گزارش مینی پروژه ۲ – پیاده‌سازی شبکه‌های عصبی بازگشتی

میثم پرویزی
دانشکده مهندسی برق و کامپیوتر
پردیس دانشکده‌های فنی
دانشگاه تهران

امیر اشتری گرگری
دانشکده مهندسی برق و کامپیوتر
پردیس دانشکده‌های فنی
دانشگاه تهران

چکیده–**شبکه های عصبی بازگشتی (RNN) قادر به یادگیری ویژگی‌ها و وابستگی‌های طولانی مدت در دنباله‌ها و سری‌های زمانی هستند. RNN ها دارای پشته‌ای از واحدهای غیرخطی هستند که حداقل یکی از اتصالات موجود بین واحدها یک مسیر بازگشتی را تشکیل می‌دهد. یک RNN که به خوبی آموزش دیده باشد می‌تواند هر سیستم دینامیکی را مدل کند؛ با این حال، آموزش RNN ها با مسائل مربوط به یادگیری وابستگی‌های طولانی مدت مواجه است. در این پروژه، ما به بررسی شیوه‌های مختلف پیاده‌سازی RNN ها خواهیم پرداخت و میزان کارایی روش‌های مختلف را تحلیل خواهیم کرد. دیتاست مورد استفاده در این پروژه اطلاعات آب و هوای شهر پکن است که باید بر اساس آن‌ها وضعیت آلودگی هوای این شهر را در روزهای آینده پیش‌بینی کنیم.**

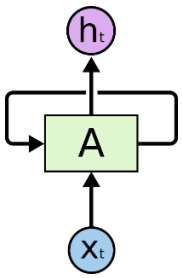
کلمات کلیدی–**یادگیری عمیق، شبکه‌های عصبی بازگشتی، LSTM، GRU، پیش‌بینی.**

۱. مقدمه

وقتی انسان‌ها به موضوعی فکر می‌کنند همواره از مجموعه اطلاعاتی که در گذشته به دست آورده‌اند برای تحلیل آن موضوع استفاده می‌کنند. همان طور که شما این متن را می‌خوانید برای درک مفهوم یک کلمه از مفهوم کلماتی که قبلاً خوانده‌اید استفاده می‌کنید. شما همه چیز را دور نمی‌اندازید تا فکر کردن را از نقطه‌ی صفر شروع کنید. افکار شما ماندگار هستند [1].

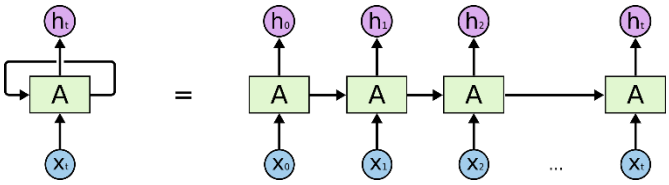
شبکه‌های عصبی معمولی نمی‌توانند این کار را انجام دهند و به نظر می‌رسد این یک نقص بزرگ باشد. به عنوان مثال تصور کنید شما قصد دارید نقاط مختلف یک فیلم را برای تشخیص اینکه چه نوع اتفاقی خواهد افتاد طبقه‌بندی کنید. مشخص نیست که یک شبکه‌ی عصبی معمولی چگونه باید از روی اتفاقات قبلی موجود در فیلم درباره اتفاقات بعدی نتیجه‌گیری کند [1].

شبکه‌های عصبی بازگشتی به همین مسئله می‌پردازند. این نوع از شبکه‌های عصبی در ساختار خود دارای حلقه‌های بازگشت هستند تا امکان باقی ماندن اطلاعات قبلی در شبکه را فراهم کنند [1].



شکل ۱. شبکه‌های عصبی بازگشتی دارای حلقه هستند.

در نمودار بالا یک تکه از شبکه عصبی به نام A بر اساس یک ورودی به نام xt خروجی ht را تولید می‌کند. یک حلقه اجازه می‌دهد اطلاعات از یک مرحله از شبکه به مرحله‌ی بعدی منتقل شود. با وجود این حلقه‌ها به نظر می‌رسد که شبکه‌های عصبی بازگشتی‌ی تفاوت زیادی با شبکه‌های عصبی معمولی داشته باشند. اما یک شبکه عصبی بازگشتی در واقع از چندین کپی از یک شبکه تشکیل شده‌اند که خروجی هرکدام به بعدی وارد می‌شود. توجه کنید که اگر حلقه را باز کنیم چه اتفاقی می‌افتد [1].



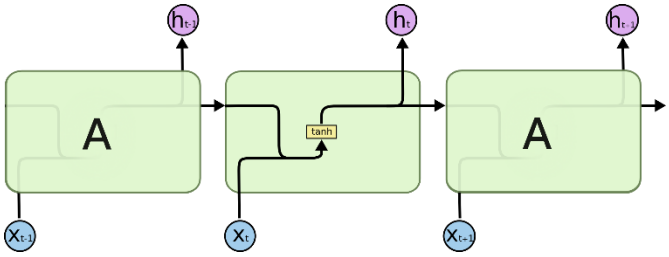
شکل ۲. یک شبکه عصبی بازگشتی باز شده

این ساختار زنجیره‌ای نشان می‌دهد که شبکه‌های عصبی بازگشتی پیوند عمیقی با مفاهیم دنباله‌ها دارند. در واقع این نوع از شبکه‌های عصبی همان

شبکه‌های عصبی معمولی هستند که برای چنین دنباله‌هایی استفاده می‌شوند. در چند سال اخیر موفقیت‌های باورنکردنی در استفاده از RNNها در موضوعات مختلفی از جمله تشخیص گفتار، مدل‌سازی زبان، ترجمه، عنوان‌گذاری روی تصاویر و... به دست آمده است [1].

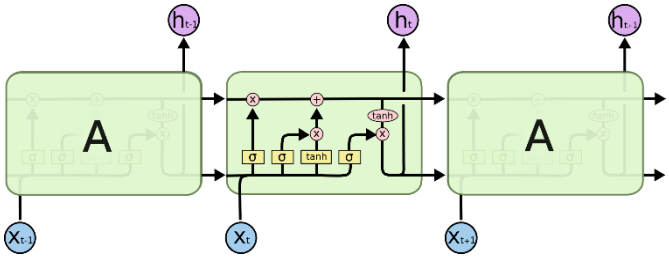
برای رسیدن به این موفقیت‌ها یک گام کلیدی نیاز بود و آن استفاده از LSTMها، یک نوع خاص از شبکه عصبی بازگشتی است که برای بسیاری از کاربردها خیلی بهتر از نسخه‌ی استاندارد کار می‌کند. تقریباً تمام نتایج فوق‌العاده در شبکه‌های عصبی بازگشتی به کمک LSTMها به دست می‌آید.

شبکه‌های حافظه‌ی طولانی کوتاه مدت (LSTM) نوع خاصی از RNN هستند که قادر به یادگیری وابستگی‌های طولانی مدت بین داده‌ها هستند. LSTMها طراحی شده‌اند تا مشکل وابستگی‌های بلندمدت بین داده‌ها را حل کنند. یادآوری اطلاعات برای بازه‌های زمانی طولانی در عمل رفتار پیش‌فرض آن‌هاست نه چیزی که آن‌ها برای یادگیری‌اش تلاش می‌کنند. تمام شبکه‌های عصبی بازگشتی دارای ساختار زنجیره‌ای از ماژول‌های تکرار شونده‌ی شبکه‌ی عصبی هستند. در RNNهای استاندارد این ماژول تکرار شونده ساختار بسیار ساده‌ای نظیر یک لایه‌ی tanh دارد [1].



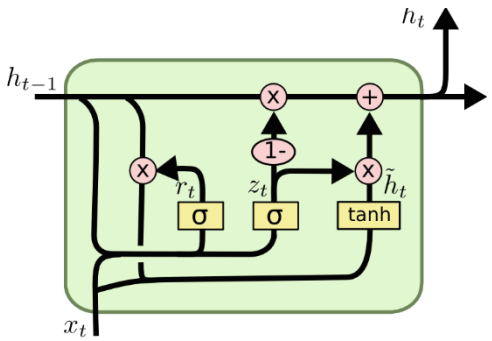
شکل ۳. ماژول تکرار شونده در RNN استاندارد فقط یک لایه دارد.

LSTMها نیز این ساختار زنجیره‌ای را دارند اما ماژول تکرار شونده یک ساختار متفاوت دارد. به جای داشتن یه لایه، چهار لایه وجود دارد که به طریق خاصی با یکدیگر تعامل می‌کنند. این ساختار باعث می‌شود تا اثر داده‌های بلند مدت نیز در شبکه باقی بماند و بتوان مدل‌های دقیق‌تری ایجاد کرد [1].



شکل ۴. ماژول تکرار شونده در LSTM دارای چهار لایه‌ی مختلف است.

یکی از معایب LSTM پیچیدگی آن است. به همین دلیل یک نسخه‌ی ساده‌تر از آم به نام GRU ارائه شده است تا پیاده‌سازی شبکه‌های عصبی بازگشتی را ساده‌تر کند [1].



شکل ۵. ساختار ماژول GRU

در این مینی‌پروژه قصد داریم با هدف پیش‌بینی میزان آلودگی هوای شهر پکن از دیتاست polution_dataSet.npy استفاده کنیم. این دیتاست شامل اطلاعات آب و هوای شهر پکن چین هست. هر ردیف شامل اطلاعات آب و هوایی یک ساعت است. اطلاعات مربوط به بازه ی زمانی ۲۰۱۰ تا ۲۰۱۵ هست و ردیف ها به ترتیب از ۲۰۱۰ تا ۲۰۱۵ برای هر ساعت ثبت شده‌اند. برای پیاده‌سازی شبکه عصبی مورد نظر از کتابخانه Tensorflow و Keras استفاده شده است. لازم به ذکر است که تنها از ۱۰۰۰۰ رکورد اول این دیتاست برای این مینی‌پروژه استفاده کرده‌ایم.

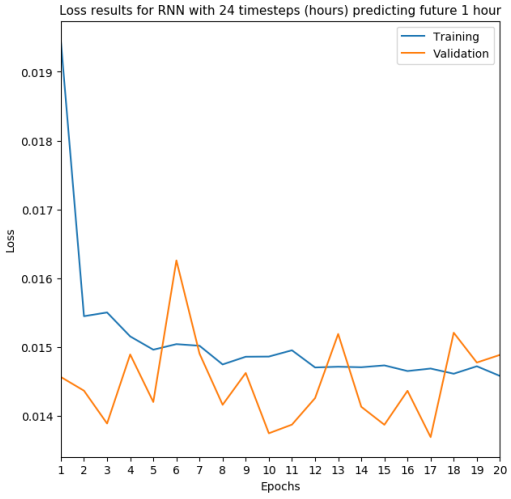
برای مقایسه عملکرد شبکه‌های عصبی بازگشتی مختلف، از هر سه ماژول RNN ساده، LSTM و GRU برای پیاده‌سازی شبکه استفاده خواهیم کرد. همچنین برای هر کدام از این سه شبکه بازگشتی داده‌ها را با شکل‌های مختلف به شبکه اعمال می‌کنیم. در حالت اول شبکه را طوری آموزش می‌دهیم تا با گرفتن داده‌های ۲۴ ساعت گذشته، آلودگی هوای ۱ ساعت آینده را پیش‌بینی کند. در حالت دوم ابتدا حجم اطلاعات را کاهش داده آن‌ها را به صورت روز به روز تبدیل می‌کنیم و شبکه را طوری آموزش می‌دهیم تا با گرفتن اطلاعات ۷ روز گذشته، آلودگی ۱ روز آینده را پیش‌بینی کند. در حالت سوم نیز مجدداً حجم اطلاعات را کاهش داده و به صورت هفته به هفته در می‌آوریم و شبکه رو طوری آموزش می‌دهیم تا با گرفتن اطلاعات ۸ هفته‌ی گذشته، آلودگی هفته‌ی آینده را پیش‌بینی کند.

۲. پیاده‌سازی با RNN

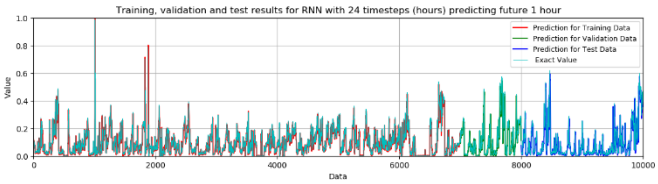
در گام اول ما برای پیش‌بینی آلودگی هوا از شبکه RNN معمولی استفاده می‌کنیم. نتایج و تحلیل روی حالت‌های مختلف در زیر آورده شده است.

الف) پیش‌بینی ساعت به ساعت

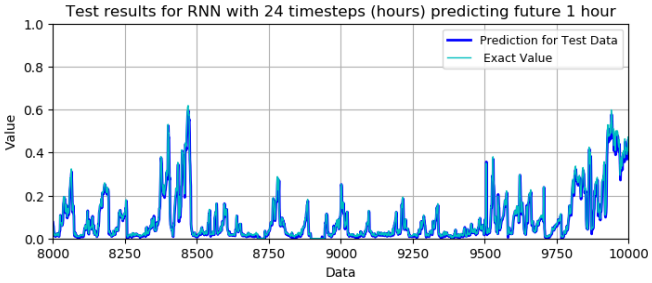
در این حالت مجموعا ۱۰۰۰۰ داده داریم که ۷۰۰۰ تای آن برای آموزش، ۱۰۰۰ تای آن برای اعتبارسنجی و ۲۰۰۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۲۴ عدد است. نمودار نتایج این حالت در شکل های ۶، ۷ و ۸ آورده شده است. کد مربوط به این حالت در فایل‌های rnn_p1_last_24_hours.py و rnn_p1_last_24_hours.ipynb موجود است.



شکل ۶. نمودار هزینه برای پیش‌بینی ساعت به ساعت در RNN



شکل ۷. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی ساعت به ساعت در RNN



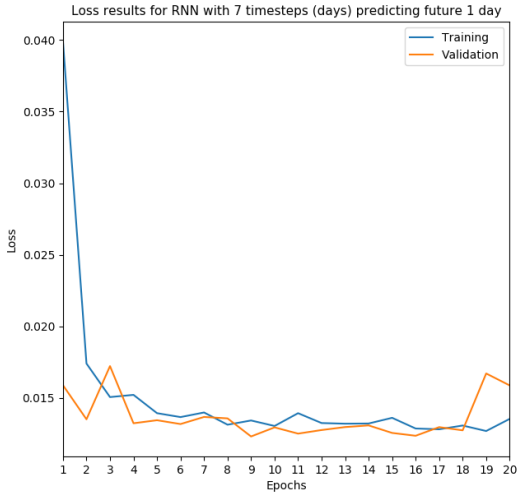
شکل ۸. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی ساعت به ساعت در داده‌های تست در RNN

ب) پیش‌بینی روز به روز

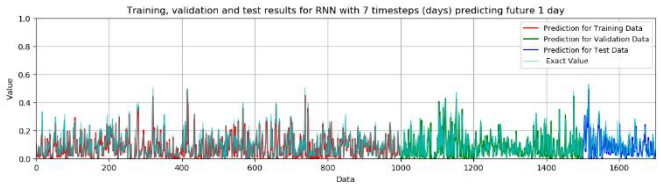
در این حالت مجموعا ۱۷۰۰ داده را به صورت ۲۴ درمیان از داده‌های اولیه انتخاب می‌کنیم که ۱۰۰۰ تای آن برای آموزش، ۵۰۰ تای آن برای اعتبارسنجی و ۲۰۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۷ عدد است. نمودار نتایج این حالت در شکل های ۹، ۱۰ و ۱۱ آورده شده است. کد مربوط به این حالت در فایل‌های rnn_p2_last_7_days.py و rnn_p2_last_7_days.ipynb موجود است.

ج) پیش‌بینی هفته به هفته

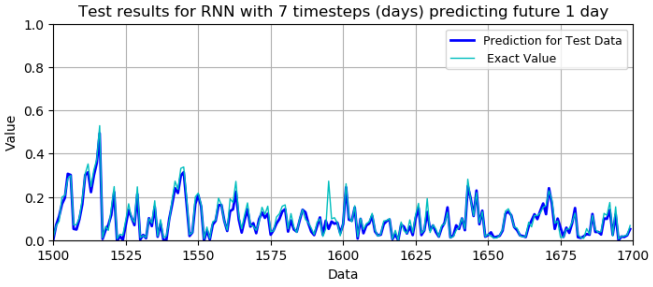
در این حالت مجموعا ۲۰۰ داده را به صورت ۲۴x۷ در میان از داده‌های اولیه انتخاب می‌کنیم که ۱۲۰ تای آن برای آموزش، ۵۰ تای آن برای اعتبارسنجی و ۳۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۸ عدد است. نمودار نتایج این حالت در شکل های ۱۲، ۱۳ و ۱۴ آورده شده است. کد مربوط به این حالت در فایل‌های rnn_p3_last_8_weeks.py و rnn_p3_last_8_weeks.ipynb موجود است.



شکل ۹. نمودار هزینه برای پیش‌بینی روز به روز در RNN



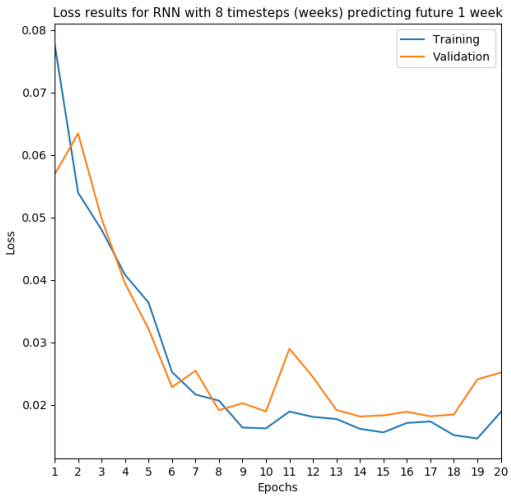
شکل ۱۰. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی روز به روز در RNN



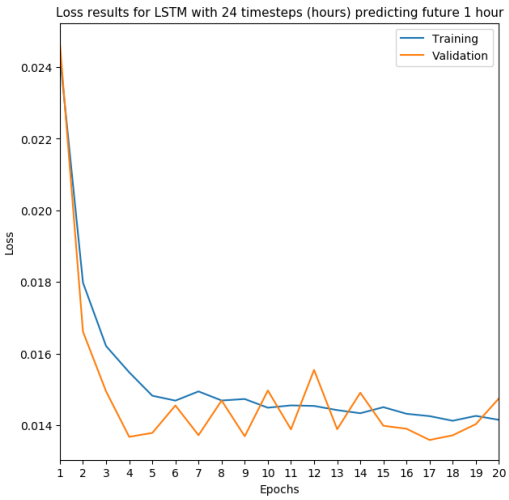
شکل ۱۱. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی روز به روز در داده‌های تست در RNN

۳. پیاده‌سازی با LSTM

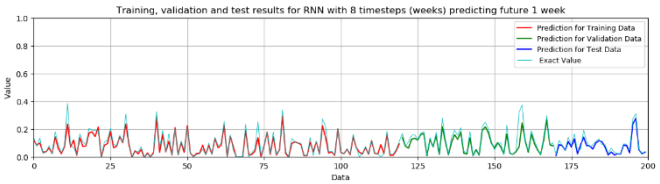
در گام دوم ما برای پیش‌بینی آلودگی هوا از شبکه LSTM استفاده می‌کنیم. نتایج و تحلیل روی حالت‌های مختلف در زیر آورده شده است.



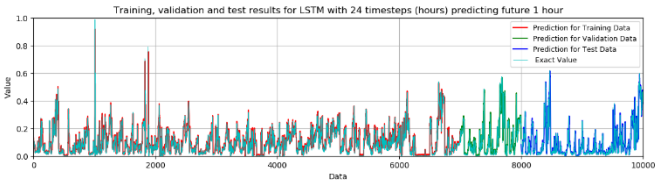
شکل ۱۲. نمودار هزینه برای پیش‌بینی هفته به هفته در RNN



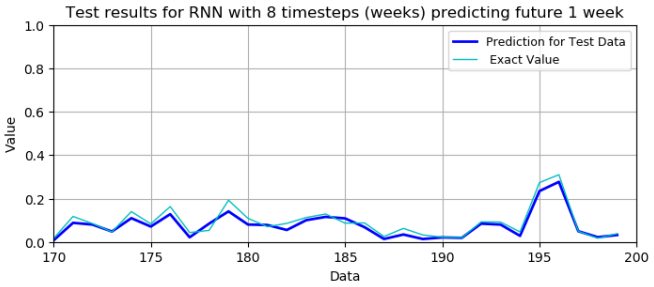
شکل ۱۵. نمودار هزینه برای پیش‌بینی ساعت به ساعت در LSTM



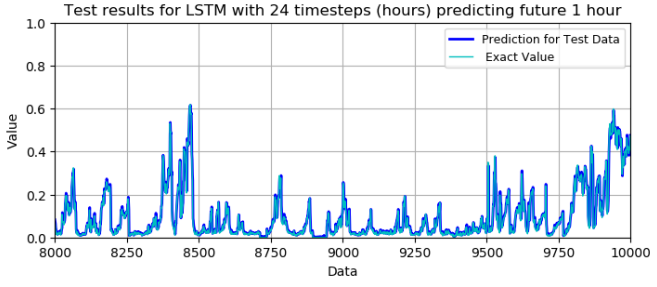
شکل ۱۳. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی هفته به هفته در RNN



شکل ۱۶. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی ساعت به ساعت در LSTM



شکل ۱۴. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی هفته به هفته در داده‌های تست در RNN



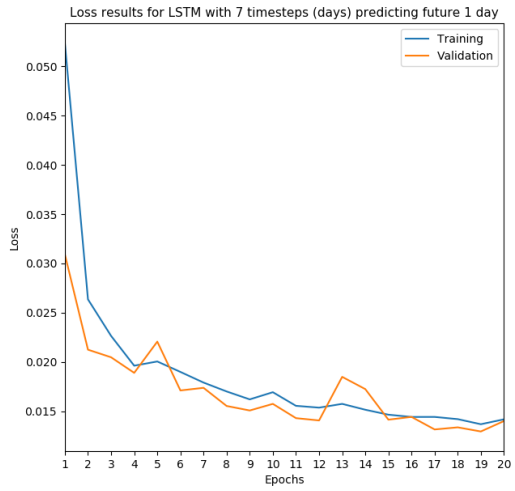
شکل ۱۷. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی ساعت به ساعت در داده‌های تست در LSTM

الف) پیش‌بینی ساعت به ساعت

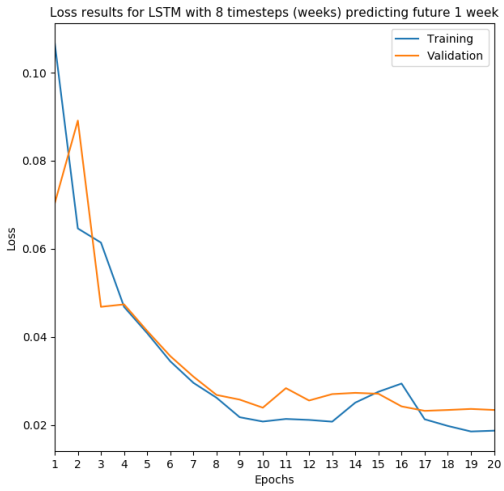
در این حالت نیز مانند RNN معمولی مجموعا ۱۰۰۰۰ داده داریم که ۷۰۰۰ تای آن برای آموزش، ۱۰۰۰ تای آن برای اعتبارسنجی و ۲۰۰۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۲۴ عدد است. نمودار نتایج این حالت در ادامه آورده شده است. نمودار نتایج این حالت در شکل های ۱۵، ۱۶ و ۱۷ آورده شده است. کد مربوط به این حالت در فایل‌های lstm_p1_last_24_hours.py و lstm_p1_last_24_hours.ipynb موجود است.

ب) پیش‌بینی روز به روز

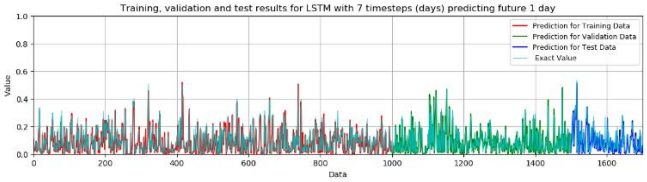
در این حالت نیز مجموعا ۱۷۰۰ داده را به صورت ۲۴ درمیان از داده‌های اولیه انتخاب می‌کنیم که ۱۰۰۰ تای آن برای آموزش، ۵۰۰ تای آن برای اعتبارسنجی و ۲۰۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۷ عدد است. نمودار نتایج این حالت در شکل های ۱۸، ۱۹ و ۲۰ آورده شده است. کد مربوط به این حالت در فایل‌های lstm_p2_last_7_days.py و lstm_p2_last_7_days.ipynb موجود است.



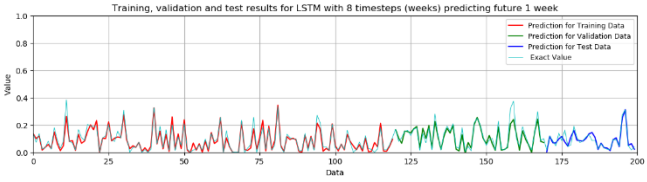
شکل ۱۸. نمودار هزینه برای پیش‌بینی روز به روز در LSTM



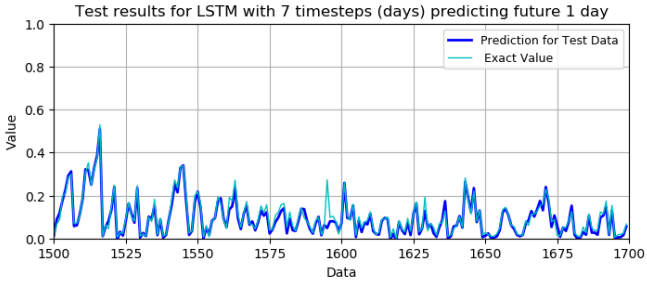
شکل ۲۱. نمودار هزینه برای پیش‌بینی هفته به هفته در LSTM



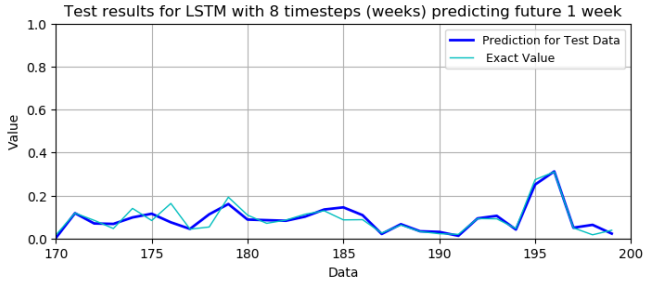
شکل ۱۹. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی روز به روز در LSTM



شکل ۲۲. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی هفته به هفته در LSTM



شکل ۲۰. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی روز به روز در داده‌های تست در LSTM



شکل ۲۳. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی هفته به هفته در داده‌های تست در LSTM

ج) پیش‌بینی هفته به هفته

در این حالت نیز مجموعا ۲۰۰ داده را به صورت ۲۴x۷ در میان از داده‌های اولیه انتخاب می‌کنیم که ۱۲۰ تای آن برای آموزش، ۵۰ تای آن برای اعتبارسنجی و ۳۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۸ عدد است. نمودار نتایج این حالت در شکل های ۲۱، ۲۲ و ۲۳ آورده شده است. کد مربوط به این حالت در فایل‌های lstm_p3_last_8_weeks.py و lstm_p3_last_8_weeks.ipynb موجود است.

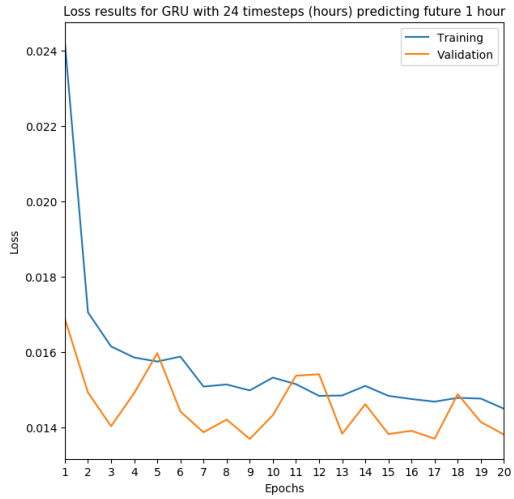
۴. پیاده‌سازی با GRU

در گام سوم ما برای پیش‌بینی آلودگی هوا از شبکه GRU استفاده می‌کنیم. نتایج و تحلیل روی حالت‌های مختلف در ادامه آورده شده است.

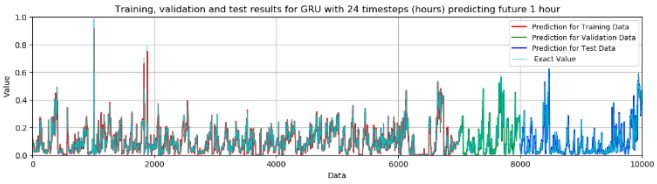
الف) پیش‌بینی ساعت به ساعت

در این حالت نیز مجموعا ۱۰۰۰۰ داده داریم که ۷۰۰۰ تای آن برای آموزش، ۱۰۰۰ تای آن برای اعتبارسنجی و ۲۰۰۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۲۴ عدد است. نمودار نتایج این حالت در ادامه آورده شده است. نمودار نتایج این حالت در شکل های ۲۴،

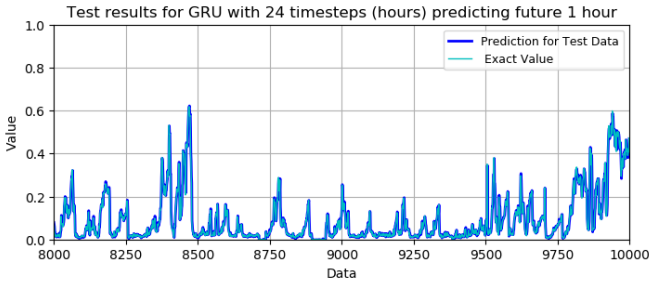
۲۵ و ۲۶ آورده شده است. کد مربوط به این حالت در فایل‌های gru_p1_last_24_hours.py و gru_p1_last_24_hours.ipynb موجود است.



شکل ۲۴. نمودار هزینه برای پیش‌بینی ساعت به ساعت در GRU



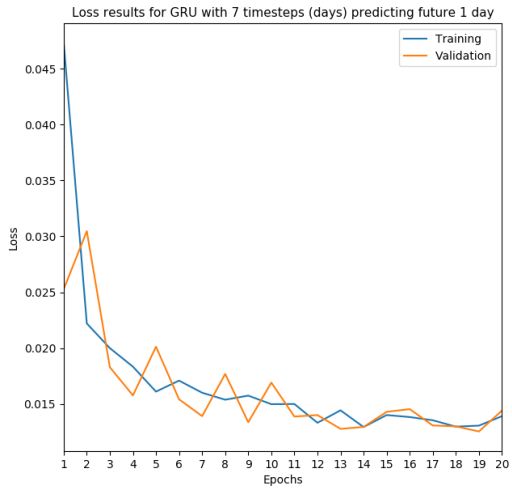
شکل ۲۵. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی ساعت به ساعت در GRU



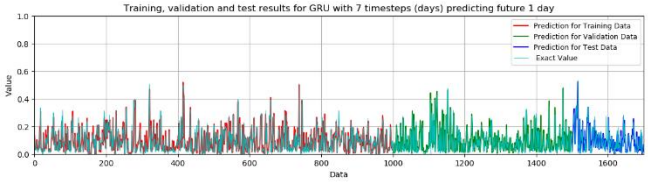
شکل ۲۶. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی ساعت به ساعت در داده‌های تست در GRU

ب) پیش‌بینی روز به روز

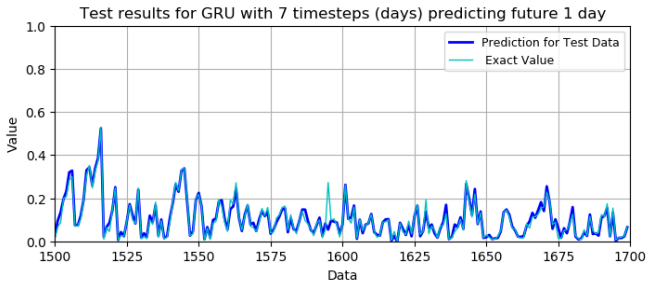
در این حالت نیز مجموعا ۱۷۰۰ داده را به صورت ۲۴ درمیان از داده‌های اولیه انتخاب می‌کنیم که ۱۰۰۰ تای آن برای آموزش، ۵۰۰ تای آن برای اعتبارسنجی و ۲۰۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۷ عدد است. نمودار نتایج این حالت در شکل های ۲۷، ۲۸ و ۲۹ آورده شده است. کد مربوط به این حالت در فایل‌های gru_p2_last_7_days.py و gru_p2_last_7_days.ipynb موجود است.



شکل ۲۷. نمودار هزینه برای پیش‌بینی روز به روز در GRU



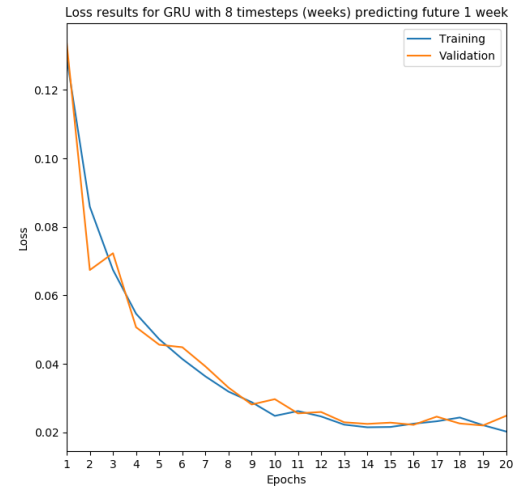
شکل ۲۸. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی روز به روز در GRU



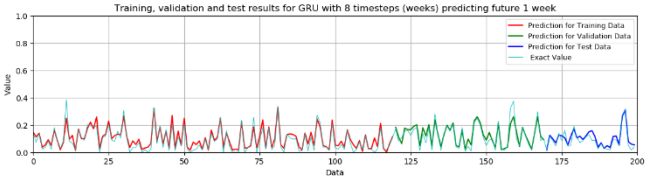
شکل ۲۹. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی روز به روز در داده‌های تست در GRU

ج) پیش‌بینی هفته به هفته

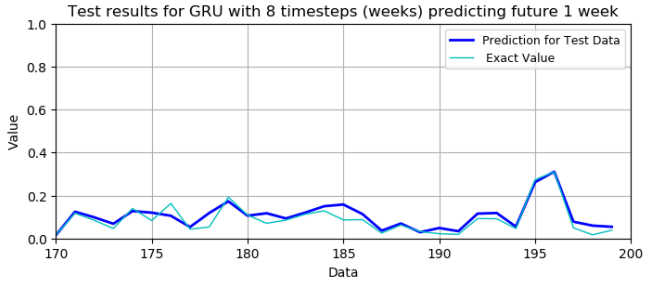
در این حالت نیز مجموعا ۲۰۰ داده را به صورت ۲۴x۷ در میان از داده‌های اولیه انتخاب می‌کنیم که ۱۲۰ تای آن برای آموزش، ۵۰ تای آن برای اعتبارسنجی و ۳۰ تای آن برای تست استفاده می‌شود. همچنین تعداد timestep ها برابر با ۸ عدد است. نمودار نتایج این حالت در شکل های ۳۰، ۳۱ و ۳۲ آورده شده است. کد مربوط به این حالت در فایل‌های gru_p3_last_8_weeks.py و gru_p3_last_8_weeks.ipynb موجود است.



شکل ۳۰. نمودار هزینه برای پیش‌بینی هفته به هفته در GRU



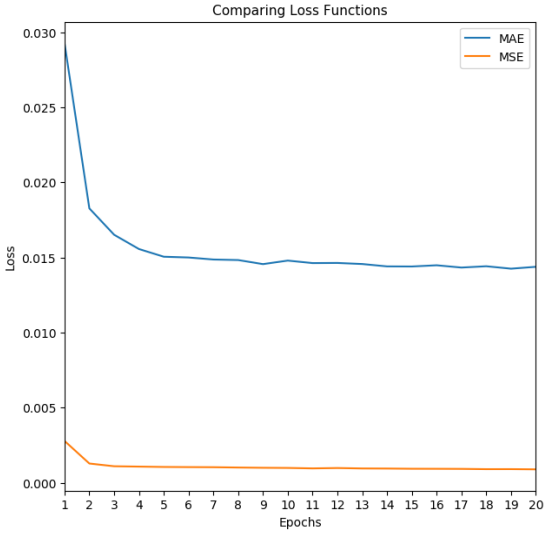
شکل ۳۱. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی هفته به هفته در GRU



شکل ۳۲. نمودار مقایسه پیش‌بینی‌ها با داده‌های واقعی برای پیش‌بینی هفته به هفته در داده‌های تست در GRU

۵. بررسی تابع‌های هزینه مختلف

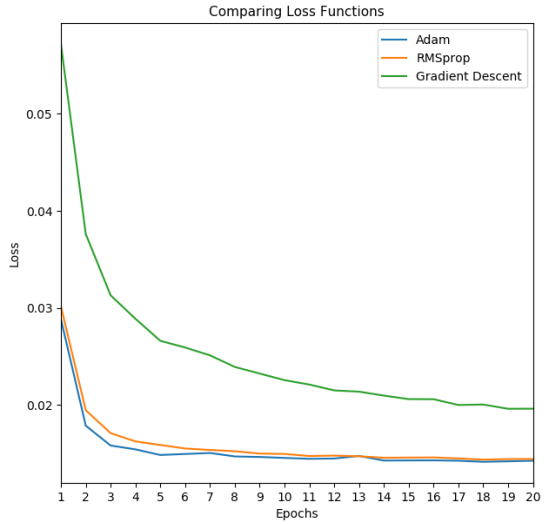
در این بخش عملکرد شبکه LSTM را برای تابع‌های هزینه میانگین مربع خطا (MSE) و میانگین قدرمطلق خطا (MAE) را بررسی می‌کنیم. همانطور که از روی شکل ۳۳ مشخص است این دو تابع در مورد این مسئله و با به کار گرفتن LSTM تفاوت قابل توجهی ایجاد می‌کنند و تابع MSE نتیجه‌ی بسیار بهتری می‌دهد.



شکل ۳۳. مقایسه توابع هزینه

۶. بررسی روش‌های بهینه‌سازی مختلف

در این بخش عملکرد شبکه LSTM را برای روش‌های بهینه‌سازی Adam، RMSProp و Gradient Descenct را بررسی می‌کنیم. همانطور که از روی شکل ۳۴ مشخص است روش Adam بهترین نتیجه را می‌دهد.



شکل ۳۴. مقایسه روش‌های بهینه‌سازی

۷. استفاده از تکنیک Dropout

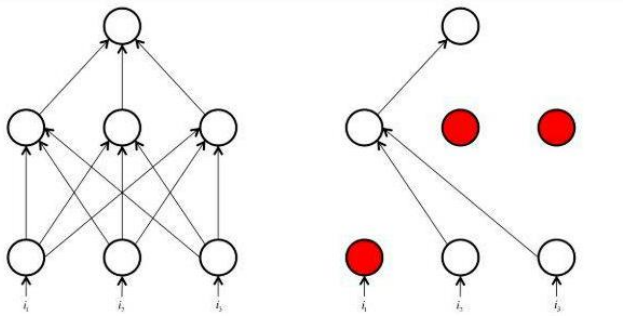
Dropout یکی از تکنیک‌های Regularization بوده، که با ارایه متدی متفاوت جهت کاهش Overfitting در تلاش است در زمان آموزش، تعدادی نرون با احتمالp را فعال نگه داشته و بقیه را صفر قرار می‌دهد. این اقدام باعث افزایش دقت سیستم حتی در نبود بخشی از اطلاعات می‌شود و وابستگی شبکه به هر یک از نورون‌ها را کاهش می‌دهد.

Dropout در لغت به معنی رها کردن و حذف تصادفی است و در شبکه های عصبی هم دقیقا به همین منظور به کار میرود. وقتی Dropout استفاده می‌کنیم به این معناست که برخی از نورون‌ها در لایه پنهان (Hidden) و یا آشکار (Visible) در هنگام آموزش به صورت تصادفی نادیده بگیریم. به طور خاص تر به این معناست که با احتمال P یا ۱-P این نورونهامون رو آپدیت کنیم یا نکنیم. این یک روش بسیار کارآمد برای انجام مدل به طور میانگین با شبکه‌های عصبی است.

حال Dropout را در شبکه GRU اعمال می‌کنیم، شبکه در حالت عادی میزان خطا برای داده تست را ۰,۰۱۷ گزارش می‌داد، حالت‌های مختلف برای Dropout را در این شبکه تست کرده و در جدول ۱ آورده شده است.

پارامترها در این اجرا برابر مقادیر زیر است:

BATCH_SIZE = 10,
NUM_EPOCHS = 20,
HIDDEN_SIZE = 40,
loss='mae',
optimizer='adam'.



شکل ۳۵. Dropout باعث غیرفعال شدن تعدادی از نورون‌ها در هنگام آموش می‌شود.

با توجه به نتایج می‌توان به خوبی مشاهده نمود زمانی که Dropout در لایه Recurrent اعمال می‌گردد همواره باعث بهبود نتیجه خروجی می‌گردد اما زمانی که Dropout در لایه Input اعمال می‌گردد باعث کاهش دقت در خروجی می‌شود، دلیل آن را می‌توان این گونه توصیف نمود که همانند مفهوم Dropout در شبکه های MLP ما در لایه های وسط همواره از Dropout استفاده میکردیم و بر روی لایه ورودی این کار را انجام نمیدادیم چون هدف کم کردن ابعاد داده‌های اصلی نیست بلکه بالابردن دقت شبکه در صورت نبود داده‌ای وجلوگیری از وابستگی شبکه به واحدی خاص است لذا با توجه به نتایج نیز می‌توان درک نمود که هرگاه Dropout را در لایه‌وسط یا همان Recurrent اعمال نماییم باعث بهبود دقت عملکرد سیستم خواهد شد.

جدول ۱. نتایج استفاده از تکنیک Dropout

Test #	Dropout	Recurrent Dropout	Error
1	0	0	**0.017**
2	0	0.2	**0.014**
3	0	0.4	**0.014**
5	0.2	0.4	**0.022**
6	0.2	0	**0.024**
7	0.4	0	**0.035**
8	0.4	0.2	**0.034**
9	0	0.6	**0.015**

لذا بهترین پارامتر مقادیر تست سوم که پارامتر Dropout برابر صفر و پارامتر Recurrent مقدار ۰,۴ دارند را دارد، اگر میزان Dropout را از این مقدار بیشتر کنیم باعث افت دقت شبکه در مرحله آموزش و در کل خواهد شد.

۸. نتیجه‌گیری

با بررسی نتایج بدست آمده در مورد این مسئله متوجه شدیم که استفاده از LSTM باعث بهبود نتایج شبکه عصبی بازگشتی می‌شود. البته با وجود عدم وجود پیچیدگی در داده‌های مسئله تفاوت LSTM و GRU با RNN ساده چندان قابل مشاهده نیست.

همچنین در مورد تابع هزینه و روش بهینه‌سازی مشاهده شد که MSE و Adam بهترین نتایج را داشتند.

مراجع

[1] C. Olah, “Understanding LSTM Networks.” [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 28-Dec-2018].