# Double Deep Q-Learning for Optimal Execution


پروژه درس ریاضیات مالی

مهدی حلاجیان

۹۹۲۰۴۸۹۲

# فهرست

## مسئله چیست؟

اجرای معاملات بهینه یکی از مسائل مهمی است که تقریباً تمام معامله‌گران با آن روبرو هستند. در اینجا، رویکرد بدون مدلی را اتخاذ کرده و یک مدل Deep Q-Learning را برای تخمین اقدامات بهینه یک معامله‌گر توسعه داده‌ایم. مدل متشکل از دو شبکه عصبی کاملاً متصل است که با استفاده از داده های بازار و تجربه های قبلی مدل آموزش داده شده است. خروجی مدل Q-value ای است که پاداش‌های آینده یک اقدام دلخواه را تخمین می‌زند. ما مدل خود را بر روی سهام اپل آموزش دادیم و متوجه شده‌ایم که با استفاده از معیارهای عملکرد متوسط و میانگین و نسبت سود به زیان، نتایج بهتر از روش پایه استاندارد می باشد.

# فرضیات

LOB همان لیست سفارشات بازار می باشد

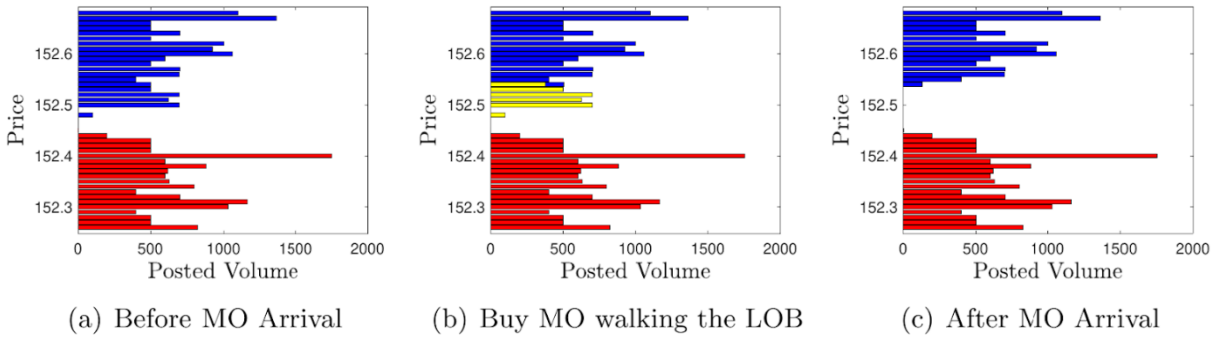



*Figure ۱-۰ LOB به معنای لیست سفارشات بازار می باشد*

در این مقاله فرض شده است که تفاوت حداکثر قیمت خرید و حداقل قیمت فروش کم می باشد. پس می‌توان فرض کرد که قیمت هر سهمی برابر با قیمت میانگین(میانگین قیمت خرید و قیمت فروش آن سهم) می‌باشد. همچنین فرض شده است که انجام معاملات تغییری در رفتار بازار ایجاد نخواهد کرد. به منظور مدل کردن تاثیر معامله کردن در رفتار بازار از یک تابع penalty استفاده شده است که همیشه به نسبت حجم معامله ای که انجامشده است اعمال می‌گردد.در بخش های بعدی به صورت مبسوط این تابع تعریف شده و تاثیر گذاری آن در نتیجه ی نهایی نشان داده می شود.

# مسئله در RL چگونه بیان می‌شود؟

## عملکرد بهینه:

ابتدا بیان خوب است که مسئله ی optimal execution را به فرم ریاضیاتی در بیاوریم. این مسئله به صورت زیر مدل می‌شود:

$$\underset{x_0,x_1,\ldots,x_{t-1}}{\operatorname{argmax}} \ \mathbb{E}\left[\sum_{t=0}^{T-1} R(p_t, x_t)\right]$$

در این مسئله ما به دنبال یافتن مسیر بهینه برای فروش تعداد مشخصی سهام در زمانی مشخص و متناهی هستیم. همان گونه که از فرمول بر می‌آید می بینیم که به دنبال این هستیم تا $x_0, x_1, \ldots, x_{t-1}$ (میزان فروش سهام در هر برهه ی زمانی) را به گونه ای تعیین کنیم که به حداکثر امید ممکن از مجموع سود برداشت شده از فروش سهام برسیم.

## تعریف تابع پاداش:

پس منطقی می باشد که تابع پاداش را به گونه ای طراحی کنیم که اندازه ی آن ارتباط مستقیمی با اندازه‌ی سود بدست آمده از فروش سهام ها باشد. تابع زیر تابعی می باشد که ما در حل این مسئله از آن استفاده کردیم.

$$Q(s,x) = \mathbb{E}\left[R(s,x) + \sum_{i=t+1}^{T} \gamma^{i-t} R(s_i^{\pi}, x_i^{\pi})\right]$$

المان اول این تابع همان تابع سود در لحظه می باشد. المان دوم این تابع نیز مجموع تنزیل شده ی سودی می باشد که مدل با انجام استراتژی بهینه اش بدست می آورد(المان دوم را به صورت تنزیل شده در تابع خود می آوریم تا تاثیر پاداش های آنی و با قطعیت بیشتر را بیش از پاداش های با فاصله ی زمانی و همراه با عدم قطعیت بیشتر بکنیم.

این تابع در دنیای واقعی باید به صورت احتمالاتی محاسبه گدد. از آن جایی که ما قطعیتی بر روی پاداش هایی که از استراتژی بهینه بدست خواهند آمد نداریم پس لازم می باشد که از فرم احتمالاتی فرمول بالا استفاده کرده و امید ریاضی تابع پاداش را محسابه کنیم.

ما تابع پاداش ارائه شده در بالا را به عنوان q-function در پیاده سازی خود استفاده کردیم و با استفاده‌ی از آن به تصمیم گیری بر روی اکشن هایی که انجام خواهیم داد پرداختیم.

## Double Deep Q-Learning:

استفاده ی از روش های معمول برای نگه داشتن جدول تمام حالات q-function بسیار پر هزینه و کند می باشد. از آن جایی که تعداد حالاتی که می تواند مسئله داشته باشد بسیار زیاد می باشد لازم است تا از مدلی استفاده گردد تا با پیچیدگی کمتر و سرعت بالاتر به پیش‌بینی مقدار q-function برای هر جفت

(state, action) پردازد. در این پیاده سازی ما از شبکه های عصبی عمیق برای پیش بینی و تخمین q-function استفاده کرده ایم.

به علاوه به منظور کمتر کردن overestimation در مدل از ۲ شبکه ی عصبی جدا استفاده کرده ایم. که یکی از آنها بهترین اکشن در هر استپ را پیشنهاد می کند و دیگری q-function را باتوجه به استیت در لحظه و پیش بینی که از آینده دارد تخمین می‌زند. دلیل جدا کردن این دو مدل از هم کم کردن بایاس مثبت مدل تخمین زننده ی q-function می باشد. در صورتی که از یک مدل هم برای تخمین q-function و هم برای پیش‌بینی بهترین اکشن استفاده گردد، تخمین مدل از q-function بایاس مثبت خواهد داشت چون که مدل دقیقا همان اکشنی را انجام میدهد که در راستای تخمینش از q-function می‌باشد و به همین دلیل این دو با یکدیگر هم راستا شده و با این روش ممکن است نقاط بهینه ی محلی گیر کنیم.

## مسئله را در reinforcement learning چگونه مدل سازی کردیم؟

### استیت مسئله:

- **وضعیت LOB در حال حاضر به علاوه ی هرگونه اطلاعاتی در مورد وضعیت دوره های گذشته**: اصلی ترین استیتی که بر روی مدل ما تاثیر گذار خواهد بود وضعیت بازار در لحظه ی حال حاضر و اطلاعات گذشته ی بازار می باشد.
- **زمان حال و باقیمانده ی لیست خرید**: هر چه زمان به جلو میرود و به ددلاین فروش سهم ها نزدیک تر می شویم، طبیعی است که مدل با سرعت و ریسک بالاتری دست به فروش سهم ها بزند. به همین دلیل منطقی می باشد که زمان را به عنوان استیت به مدل داده تا با توجه به آن بتواند استراتژی مناسب را در پیش گیرد. به طور مشابه با داشتن مقدار باقیمانده از لیست خرید، مدل می تواند بسته به شرایط استراتژی خود را تغییر داده و ریسک بیشتر یا کمتری را متحمل بشود.

### اکشن ها:

اکشن اصلی مدل مقدار سهم هایی که در هر بازه ی زمانی می خواهد بفروشد می باشد. به منظور ساده سازی مسئله فرض می شود که مدل فقط مقادیر طبیعی از سهام ها را در هر مرحله می تواند به فروش برساند.

### تابع پاداش:

تابع پاداش به صورت زیر تعریف می گردد. در هر ثانیه پاداشی گه مدل دریافت می کند برابر است با اختلاف قیمت سهام با ثانیه ی قبلی ضرب در تعداد سهم های فروخته شده در آن ثانیه، منهای پنالتی ای که برای هر معامله در نظر گرفته می شود. تابع پنالتی برابر است با یک ضریب ثابت(در پروژه بر اساس نتایج مقاله ۰/۰۱ در نظر گرفته شده است) در تعداد سهم های فروخته شده در این ثانیه به توان دو.($x_{T_k}$ تعداد سهم های فروخته شده در دوره و $M_k$ اندازه ی زمانی دوره به ثانیه می باشد) این پنالتی به منظور مدل کردن تاثیر گذاری ایجنت معامله کننده در بازار و مدل کردن خطا تقریب قیمت ها با قیمت میانگین می باشد.

$$\check{R}_{k,i} = q_{t_{k,i}} \left( p_{t_{k,i+1}} - p_{t_{k,i}} \right) - a \left( \frac{x_{T_k}}{M_k} \right)^2$$

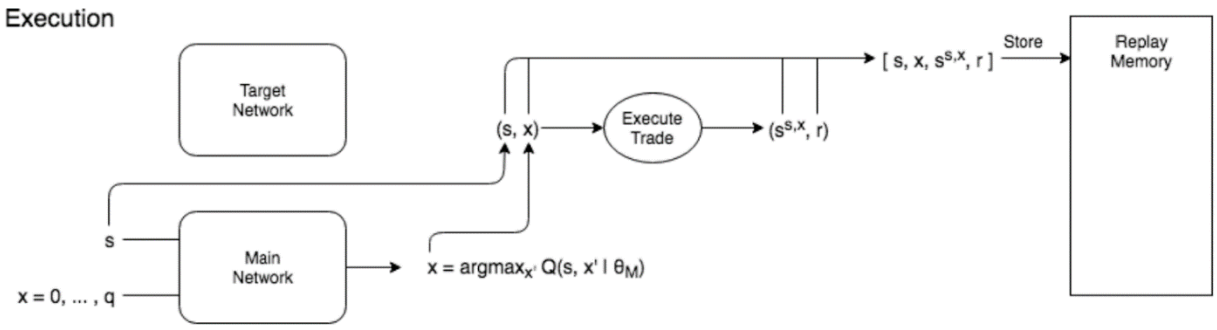
$$R_k = \sum_{i=0}^{M_k - 1} \check{R}_{k,i} \, ,$$

همچنین پاداش هر دوره برابر با مجموع پاداش های دریافت شده در طول دوره می باشد.

# مدل چگونه کار می کند؟

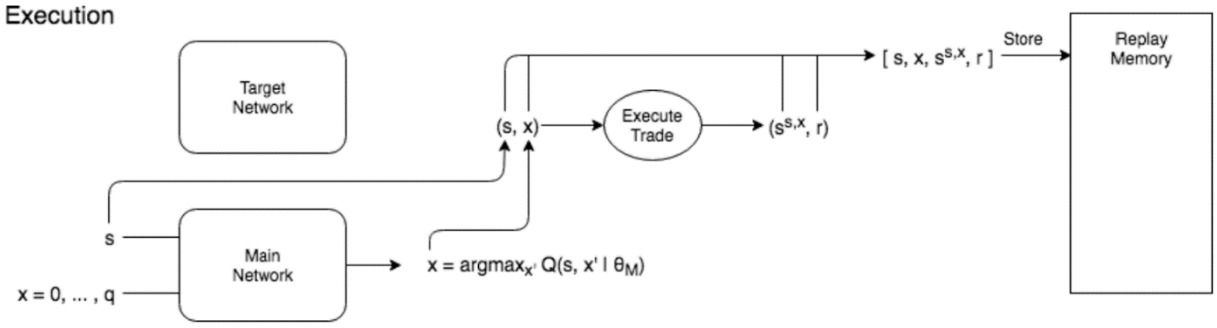
## معماری مدل:

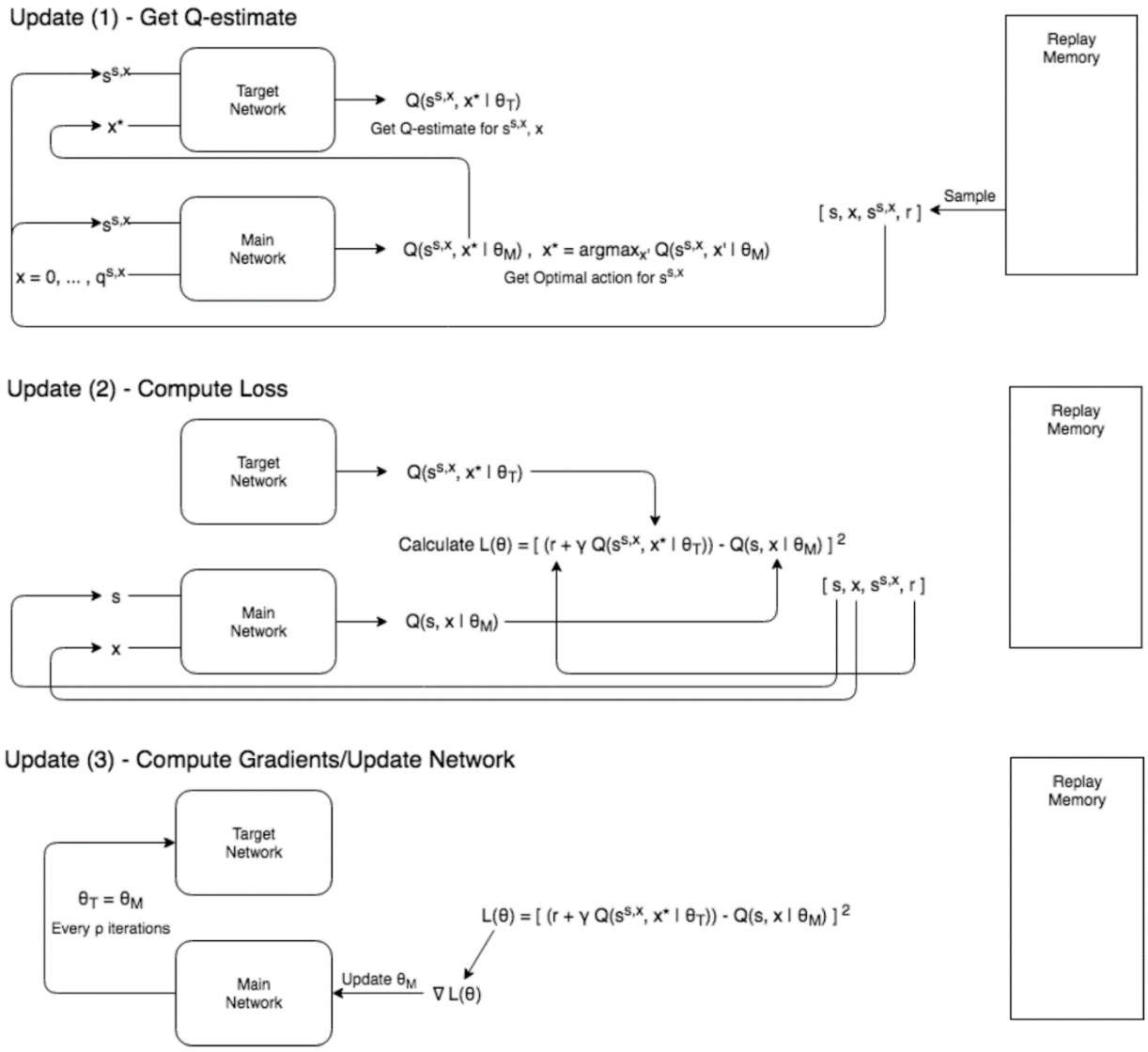
همان گونه که در بخش های قبل اشاره گردید به منظور کمتر کردن overstimulation.در تخمین q-function در این پروژه از دو مدل شبکه عصبی عمیق جدا از هم استفاده شده است. یکی برای تخمین q-function و دیگری برای انتخاب بهترین اکشن است. به مدل تخمین زننده ی q-function مسئله مدل هدف (target network) و به مدلی که بهترین اکشن را بر می گزیند، مدل اصلی(main network) می‌گوییم.




در مرحله ی اول مدل اصلی به پیش بینی بهترین اکشن ممکن با توجه به q-function ای که با استفاده از مدل هدف ساخته شده است می پردازد. سپس در مرحله ی بعد این اکشن را انجام داده و استیت و پاداش بدست آمده از انجام این اکشن را در حافظه ذخیره می کند.

ما مدل اصلی را در پایان بازه ی زمانی و پس از پایان یافتن انجام هر اکشن آپدیت می کنیم. مدل هدف اما هر بار آپدیت نشده و ئس از تعداد مشخصی دوره که گذشت آپدیت میگردد. این مقدار بسته به شرایط مسئله قابل تغییر می باشد و مقدار بهینه ی آن به صورت تجربی به دست می آید.






## استراتژی epsilon-greedy

در صورتی که مدل تنها با استفاده از فرضیات و اطلاعات قبلی خود به کار در محیط بپردازد همیشه رو به بهبود خواهد رفت. اما ممکن است این بهبود یک اکسترمم مجلی باشد و در صورتی که مدل از بایاس های اولیه اش خارج گردد تواند یک اکسترمم بهینه تر را پیدا کند. به همین منظور این استراتژی را دنبال می کنیم که با احتمال اپسیلون، مدل حرکت رندومی انجام دهد تا محیط های دیده نشده و کشف نشده را نیز در طر زمان با این روش تجربه کند. حال اگر تجربه ی جدید بهبود در عملکرد ایجنت ایجاد کرد مدل آن را پذیرفه و به عنوان جزئی از استراتژی پس از آن استفاده خواهد کرد. در غیر این صورت مدل به بایاس های اولیه اش باز گشته و استراتژی قبلی ای که برگزیده بود را ادامه خواهد داد. به این روش برقراری تعادل بین exploration vs exploitation روش epsilon-greedy می گوشیند. به این معنا که مدل در عموم حالات به صورت greedyعمل کرده و استراتژی اولیه اش را پیش می برد ولی با احتمال اپسیلون اکشن رندوم انجام خواهد داد و به کشف محیط با امید یافتن روش ها و متد های بهتر خواهد پرداخت.

# داده های پروژه

ما در این پروژه از داده های سهام شرکت اپل در بازه ی ۲۲ ژانویه ی سال ۲۰۱۹ الی ۲ آگوست سال ۲۰۱۹ استفاده کرده ایم.

در خود مقاله ی پروژه از داده های چندین شرکت دیگر و با بازه ی زمانی بزرگتر استفاده شده بود. در این پروژه، به دلیل سختی دسترسی به داده های خارج از ایران، به همین داده های بدست آمده از سهام شرکت اپل کفایت شده است.

# نتایج

## معیار بررسی نتایج:

در این پژوهش از معیار P&L(profit and loss) برای بررسی و مقایسه ی عملکرد مدل استفاده کرده‌ایم. نحوه ی محاسبه ی این معیار از روش زیر می باشد.

$$P\&L_b = \sum_{k=0}^{N-1} \sum_{i=0}^{M_k-1} \left\{ x_{t_{k,i}} p_{t_{k,i}} - a \left( \frac{x_{T_k}}{M_k} \right)^2 \right\}$$

همان گونه که از فرمول بالا بر می آید، ما پنالتی را هم در معیار P&L لحاظ کرده ایم تا داده های بدست آمده از این معیار مقایسه پذیرتر و واقعی تر باشند.

همچنین مدل معیار برای مقایسه ی عملکرد مدل روش TWAP(time weighted average price) می باشد. در این روش خرید به صورت یکنواخت بین بازه های زمانی تقسیم شده و در نهایت قیمت بدست آمده از فروش سهام برابر با قیمت میانگین آن در آن بازه ی زمانی می‌شود.

## مقایسه ی P&L مدل آموزش دیده با مدل ساده ی خرید با میانگین وزن دار

در پایان دوره P&L بدست آمده ی مدل آموزش دیده برابر با ۲۲۷ واحد گردید در حالی که P&L بدست آمده از مدل TWAP برابر با ۲۰۷ می باشد.(از آن جایی که ورودی‌های مسئله بدون واحد شده اند پس این مقدار معنای واقعی دقیقی ندارد و صرفا یک معیار مقایسه می باشد.) اختلاف تقریبا ۱۰ درصدی ما بین این مدل نشان دهنده ی این می باشد که مدل آموزش دیده بهتر از یک ایجنت یونیفرم عمل خواهد کرد که نشان دهنده ی هوشمندی مدل در بکار گیری اطلاعات بازار در ئیشبینی رفتار آن می باشد.

در ادامه این بخش به منظور بررسی بیشتر نمودار های P&L ایجنت آموزش دیده و TWAP نسبت و زمان و هیستوگرام هایشان آورده شده است.

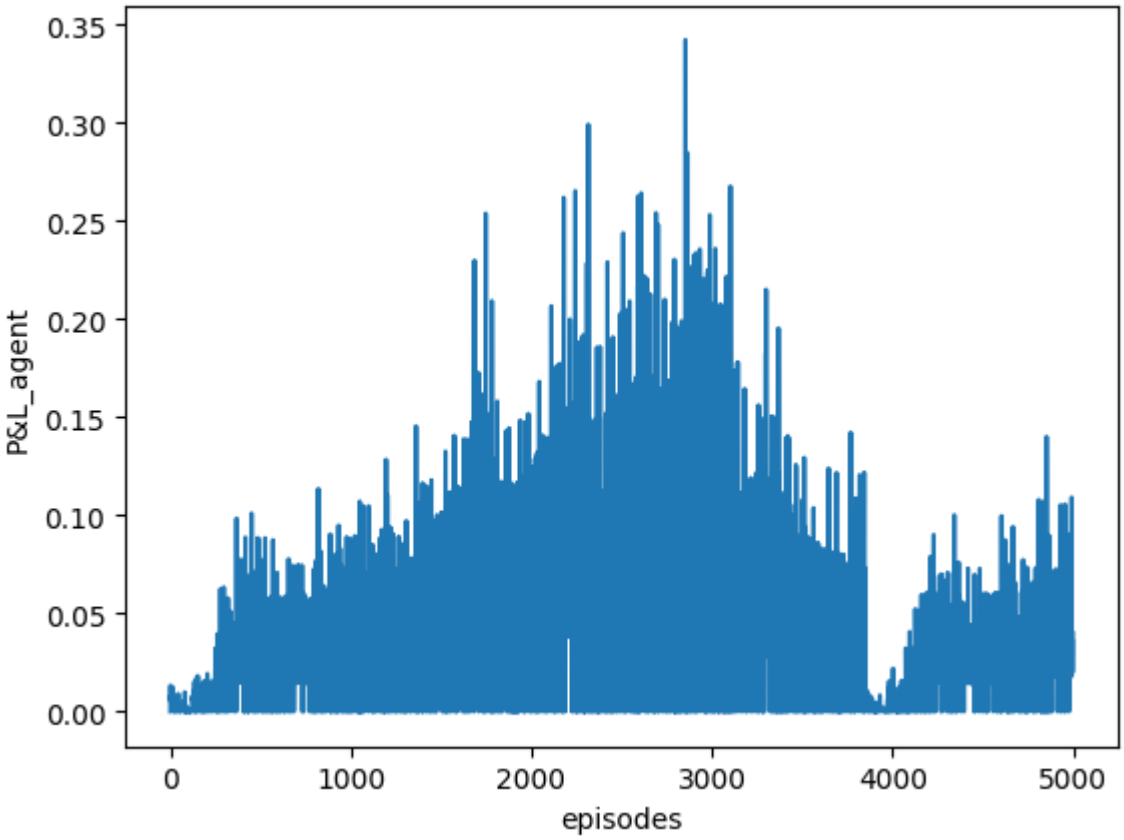



*Figure ۱-۰ P&L مدل اموزش دیده در طی زمان*

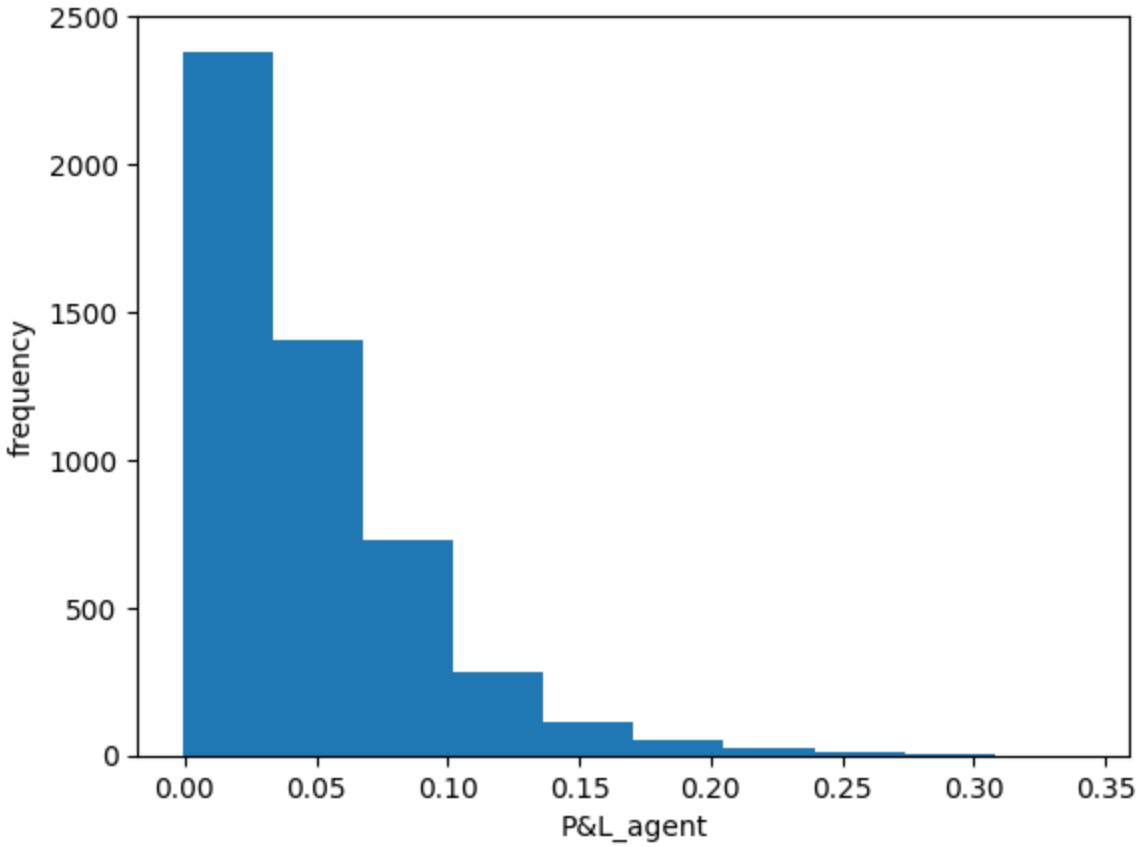



*Figure ۲-۰ هیستوگرام P&L مدل آموزش دیده*

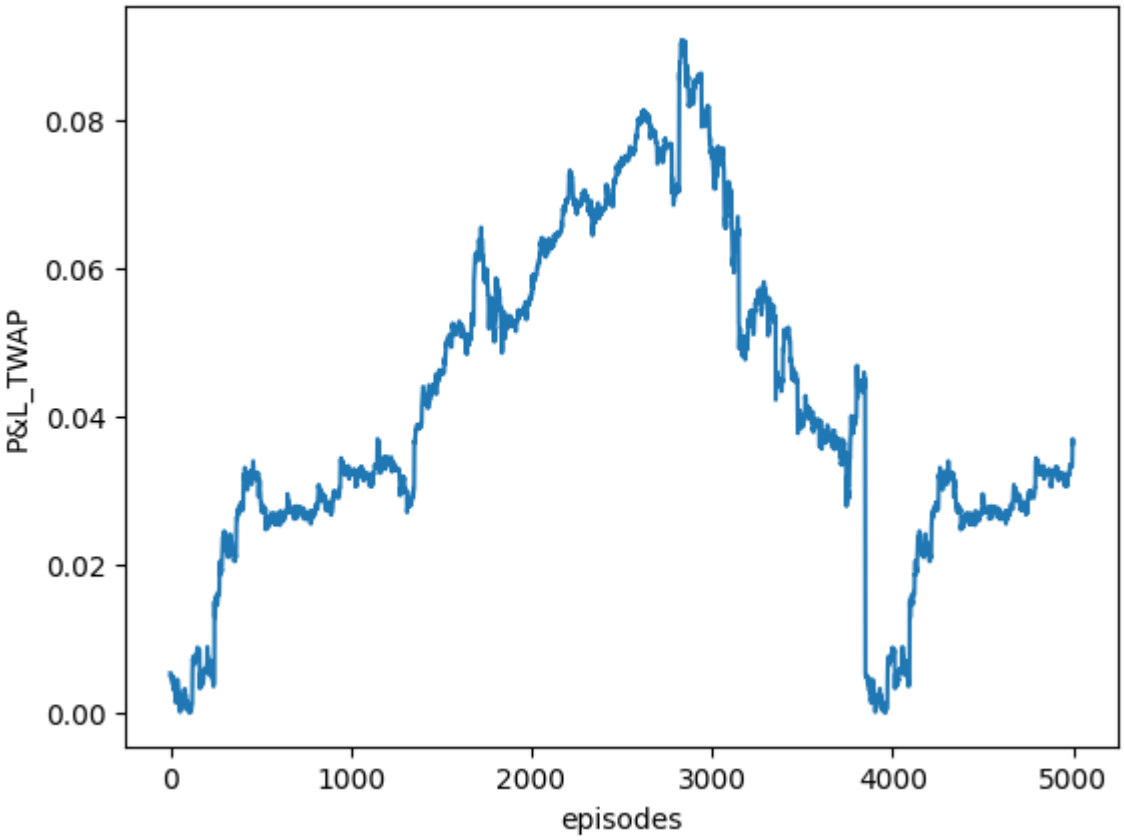



*Figure ۳-۰ P&L مدل TWAP در طی زمان*

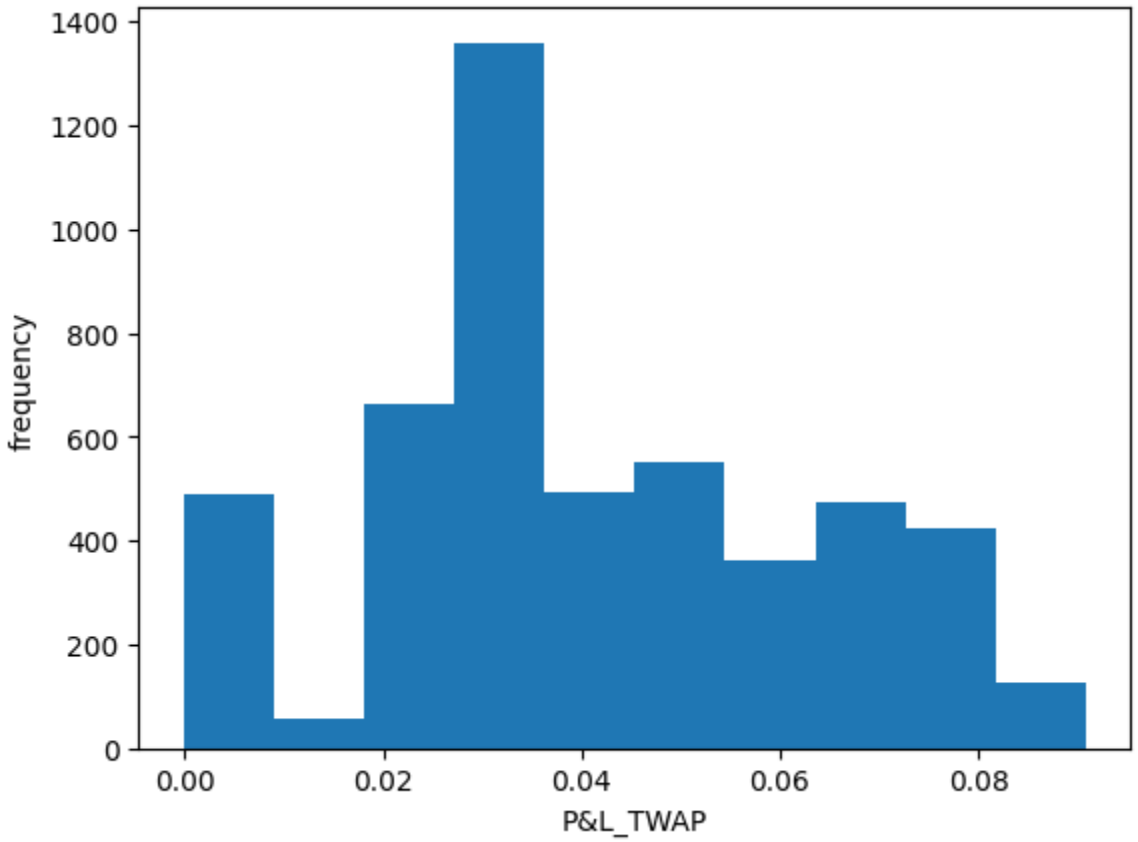



*Figure ۴-۰ هیستوگرام P&L مدل TWAP*

همان گونه که از نمودار ها پیداست مدل آموزش دیده نوسان بسیار زیادی داشته است. این موضوع چند دلیل می تواند داشته باشد.

۱. یک مسئله ی اصلی این می باشد که بنده در پیاده سازی بخش حافظه ی جدا برای مدل در این پروژه با مشکل روبه رو شدم. به همین دلیل مدل صرفا به حافظه ی چند حرکت قبلش فقط دسترسی خواهد داشت و نمی تواند به صورت رندوم از نتایجی که در طی زمان بدست آورده بهره ببرد. این مسئله خلاف روش پیشنهادی در مقاله می باشد و به تصور بنده بخشی از نوسانات مدل می توانست با پیاده سازی درست این بخش گرفته شود.
۲. محدودیت داده مسئله ی دیگری می باشد که باعث میگردد تا آموزش مدل به صورت کامل صورت نگیرد و اصطلاحا converge نکند. با داشتن داده های بیشتر و بزرگ تر کردن استخر داده های آموزش مدل نیز بهتر آموزش خواهد دید و از نتایج نوسانی آن کاسته خواهد شد.

# نتیجه گیری و کار های پیش رو

در این مقاله، مسئله اجرای بهینه را به عنوان یک مسئله یادگیری تقویتی فرموله کردیم. یک تکنیک یادگیری تقویتی عمیق توسعه دادیم و آن را بر روی داده های سهام شرکت اپل آموزش دادیم. در نهایت نیز نتایج نشان می دهد که این روش در فروش بهینه ی سهام شرکت اپل ۹ درصد بهتر از روش TWAP عمل کرده است که عدد قابل توجهی می باشد.

البته چندین مسیر برای بررسی و تحقیق وجود دارد. دو مسیر دیگر افزایش تعداد دارایی های تحلیل شده و شامل تعدادی ویژگی اضافی، مانند تاریخچه قیمت و تاریخچه سفارشات محدود و غیره است. جهت دیگر، ترکیب روش های تحلیلی با Q-learning عمیق با استفاده از استراتژی معاملات بهینه تحلیلی به عنوان نقطه شروع برای یادگیری تقویت شده است.

# مراجع

Ning, B., Lin, F. H. T., & Jaimungal, S. (2020). Double Deep Q-Learning for Optimal Execution.