزینه `binary_crossentropy` کامپایل می‌شود. `metrics=['acc']` دقت مدل را به عنوان یک معیار ارزیابی می‌گیرد.

۸. آموزش مدل:

- مدل با استفاده از داده‌های آموزشی `XX_train` و `y_train` آموزش داده می‌شود. تعداد دوره‌ها (epoch) برابر با `۱۵` و اندازه دسته‌بندی (batch size) برابر با `۵۰۰` است. همچنین ۱۰٪ از داده‌ها برای ارزیابی در هر دوره استفاده می‌شوند.

نتایج مربوط به این مدل به صورت زیر به دست آمده است:



شکل ۱۷: **loss** و دقت مدل **lstm** یک لایه

۲-Layer LSTM

این کد یک مدل شبکه عصبی بازگشتی (RNN) با استفاده از کتابخانه‌های TensorFlow و Keras برای کار بر روی داده‌های متنی می‌سازد. در ادامه، هر بخش از کد به زبان فارسی توضیح داده شده است:

۱. وارد کردن کتابخانه‌ها:

- کتابخانه‌های لازم برای ساخت مدل، از جمله لایه‌های `Embedding`, `Dense`, و `LSTM` وارد می‌شوند.

۲. ایجاد مدل:

- یک مدل `Sequential` ساخته می‌شود که به این معنی است که لایه‌های مدل به ترتیب و به صورت خطی اضافه خواهند شد.

۳. لایه Embedding:

- این لایه برای تبدیل اندیس‌های کلمات به بردارهای کثیف استفاده می‌شود. `n_most_common_words` تعداد کلمات در واژگان و `emb_dim` بعد بردار کلمات است.

۴. لایه‌های LSTM:

- دو لایه LSTM اضافه شده‌اند. اولین لایه `۵۰` نورون دارد و دومین لایه `۲۰۰` نورون. همچنین تنظیمات `dropout` و `recurrent_dropout` برای جلوگیری از overfitting استفاده می‌شوند.

۵. لایه Dense:

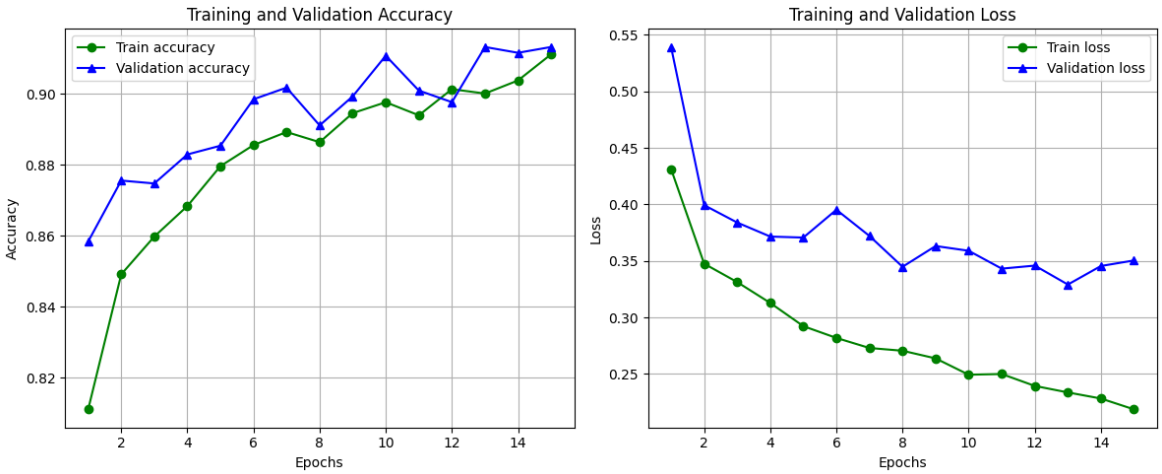
- این لایه برای خروجی نهایی مدل است. `۲` نشان‌دهنده تعداد خروجی‌ها (برای طبقه‌بندی دو کلاسه) و `activation='softmax'` احتمالات خروجی را محاسبه می‌کند.

۶. کامپایل کردن مدل:

- مدل با بهینه‌ساز `Adam` و تابع هزینه `binary_crossentropy` کامپایل می‌شود. `metrics=['acc']` دقت مدل را به عنوان یک معیار ارزیابی می‌گیرد.

۷. آموزش مدل:

- مدل با استفاده از داده‌های آموزشی `XX_train` و `y_train` آموزش داده می‌شود. تعداد دوره‌ها (epoch) برابر با `۱۵` و اندازه دسته‌بندی (batch size) برابر با `۲۰۰` است. همچنین ۲۰٪ از داده‌ها برای ارزیابی در هر دوره استفاده می‌شوند.



شکل ۱۸: **s** مدل **lstm** دو لایه

CNN + LSTM

این کد یک مدل شبکه عصبی ترکیبی از لایه‌های کانولوشنی (Convolutional) و بازگشتی (Recurrent) با استفاده از کتابخانه‌های TensorFlow و Keras برای پردازش داده‌های متنی می‌سازد. این مدل برای مواردی مانند تحلیل احساسات یا طبقه‌بندی متن استفاده می‌شود. در ادامه هر بخش از کد توضیح داده شده است:

۱. وارد کردن کتابخانه‌ها:

- لایه‌های مورد نیاز برای ساخت مدل، از جمله `Conv۱D`, `MaxPooling۱D`, `Dense`, `Flatten`, `Dropout`, و `Embedding` وارد می‌شوند.

۲. ایجاد مدل:

- یک مدل `Sequential` ساخته می‌شود، که نشان دهنده ترتیب خطی لایه‌ها در مدل است.

۳. لایه Embedding:

- این لایه برای تبدیل اندیس‌های کلمات به بردارهای کثیف استفاده می‌شود. `n_most_common_words` تعداد کلمات در واژگان و `emb_dim` بعد بردار کلمات است.

۴. لایه‌های Conv۱D و MaxPooling۱D:

- لایه `Conv۱D` برای استخراج ویژگی‌ها از داده‌های متنی استفاده می‌شود. `۳۰۰` تعداد فیلترها و `۶` اندازه هسته کانولوشن است.

- لایه `MaxPooling۱D` برای کاهش اندازه نمایه‌های استخراج شده به کار می‌رود.

۵. لایه Dropout:

- برای جلوگیری از overfitting با غیرفعال کردن تصادفی نورون‌ها (در اینجا ۶۰٪).

۶. لایه‌های LSTM:

- دو لایه LSTM برای پردازش داده‌های دنباله‌ای مانند متن استفاده می‌شوند. اولین لایه `۱۰۰` و دومین لایه `۵۰` نورون دارند.

۷. لایه Dense:

- برای خروجی نهایی مدل. `۲` نشان‌دهنده تعداد خروجی‌ها (برای طبقه‌بندی دو کلاسه) و `activation='softmax'` برای محاسبه احتمالات است.

۸. کامپایل کردن مدل:

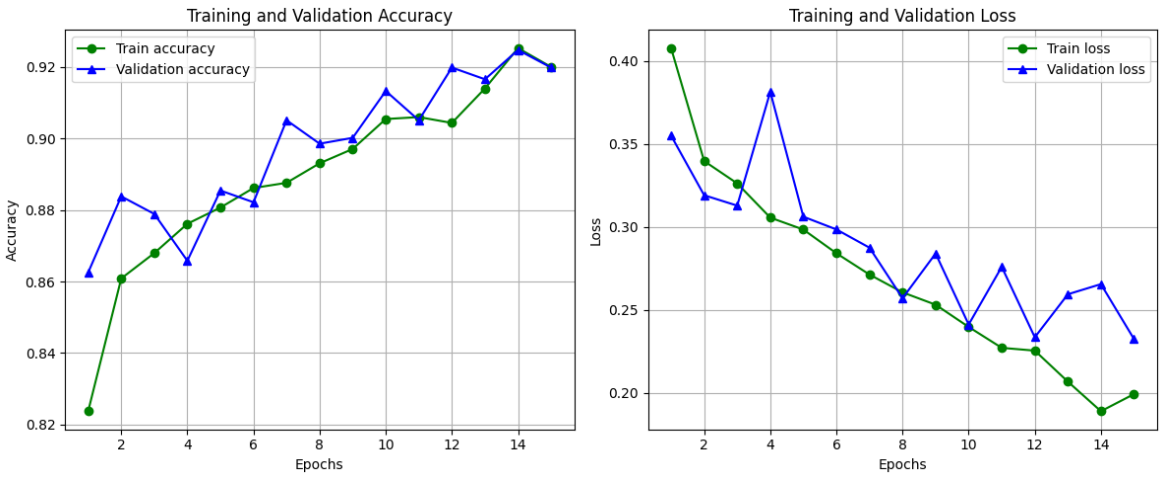
- مدل با بهینه‌ساز `Adam` و تابع هزینه `binary_crossentropy` کامپایل می‌شود. `metrics=['acc']` برای ارزیابی دقت مدل استفاده می‌شود.

۹. آموزش مدل:

- مدل با استفاده از داده‌های آموزشی `XX_train` و `y_train` آموزش داده می‌شود. تعداد دوره‌ها `۱۵`، اندازه دسته‌بندی `۲۰۰` و ۱۰٪ از داده‌ها برای ارزیابی در

هر دوره استفاده می‌شوند.

این نوع مدل به دلیل ترکیب لایه‌های کانولوشنی و بازگشتی، قادر است ویژگی‌های مهم متنی را استخراج کرده و سپس آن‌ها را در زمان پردازش کند، که برای متن‌های پیچیده بسیار مفید است.



شکل ۱۹: **Loss** مربوط به **LSTM+CNN**

۲-۴. مقایسه نتایج

بیایید عملکرد را بر اساس نمودارهایی که ارائه کرده‌اید تجزیه و تحلیل کنیم:

۱ LSTM لایه: نمودار احتمالاً مربوط به یک مدل LSTM ساده با یک لایه بازگشتی است. این افزایش مداوم در دقت آموزش در طول دوره‌ها را نشان می‌دهد، که نشان می‌دهد مدل از داده‌های آموزشی یاد می‌گیرد. با این حال، مشخص نیست که چقدر خوب تعمیم می‌یابد زیرا خط دقت اعتبارسنجی وجود ندارد.

LSTM دو لایه: این مدل دارای دو لایه LSTM است که روی یکدیگر چیده شده اند که به طور بالقوه می تواند الگوهای پیچیده تری را در داده ها ثبت کند. نمودار دقت شروع بالاتری را برای آموزش و اعتبارسنجی و شکاف کمتری بین این دو نشان می‌دهد که تعمیم بهتری را نسبت به ۱ LSTM لایه نشان می‌دهد.

CNN + LSTM: ترکیب LSTM با CNN می تواند برای کارهایی که در آنها روابط مکانی (که توسط CNN ها انجام می شود) و دنباله های زمانی (که توسط LSTM ها انجام می شود) مؤثر باشد. نمودار این معماری بالاترین دقت آموزش اولیه و پایدارترین دقت آموزشی و اعتبار سنجی را نشان می دهد که نشان دهنده قابلیت های خوب یادگیری و تعمیم است.

به طور خلاصه، از نمودارها، به نظر می رسد که معماری ترکیبی CNN + LSTM با دقت بالاتری شروع می شود و عملکرد پایداری را حفظ می کند، که نشان می دهد ممکن است بهترین مدل در بین این سه مدل باشد. ۲ LSTM لایه تعمیم بهتری را نسبت به ۱ LSTM لایه نشان می دهد همانطور که توسط آموزش دقیق تر و دقت های اعتبار سنجی نشان داده شده است. با این حال، بدون دانستن مقیاس های دقیق و زمینه مسئله، این تحلیل کاملاً کلی است.

دلیل بهتر بودن CNN+LSTM به این خاطر است که CNN تلاش می‌کند ویژگی‌های Local را حفظ کند ولی LSTM تلاش می‌کند، ویژگی‌های global را حفظ کند، لذا این مدل از هر دو مدل بهتر عمل خواهد کرد.

همچنین مدل LSTM دولایه به واسطه عمیق تر بودن مشخصا از LSTM تک لایه بهتر عمل خواهد کرد.