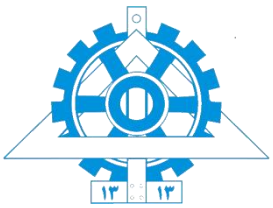


دانشگاه تهران

پردیس دانشکده های فنی

دانشکده مهندسی برق و کامپیوتر

پروژه درس یادگیری تعاملی

استاد: جناب آقای دکتر نیلی

حافظ قائمی: ۸۱۰۱۹۹۲۳۹

محمد مهدی مهمانچی :۸۱۰۱۹۹۲۸۷

نیمسال اول سال تحصیلی ۱۴۰۰-۱۳۹۹

مقدمه و کاربرد مسئله در دنیای واقعی

امروزه خرید و فروش سهام یکی از رایج ترین روش‌ها برای سرمایه گذاری و افزایش دارایی می باشد. بعضا حتی افرادی را می‌بینیم که تمام وقت خود را صرف بررسی شرایط بازار و یافتن استراتژی های سرمایه گذاری در بورس می کنند. در چنین شرایطی اصلا جای تعجب ندارد که مشاهده کنیم امروزه بسیاری از سرمایه گذاران در جهان به روش های یادگیری ماشین برای آنالیز و پیش بینی شرایط سهام بورس رو آورده‌اند. اما همانطور که می‌دانیم پیش بینی آینده بازار بر اساس داده های گذشته آن کار بسیار دشواری می باشد.

در این پروژه ما از شبکه های عصبی به علت توانایی آن ها در مدل سازی غیر خطی بین متغیر ها در بخش function approximation یادگیری تعاملی (تقویتی) در محیط‌های پیوسته برای کمک به تعیین استراتژی سرمایه‌گذاران بورس استفاده می‌کنیم. از آن جایی که مدل سازی محیط و تعین MDP در این مسئله غیر ممکن است از الگوریتم Q-learning که یک الگوریتم model-free است استفاده می کنیم. این روش همچنین این امکان را فراهم می کند تا بتوانیم بین exploration و exploitation به کمک روش epsilon greedy تعادل برقرار نماییم.

اکثر پورتفولیوها در واقعیت شامل تعدادی سهام با نوسان زیاد (high beta) و تعدادی سهام با نوسان کم (low beta) می باشند. استراتژی معقول بدین صورت است که سرمایه‌گذار هنگام شروع به بالا رفتن سهام با ضریب بتای بالا، مقداری از این سهام را می‌فروشد و تعدادی از سهم‌های با بتای پایین که ریسک نوسان کمی دارد می‌خرد. بعد از گذشت زمان و هنگامی که سرمایه‌گذار احساس کرد که سهم‌های با بتای بالا به بیشترین مقدار خود در آن بازه زمانی رسیده‌اند، آنها را می‌فروشد و با سود به دست آمده دوباره سهام با بتای پایین می‌خرد. این چرخه، با تکرار خود استراتژی سودمندی محسوب می‌شود. در واقعیت مشکل اینجاست که مدیریت پورتفولیو و تشخیص زمان مناسب برای اجرای این استراتژی به صورت دستی یا با تحلیل‌های متداول سخت و زمانبر است. استفاده از روش‌هایی مانند Long Short-Term Memory Recurrent Neural Network(LSTM RNN) [1, 2] و روش‌های یادگیری تقویتی عمیق مانند Deep Q-Network (DQN) [3, 4] و Deep Deterministic Policy Gradient (DDPG) [5, 6] می‌تواند سرمایه‌گذاران را برای پیش‌بینی وضعیت بازار و بیشینه کردن سود خود یاری کند.

در این پروژه برای ساده سازی و اینکه یک مسئله واقعی را مدل کرده باشیم، پورتفولیویی شامل یک سهام با نوسان بالا و یک سهام با نوسان پایین را در نظر می گیریم. هدف آن است که به شخص پیشنهاد بدهیم در هر time step چه مقدار از یکی از سهام ها را بفروشد و به جای آن سهام دیگر را بخرد.

در انتهای این بخش پارامتر مهم Beta را تعریف می کنیم.

بتا پارامتری برای اندازه گیری میزان ناپایداری و نوسان یک سهام می باشد که از رابطه زیر محاسبه می شود:

$$\beta = \frac{Cov(r_i, r_m)}{Var(r_m)}$$

که در آن r_i میزان بازگشت سهام مورد نظر و r_m میزان بازگشت کل بازار در یک دوره زمانی مشخص است. معمولا میزان بتا برای یک دوره چندساله و با بازه‌های یک ماهه (برای محاسبه بازگشت) محاسبه می‌شود.

معمولا سهم‌های با مقدار بتا کمتر از ۱ کم نوسان و سهم‌های با مقدار بتا بیشتر از ۱ پرنوسان محسوب می‌شوند.

مجموعه داده

در این پروژه تعداد هفت سهام با ضریب بتا بالا و هفت سهام با ضریب بتا پایین برای تشکیل پرتفولیوها در نظر گرفته شده‌اند. این سهام‌ها با توجه به معیار S&P 500 [7] که عملکرد ۵۰۰ شرکت بزرگ موجود در بازار سهام آمریکا را بررسی می‌کنند، انتخاب شده و داده‌های مربوط به آنها از طریق API پایتون Yahoo Finance [8] استخراج شده است. سهام‌های انتخاب شده به همراه ضریب بتای آنها در جدول ۱ آمده‌اند.

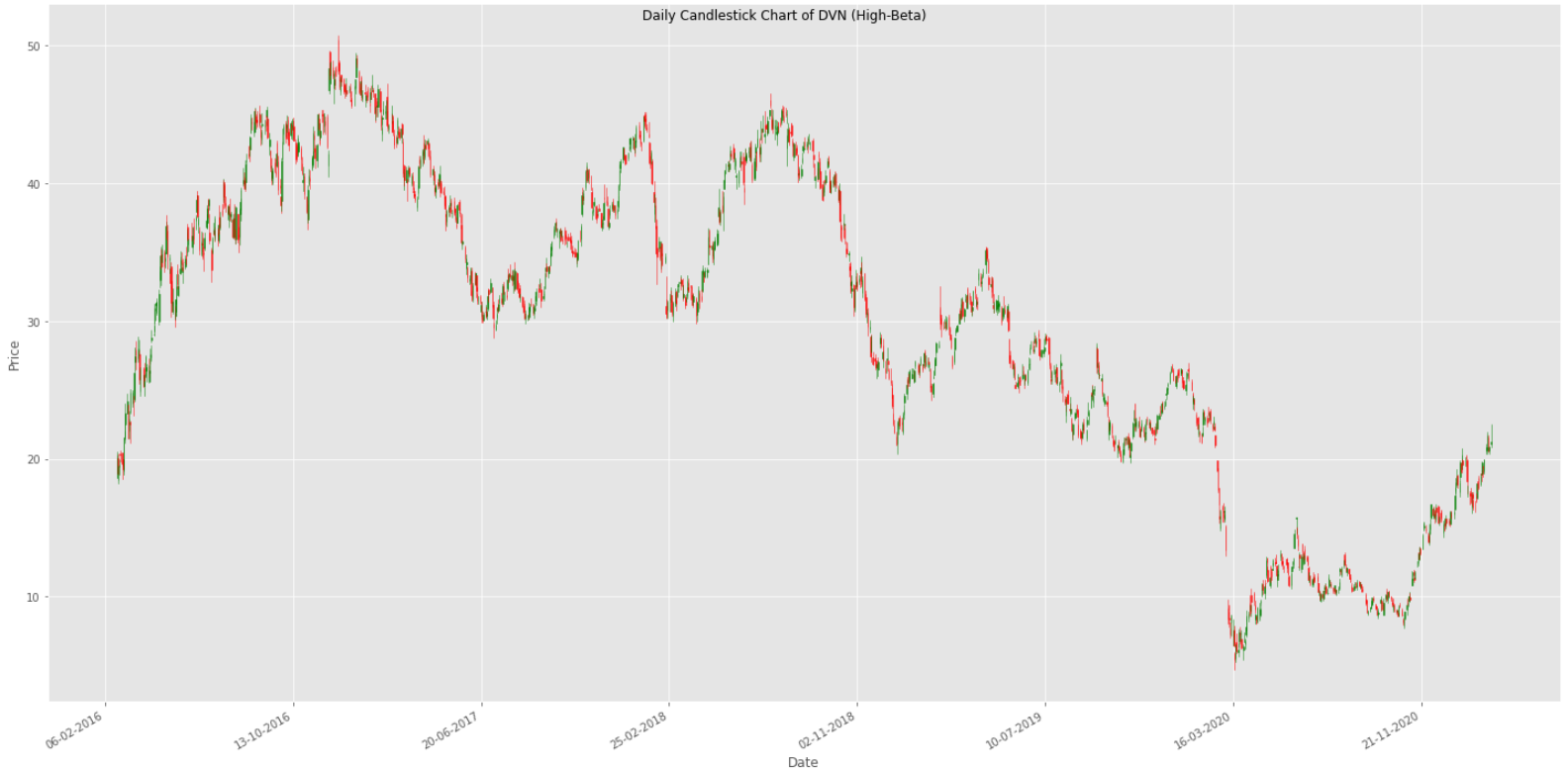
جدول ۱- سهام پرنوسان و کم نوسان درنظرگرفته شده برای ساخت پورتفولیو

نام شرکت	مخفف نام سهم (سمبل)	Beta (5 year/monthly)	high/low beta
Advanced Micro Devices (AMD)	AMD	2.20	High
NIC, Inc.	EGOV	0.26	Low
United Rentals, Inc.	URI	2.12	High
Meridian Bioscience, Inc	VIVO	0.54	Low
Freeport-McMoRan Inc.	FCX	2.23	High
Owens & Minor, Inc.	OMI	0.26	Low
Devon Energy Corporation	DVN	3.45	High
Lakeland Industries, Inc.	LAKE	0.06	Low

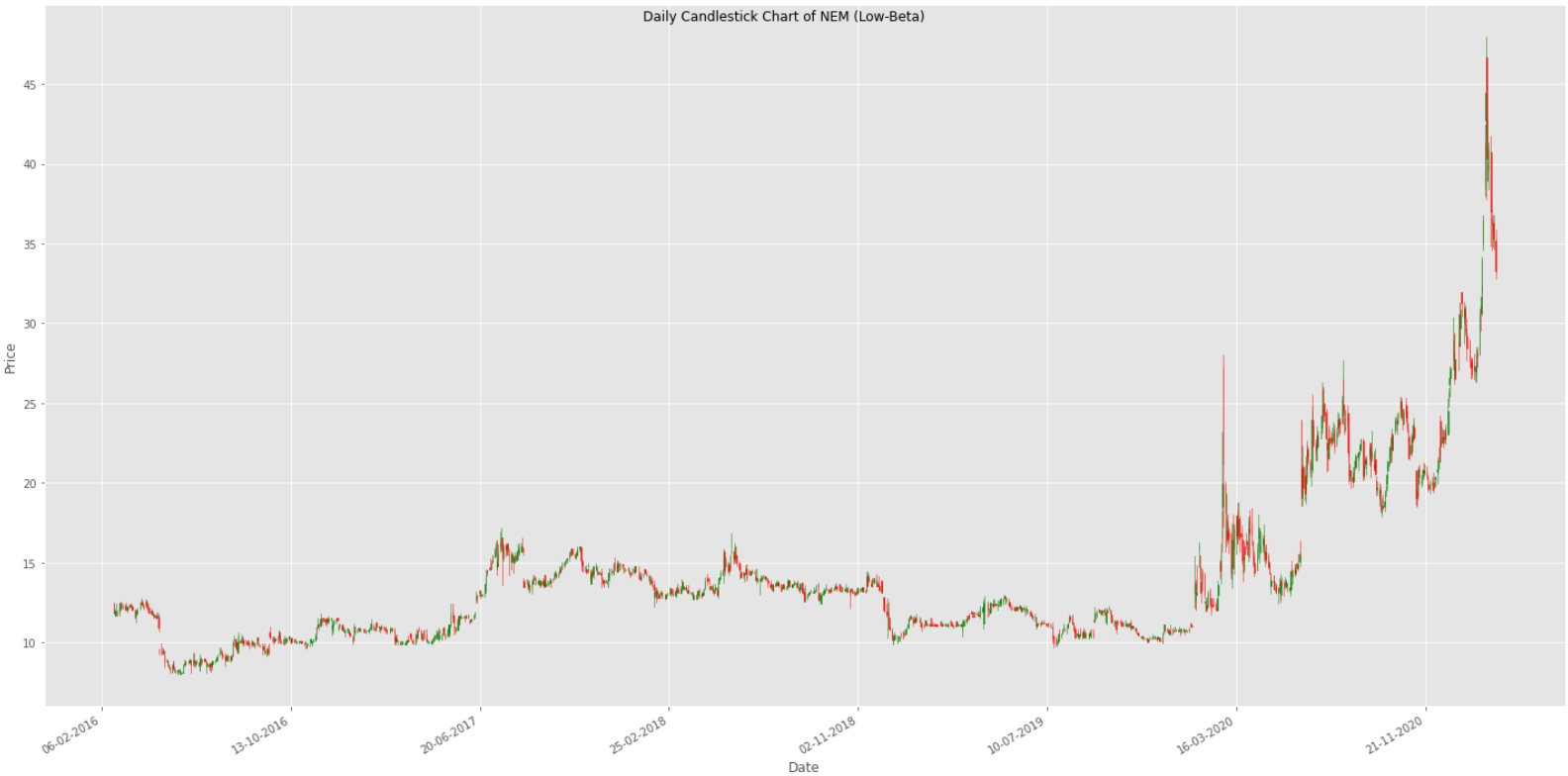
IPG Photonics Corporation	IPGP	1.47	High
Portland General Electric Company	POR	0.32	Low
SVB Financial Group	SIVB	2.08	High
Newmont Corporation	NEM	0.25	Low

چون ضریب بتا برای یک دوره ۵ ساله محاسبه شده‌است، داده‌های هر سهم را برای این ۵ سال استخراج می‌کنیم. از آنجا که به تعداد شش سهم از هر یک از سهام پرنوسان و کم نوسان داریم، می‌توان ۳۶ پورتفولیو دو سهمی مختلف با یک سهم high-beta و یک سهم low-beta ساخت. از این ۳۶ پورتفولیوی ممکن ۸۰ درصد را به عنوان داده آموزش و ۲۰ درصد را به عنوان داده تست در نظر می‌گیریم.

نمودار شمعی مربوط به دو سهم (یکی پر نوسان و دیگری کم نوسان در زیر کشیده شده‌اند.)



شکل ۱- نمودار شمعی یک سهم پرنوسان



شکل ۲- نمودار شمعی یک سهم کم نوسان

مدل سازی مسئله

همانطور که در مقدمه ذکر کردیم، برای حل این مسئله از روش Deep Q-learning استفاده خواهیم کرد. در ادامه توضیحات بیشتری درباره مدل سازی و چگونگی پیاده سازی روش خود خواهیم داد.

هر حالت در این مسئله به صورت یک $2n + 4$ تایی مرتب تعریف می‌شود؛ n مولفه اول تاریخچه قیمت سهام شماره یک در بازه‌های n روزه است، مولفه دوم تاریخچه قیمت سهام شماره دو در بازه‌های n روزه است، چهار مولفه آخر نیز به ترتیب تعداد سهم سهام های شماره یک و دو در بازه n، مقدار کل دارایی شخص در انتهای روز آخر بازه n روزه، و مقدار پول نقد باقیمانده سرمایه‌گذار (این مقدار کم باقیمانده از مابه‌التفات خرید و فروش سهام و همچنین کم کردن هزینه تبادل سهام (transaction cost) که مقدار اندکی (۰.۰۰۱ دلار) در نظر گرفته شده است. به دست می‌آید.) می‌باشد. مقدار سرمایه اولیه یک میلیون دلار در نظر گرفته شده است که در ابتدا، عامل یادگیری نصف آن را سهام پرنوسان و نصف دیگر را سهام کم نوسان می‌خرد و با توجه به پورتفولیوی انتخاب شده در اپیزود حالت ابتدایی مشخص می‌شود. هدف این است که در پایان هر بازه n روزه (معاملات عامل در پایان هر بازه n روز انجام می‌شود.) یک عمل مناسب به عامل پیشنهاد داده شود. مقدار n یک ابرمتغیر است که می‌تواند تغییر کند.

برای انتخاب مجموعه عمل های ممکن با انتخاب های گوناگونی روبه رو هستیم. می توانیم هیچ محدودیتی برای اعمال در نظر نگیریم و شخص بتواند هر مقداری از یکی از سهام ها را بفروشد و به جای آن سهام دیگر را بخرد. در چنین شرایطی واضح است که فضای عمل به صورت پیوسته خواهد بود و پیچیدگی زیادی را به مسئله تحمیل می‌کند. در اینحالت باید از الگوریتم‌های دیگری به جز DQN مانند DDPG استفاده کنیم. به همین دلیل به گونه ای سعی کردیم فضای عمل را گسسته سازی نماییم. مجموعه اعمالی که برای این مسئله انتخاب کردیم شامل ۷ عمل می باشد؛ یکی از این اعمال این است که شخص هیچ خرید و فروشی انجام ندهد و تعداد هر یک از دو سهم خود را برای n روز آینده ثابت نگه دارد. همچنین ۶ عمل متناظر با اعداد ۰.۲۵-، ۰.۱-، ۰.۰۵-، ۰.۰۵، ۰.۱ و ۰.۲۵ تعریف می کنیم. عمل متناظر با اعداد منفی به معنی فروش سهام با نوسان بالا و خرید سهام با نوسان پایین است و عمل متناظر با اعداد مثبت عکس آن می باشد. به طور مثال عمل متناظر با عدد ۰.۱ بدین معناست که شخص به ارزش ۰.۱ از تمام دارایی خود، از سهام با نوسان پایین بفروشد و معادل آن سهام با نوسان بالا بخرد.

هدف این مسئله آن است که شخص بتواند با سرمایه گذاری مناسب دارایی خود را افزایش دهد. بنابراین تابع پاداش باید به گونه ای در راستای این هدف باشد. به همین دلیل تابع پاداش را به صورت اختلاف سرمایه شخص

در آخرین روز بازه n روزه قبلی و آخرین روز بازه n روزه فعلی تعریف می کنیم. البته برای کنترل نوسان آن یک ترم جریمه به آن اضافه می کنیم. در نهایت تابع پاداش را به صورت زیر تعریف می کنیم:

$$R = (v_t - v_{t-1}) - \lambda\, std(r_t); \quad \forall t \in [1, T]$$

که در آن T زمان فعلی می باشد.

برای پیاده سازی مدل خود از کتابخانه PyTorch برای شبکه عصبی و کتابخانه Pandas برای کار کردن با مجموعه داده استفاده کرده‌ایم.

جدول۲ - معماری شبکه

Layer	Output Dimension	Activation
Input	$(state, 128)$	ReLu
Dense	(128,128)	ReLu
Dense	(128,128)	ReLu
Dense	(128, n_actions)	Sigmoid

معماری شبکه DQN در جدول ۲ آمده است. لازم به ذکر است که برای کنترل پایداری از دو شبکه یکی اصلی و دیگری هدف (target network) استفاده کرده‌ایم. همچنین برای از بین بردن همبستگی بین نمونه‌ها (حالت‌ها) از روش experience replay بهره برده‌ایم. تابع Loss به صورت Mean Squared Error بین Q-value های دو شبکه اصلی و هدف تعریف می‌شود. فرمولهای محاسبه این Q-value ها و همچنین آپدیت وزن‌های نتورک هدف در زیر آمده است:

نحوه محاسبه Q-value های نتورک اصلی:

$$Q(s, a) = r_t + \gamma \max_{a'} Q(s', a')$$

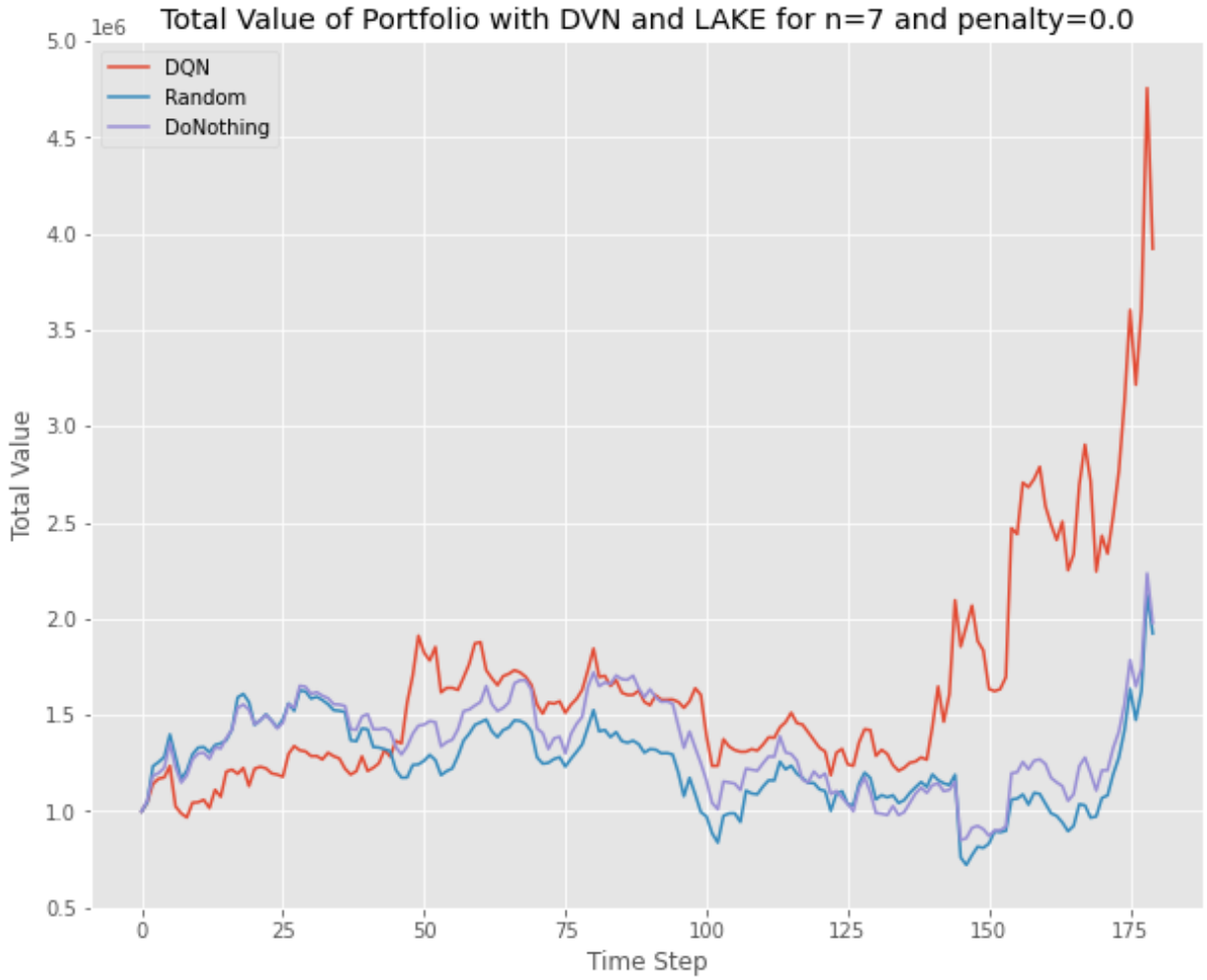
که در آن مقادیر Q-value سمت راست خروجی نتورک هدف می‌باشند. مقادیر $reward$ روی هر $batch$ نرمال می‌شوند تا با خروجی نتورک که بین ۰ و ۱ است هماهنگ شوند.

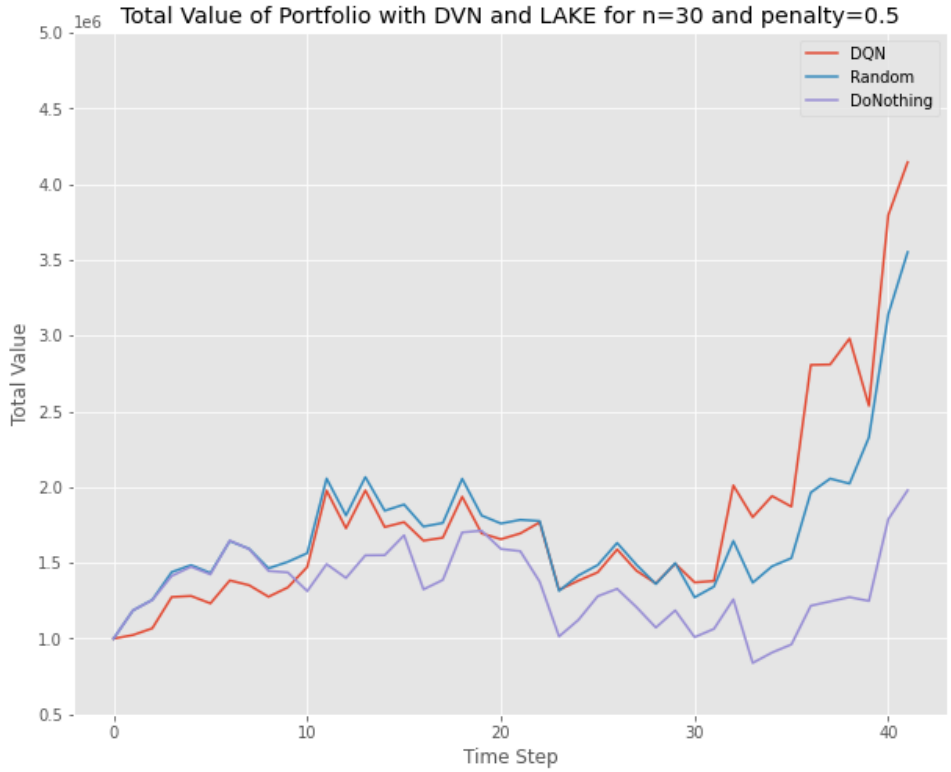
وزنهای نتورک هدف با مقدار τ کوچک به صورت زیر آپدیت می‌شوند:

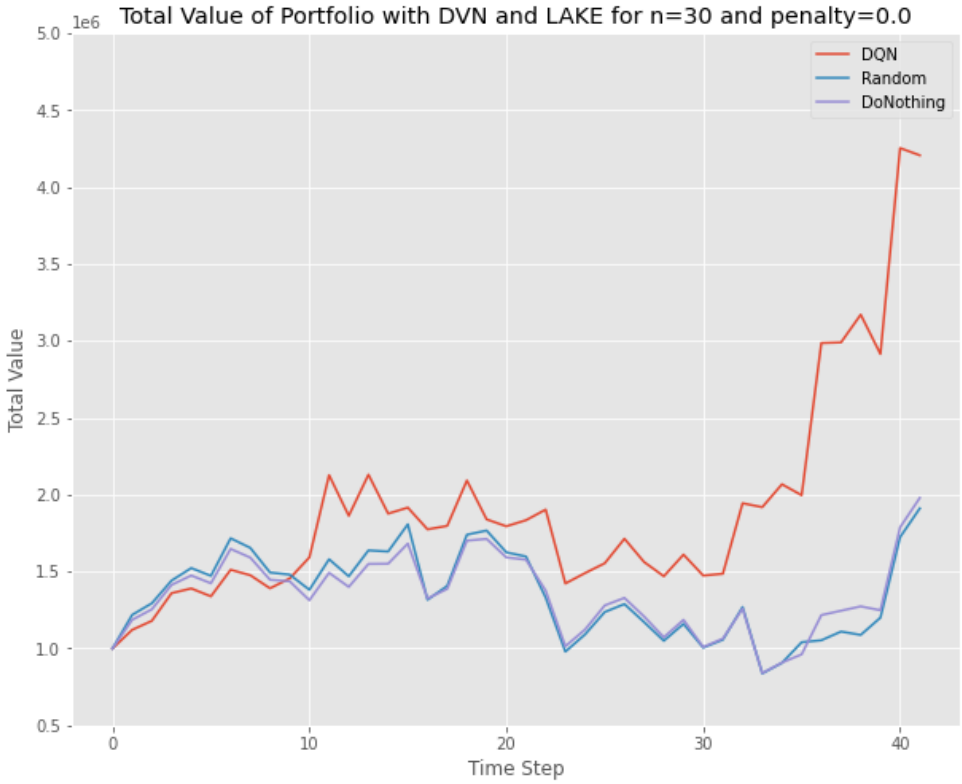
$$\theta_{Target} = (1 - \tau)\theta_{Target} + \tau\theta_{Main}$$

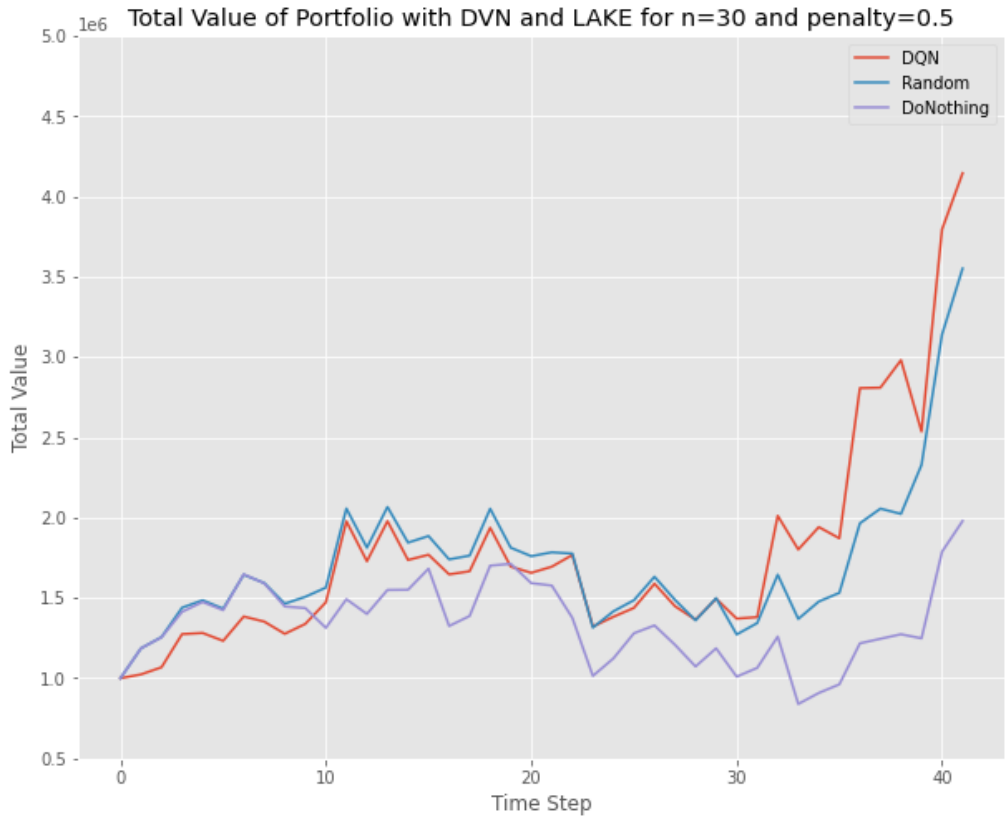
نتایج

معیار ارزیابی ما برای عملکرد مدل مقایسه سرمایه بازگشتی حاصل از استراتژی مدل و دو مدل مرجع (یکی اکشن‌های رندم در هر مرحله و دیگری عدم انجام هیچ معامله‌ای در بازه زمانی مورد بررسی) است. در زیر تعدادی از نمودارهای بازگشت سرمایه برای یکی از پرتفولیوهای حاصل از استراتژی این سه مدل داده شده است: (این نمودارها برای دو مقدار n و دومقدار reward penalty کشیده شده‌اند.)




Total Value of Portfolio with DVN and LAKE for n=30 and penalty=0.5
1e6
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
Total Value
0
10
20
30
40
Time Step
DQN
Random
DoNothing


Total Value of Portfolio with DVN and LAKE for n=30 and penalty=0.0
1e6
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
Total Value
0
10
20
30
40
Time Step
DQN
Random
DoNothing



جمع بندی و نتیجه گیری

با توجه به نمودارهای بازگشت واضح است که در اکثر موارد استراتژی DQN از رندم و عدم معامله در طول بازه بهتر عمل کرده است. البته این موضوع همواره صادق نیست که می‌تواند نتیجه عدم وجود دیتای کافی برای درک کردن یک سری از الگوهای موجود در داده تست باشد. به طور کلی این روش می‌تواند برای تعداد بیشتر سهم موجود در پورتفولیو و فضای عمل پیوسته تعمیم داده شود. البته در این حالت باید از روش‌های دیگری مانند الگوریتم DDPG که برای فضای عمل‌های پیوسته کاربرد دارد استفاده کنیم.

مراجع

[1] Chen, Kai, Yi Zhou, and Fangyan Dai. "A LSTM-based method for stock returns prediction: A case study of China stock market." *2015 IEEE international conference on big data (big data)*. IEEE, 2015.

[2] Nelson, David MQ, Adriano CM Pereira, and Renato A. de Oliveira. "Stock market's price movement prediction with LSTM neural networks." 2017 International joint conference on neural networks (IJCNN). IEEE, 2017.

[3] Gao, Ziming, et al. "Application of deep q-network in portfolio management." 2020 5th IEEE International Conference on Big Data Analytics (ICBDA). IEEE, 2020.

[4] Jin, Olivier, and Hamza El-Saawy. "Portfolio management using reinforcement learning." Stanford University (2016).

[5] Yue, Q. I. "Portfolio management based on DDPG algorithm of deep reinforcement learning." Computer and Modernization 05 (2018): 93.

[6] Lin, Fang, et al. "A DDPG Algorithm for Portfolio Management." 2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES). IEEE, 2020.

[7] https://en.wikipedia.org/wiki/S%26P_500\

[8] https://finance.yahoo.com/