Adversarial Attacks on Deep Algorithmic Trading Policies

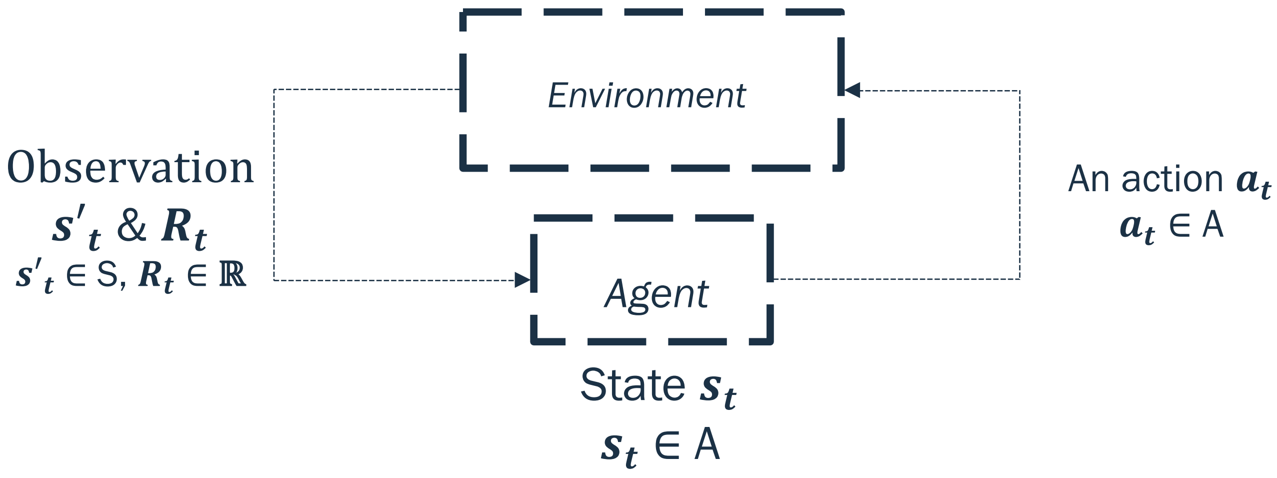
Piazza, Nancirose, Yaser Faghan, Vahid Behzadan, Ali Fathi

Secure and Assured Intelligence Learning (SAIL)

University of New Haven

Reinforcement Learning

Reinforcement Learning (RL) is learning to interact with an environment through experience (trial and error).



A: action space
S: state space
$\{\_\}_t$: at timestep t
s': successor state
R_t: reward R(s, a, s')
P: probabilistic state transition P($s'|s, a$)
γ: discount factor [0,1]

- Markov Decision Process: discrete-time, stochastic decision-making process/framework

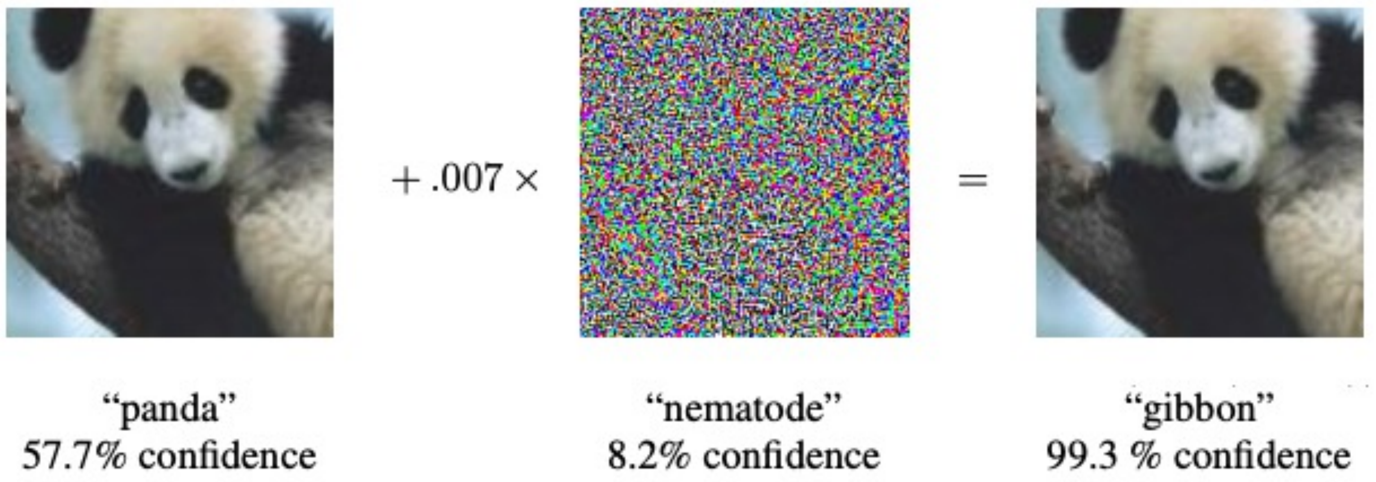
- End Goal: Find an optimal policy (a mapping from states to actions) which maximizes the expected total sum of discounted rewards.

Why Deep Reinforcement Learning in Trading?

	Interest to Traders?	Immediate Problems?
High frequency trading where there is the automation of large volumes and fast intervals of trading.	☒	How? Through RL.
■ Reinforcement Learning (RL) – *Uses the Markov Decision Process (MDP) which is a discrete-time, stochastic control process. MDP is a mathematical framework for decision-making with some assumptions.*	☒	But RL only works for discrete state table? Use function approximator.
■ Deep Learning’s Neural Networks (NN) – *Ability to feature engineer high dimensional data* – *Generalization*	☒	We’ll get to it.

Adversarial Example

- Deep Architectures are known to be susceptible to adversarial examples.
- Does this apply to DRL? Yes → Does this apply to DRL trading agents? ...



(Goodfellow, 2014)

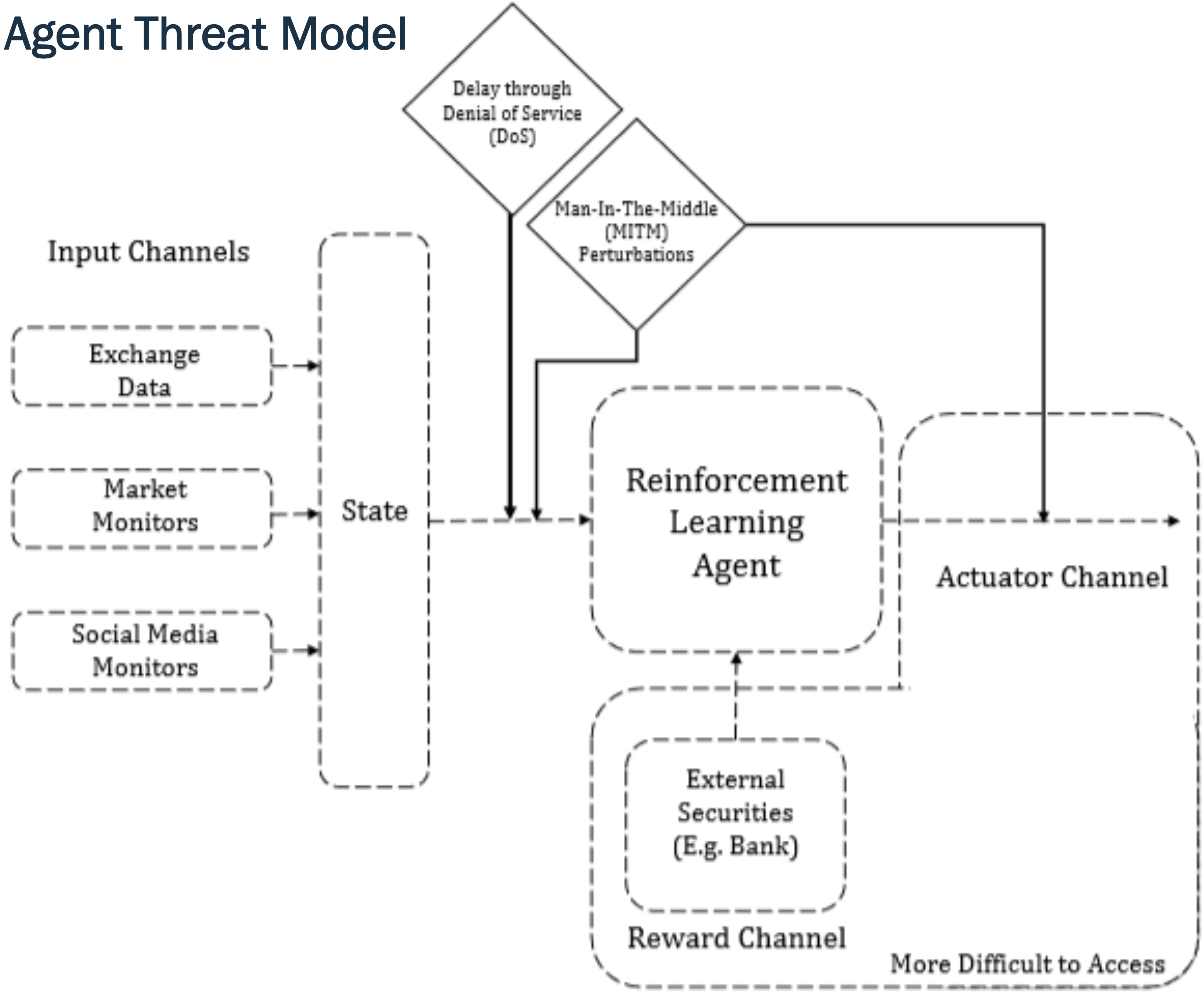
- What are an adversary’s intentions? Why? How? We threat model it.

Adversarial Objective

- Well known in Computer Security:
  - ***Confidentiality, Integrity, Availability (CIA)***

- An adversary will aim to violate:
  - ***Confidentiality** of the model*
    - Intellectual property. Trading DRL agents are expensive to train.
  - *Privacy of training or testing data*
    - Balance, PID, History?
  - ***Integrity** of the predictions*
    - Can the model be trusted to make decisions for your benefit?
  - ***Availability** of the agent or the system hosting the agent*
    - No trading means losing value

DRL Trading Agent Threat Model



DRL Trading Agent Threat Model

- Adversarial attacks are inevitable but using a trading DRL threat model can outline channel attacks such that proper preparations can mitigate damages.
- The attacks shown here are performed as test-time attacks, but they are not exclusively test-time. The reward channel and actuator channel may also be attacked.
- Input channels may be individually attacked, but we target the observational channel.
- This trading threat model uses a DRL threat model framework proposed by Behzadan [3].
- An adversary budget is a measurement of an adversary’s resource to successfully perform an attack.

Attack Flavors

- There are targeted and non-targeted attacks
  - *Non-targeted – agent takes any other action than optimal action* $\boldsymbol{a_t}$ *at timestep* $\boldsymbol{t}$*.*
  - *Targeted - agent takes adversarial action* $\boldsymbol{a'_t}$ *instead of optimal action* $\boldsymbol{a_t}$ *at timestep* $\boldsymbol{t}$.
- There are whitebox and blackbox attacks
  - *Whitebox – Enough adversarial knowledge of the target agent to craft (optimization-based) adversarial attacks.*
  - *Blackbox – There is not enough adversarial knowledge to craft an attack but has direct/indirect accessibility to agent.*
- There are active and passive attacks
  - *Active – change an agent's trajectory.*
  - *Passive – gather adversary information on target agent.*
- There are test-time and train-time attacks
  - *Test-time – An attack on an agent that's using a fixed policy.*
  - *Train-time –An attack on an agent during its training phase.*

Observation (Feature) Space

- What can constitute the observation space/state space for a deep RL trading algorithm?
- In trading, there are time frequency intervals (e.g. milliseconds, minutes, hours, etc.) Each interval is called a **bar**. A bar may include:
  - *High Price*
  - *Open Price*
  - *Close Price*
  - *Low Price*
- Technical Indicators (TI) are financial calculations on historic values which are often used to forecast financial market direction.
  - *Moving Average Convergence Divergence (MACD)*
  - *Relative Strength Indicator (RSI)*
- Information can be public or private, but we only have access to public.

Reward Function

- Reward function is important, it determines the optimal policy. What can be used as a reward function?
- Profit/Loss (Basic DQN)
  - *Simple, but doesn't appeal to traders who desire risk-awareness.*
- Financial Metrics
  - *Sharpe Ratio (TT DQN)*
    - Performance in comparison to a risk-free investment.
    - More often used for low volatility investment profiles
  - *Sortino Ratio*
    - Variant of Sharpe Ratio
    - More often used for high-volatility investment profiles

$$S_a = \frac{E\left[R_a - R_b\right]}{\sigma_a}$$

S_a = Sharpe ratio

E = expected value

R_a = asset return

R_b = risk free return

σ_a = standard deviation of the asset excess return

Action Space

- Discrete
  - *Buy, Sell, Wait (for one stock)*
  - *Buy/Sell in interval quantities or intervals proportional to available stock*
  - *Cartesian product of finite quantities*
- Continuous
  - *A real interval*
    - [0,1] – percentage to buy of a single stock
    - Etc.
- Designer discretion

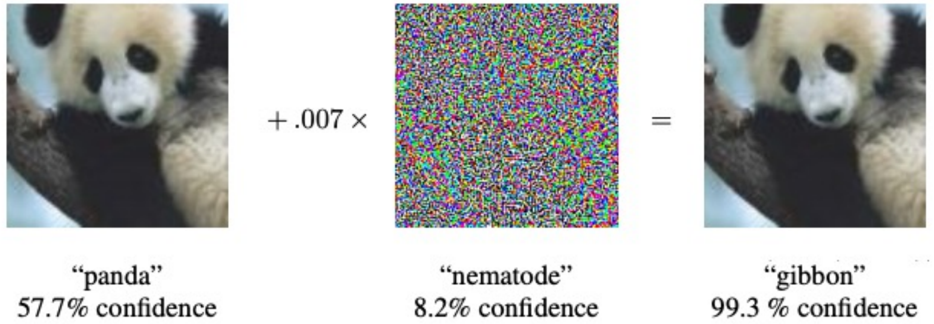
Our Investigated Agents

- Basic DQN
  - *Observation: Relative High/Low/Close to Open Price, [0 or 1] indicator of bought stock [size 4]*
  - *Reward: Profit/Loss*
  - *Action: discrete buy, sell, wait [size 3]*
  - *Window size: 10 historic observation tuples*
- TensorTrade (TT) DQN
  - *Obsevation: MACD, RSI, Difference in log of closing price for two consecutive timesteps [size 3]*
  - *Reward: Sharpe Ratio*
  - *Action: discrete buy, sell % based on owned quantity [size 180]; product of trade size, stop, take. Wait [size 1]*
  - *Window size: 20 historic observation tuples*
- Termination Condition: T > 250 timesteps

Active, Test-Time Attacks

- Attacking the Observation Channel – Adversarial (whitebox) Attacks like Fast Gradient Sign Method [1]. (FGSM), Carlini and Wager [2] (C&W) Attack.

FGSM example
Outside of RL:



(Goodfellow, 2014)

FGSM example
For Trading DRL
(Basic DQN):

(RHigh, RLow, Rclose)

E.g. relative high price = $\frac{HP - OP}{OP}$

where HP is high price and OP is open price.

t	x	x'	a	a'
894	0.0,-0.00354677, -0.00354677	0.0000, -0.0045, -0.0025	1	0
3973	0.0, -0.00048828,-0.00048828	0.0000, -0.0006, -0.0004	1	0
9599	0.00294118,-0.0004902,0.00294118	0.0027, -0.0002, 0.0027	0	1
16323	0.00435098,0.0, 0.00290065	0.0041, 0.0000, 0.0032	2	0
23283	0.00074322,-0.00371609,0.00074322	0.0001, -0.0044, 0.0001	0	1

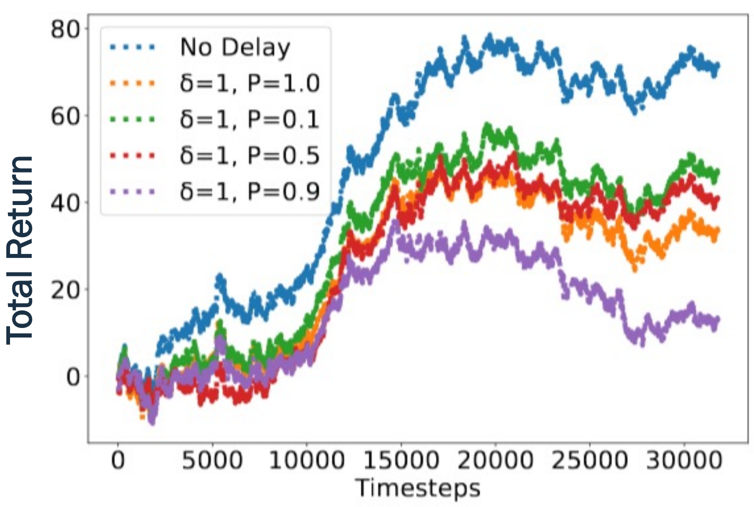
Table 5: Successful Basic DQN Non-Target FGSM Observations Samples

- We perturb the most recent tuple in its N-window. More will be explained later.

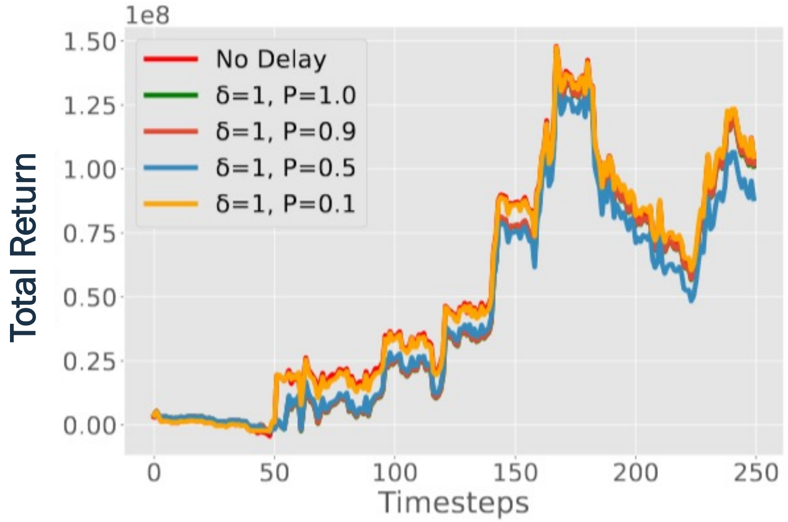
0 = Wait, 1 = Buy, 2 = Sell/Close Position

Observational Delay

- Observational Delay is an attack on the observation channel. A tuple originally received at timestep $\boldsymbol{t}$ is seen at timestep $\boldsymbol{t+1}$. Timestep of 1 is the minimal amount of possible delay.
- Adversary budget depends on how the delay is implemented. If there is 1 adversary intervention for N timesteps, budget is N. If adversary can induce the delay another way, it will have a different budget.



(a) Basic DQN



(b) TensorTrade DQN

Non-Targeted FGSM & Non-Targeted C&W Attack Samples

FGSM parameters

Basic DQN - started with ε = 0.0001, up to 5 attacks iterations with max ε = 0.001.

TensorTrade DQN - started with scalar ε = 0. 1, up to 5 attack iterations with max ε = 3.0 with $k_0 = 0.01, k_1 = 0.01, k_2 = 0.1$.

t	x	x'	a	a'
894	0.0,-0.00354677, -0.00354677	0.0000, -0.0045, -0.0025	1	0
3973	0.0, -0.00048828,-0.00048828	0.0000, -0.0006, -0.0004	1	0
9599	0.00294118,-0.0004902,0.00294118	0.0027, -0.0002, 0.0027	0	1
16323	0.00435098,0.0, 0.00290065	0.0041, 0.0000, 0.0032	2	0
23283	0.00074322,-0.00371609,0.00074322	0.0001, -0.0044, 0.0001	0	1

Table 5: Successful Basic DQN Non-Target FGSM Observations Samples

t	x	x'	a	a'
1602	0.00203314, 0.0, 0.00203314	0.0003, 0.0000, 0.0003	0	1
4735	0.00707071, 0.0,0.00707071	0.0002, 0.0000, 0.0002	0	1
5346	0.0032695 ,-0.00140121,0.0032695	0.0002, -0.0002, 0.0002	0	1
17424	0.0010985,-0.0010985 , 0.0010985	0.0002, -0.0002, 0.0002	2	0
29779	0.00039904,-0.00079808, 0.00039904	0.0003, -0.0003, 0.0003	0	1

Table 6: Successful Basic DQN Non-Target C&W Observations Samples

Non-Targeted FGSM & Non-Targeted C&W Attack Failure Attempts

Basic DQN – c = 0.1, 100 iterations.

TensorTrade DQN – c = 0.1, 100 iterations.

with $k_0 = 0.1, k_1 = 1.0, k_2 = 1.0$

FGSM				C & W			
Chance	No. Attempts	No. Failures	N.C.N	Chance	No. Attempts	No. Failures	N.C.N
0.1	26	25	2	0.1	17	17	0
0.5	123	117	7	0.5	114	110	3
1.0	242	236	7	1.0	246	240	3

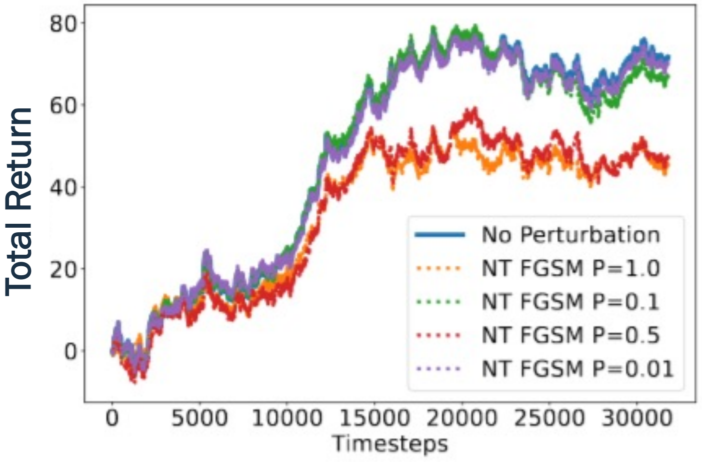
Table 9: Non-Target FGSM and C&W Attacks Attempts and Failures on TensorTrade's DQN

FGSM			C & W		
Chance	No. Attempts	No. Failures	Chance	No. Attempts	No. Failures
0.01	286	6	0.01	329	163
0.1	3349	176	0.1	3016	1751
0.5	15818	3329	0.5	15979	9358
1.0	31779	10778	1.0	31779	18716

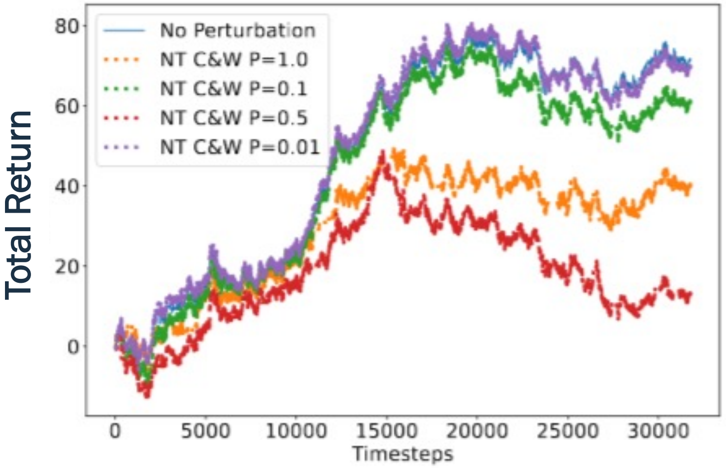
Table 10: Non-Target FGSM and C&W Attacks Attempts and Failures on the Basic DQN

Performance Impact

BasicDQN

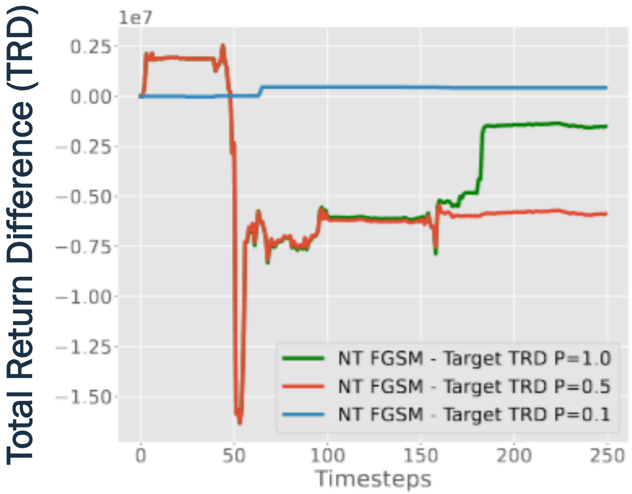


(a) Non-Targeted FGSM Attack

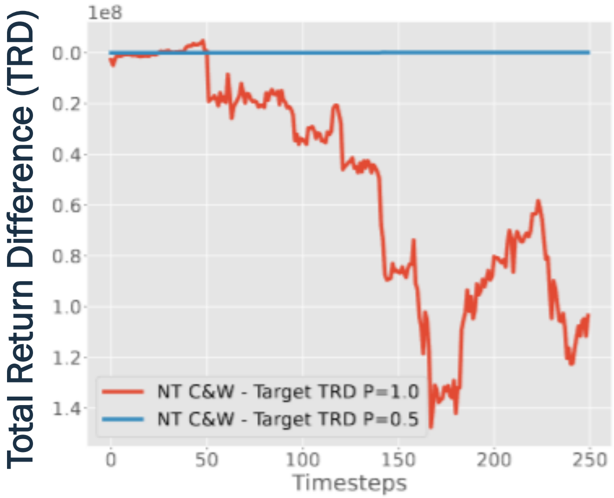


(b) Non-Targeted C&W Attack

TensorTrade DQN

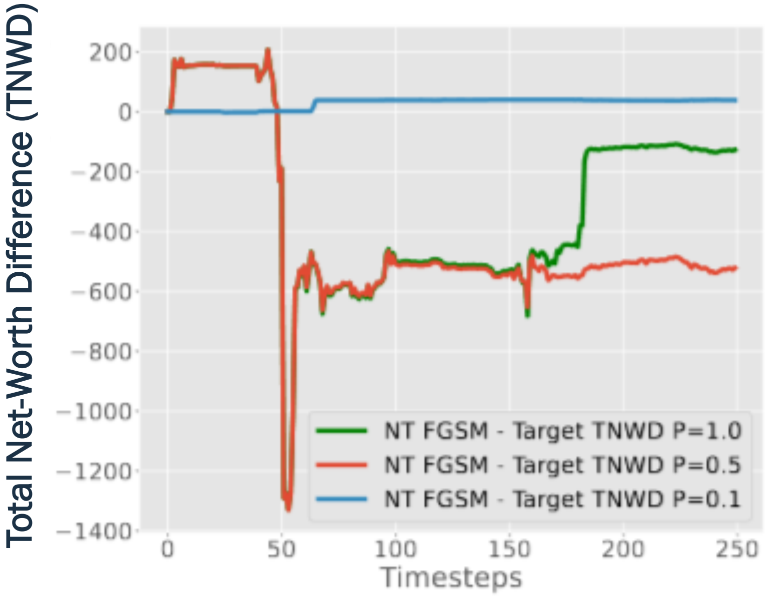


(a) Non-Targeted FGSM Attack Reward Difference

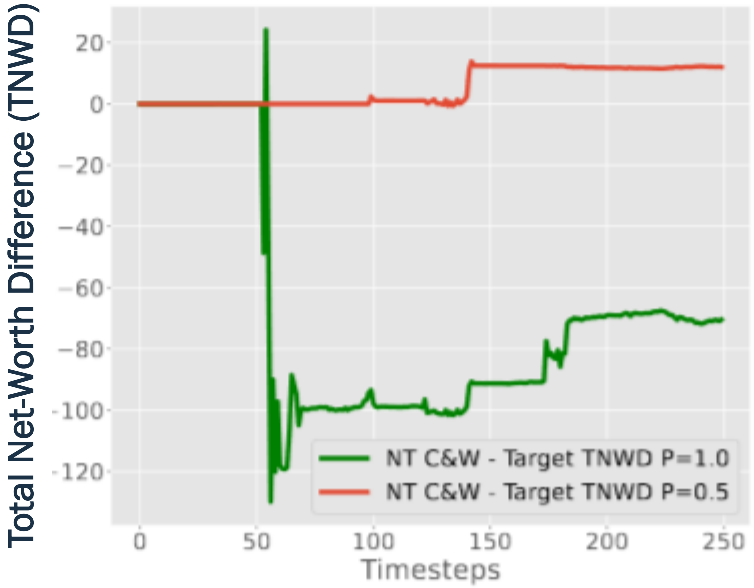


(b) Non-Targeted C&W Attack Reward Difference

Net-Worth Impact Non-Target - TensorTrade



(a) Non-Targeted FGSM Attack Total Net-Worth Difference



(b) Non-Targeted C&W Attack Total Net-Worth Difference

Targeted FGSM & Targeted C&W

- We can apply a direct Targeted C&W attack to Basic DQN like before with post constraints.
- Like prior with non-targeted C&W, we have the same setup but adjust failure to consider partial success.

Targeted FGSM & Targeted C&W Attack Failure Attempts

Basic DQN – c = 0.1, 100 iterations.

TensorTrade DQN – c = 0.1, 100 iterations.

with $k_0 = 0.1, k_1 = 1.0, k_2 = 1.0$

FGSM				C & W			
Chance	No. Attempts	No. Failures	No. Non-Target	Chance	No. Attempts	No. Failures	No. Non-Target
0.01	337	6	4	0.01	327	294	89
0.1	3148	191	98	0.1	3135	2915	903
0.5	15905	4666	1581	0.5	15882	15291	4837
1.0	31779	16000	5334	1.0	31779	30779	9953

Table 15: Targeted FGSM and C&W Attacks Attempts and Failures on Basic DQN

FGSM					C & W				
Chance	No. Attempts	No. Failures	Non-Targeted	P.S.	Chance	No. Attempts	No. Failures	Non-Targeted	P.S.
0.1	248	248	146	230	0.1	26	26	5	26
0.5	123	123	65	122	0.5	131	131	25	127
1.0	28	28	9	27	1.0	249	249	70	243

Table 16: Target FGSM and C&W Attacks Attempts and Failures on TensorTrade's DQN

Targeted FGSM & Targeted C&W Attack Samples

FGSM parameters

Basic DQN - started with ε = 0.0001, up to 5 attacks iterations with max ε = 0.001.

TensorTrade DQN - started with scalar ε = 0. 1, up to 5 attack iterations with max ε = 3.0 with $k_0 = 0.01, k_1 = 0.01, k_2 = 0.1$.

t	x	x'	a	a'	P.S. or S.
1	6.1583184e-03, 4.1991682e+00, 1.0000000e+02	3.444168e-02, 8.991680e-01, 1.009300e+02	0 (W)	144 (S)	P.S.
65	-4.5726676e-03, 1.6424809e+01, 6.1849476e+01	-1.4827332e-02, 2.5724810e+01, 6.2779472e+01	0 (W)	122 (S)	P.S.
138	-3.4072036e-03 -4.1683779e+00 5.6641838e+01	-1.5992796e-02 -3.7883778e+00 5.7571835e+01	0 (W)	96 (S)	P.S.
179	1.1997896e-06 -1.0627291e+01 6.5866417e+01	1.9401200e-02 -7.3272896e+00 6.6196411e+01	1 (B)	78 (S)	P.S.
249	6.7891073e-03 -1.1369240e+01 5.6471169e+01	1.2610892e-02 -2.0669241e+01 5.5541172e+01	1 (B)	46 (S)	P.S.

Table 13: Successful TensorTrade DQN Target FGSM Observations Samples

t	x	x'	a	a'	P.S. or S.
1	6.1583184e-03, 4.1991682e+00, 1.0000000e+02	1.61259882e-02, 5.19593525e+00, 1.00003235e+02	0 (W)	144 (S)	P.S.
2	3.119729e-03, 8.207591e+00, 1.000000e+02	1.30873993e-02, 9.20435810e+00, 1.00003235e+02	0 (W)	15 (B)	P.S.
189	-5.3039943e-03 -5.4235260e+01 4.5886791e+01	-4.6636765e-03 -5.3238495e+01 4.5890026e+01	0 (W)	78 (S)	P.S.
244	-5.3748242e-03, 8.7277918e+00, 5.8095055e+01	-4.5928466e-03, 9.7245588e+00, 5.8098289e+01	1 (B)	96 (S)	P.S.
247	-4.9919840e-03, -9.8234949e+00, 5.4668602e+01	-4.9756868e-03, -8.8267279e+00, 5.4671837e+01	0 (W)	15 (B)	P.S.

Table 14: Successful TensorTrade DQN Target C&W Observations Samples

t	x	x'	a	a'
301	0. , -0.0048627,-0.00243129	0.0000, -0.0040, -0.0033	0	1
5254	0.00094877,-0.00332068,-0.00142315	0.0016, -0.0027, -0.0021	0	1
12228	0.00037272,-0.00260902,-0.00111815	0.0012, -0.0018, -0.0018	2	1
21009	0.0,-0.0027894, -0.00209205	0.0000, -0.0025, -0.0024	0	1
24764	0.00119332,-0.00357995,-0.00159109	0.0018, -0.0029, -0.0022	0	1

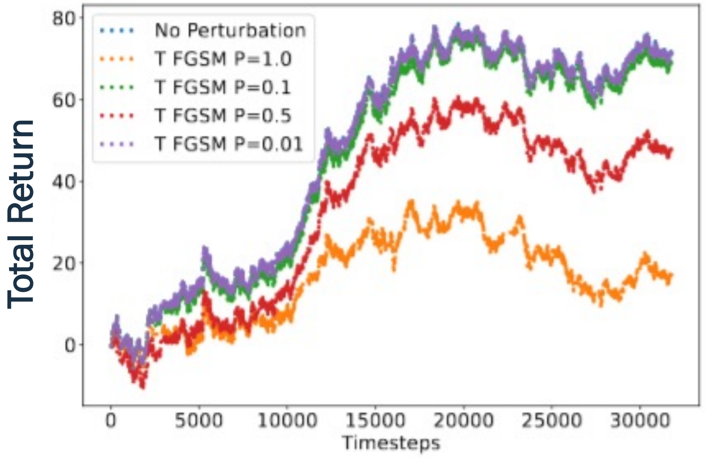
Table 11: Successful Basic DQN Target FGSM Observations Samples

t	x	x'	a	a'
2233	0.00490773,0.0, 0.00490773	0.0003, 0.0000, 0.0003	0	1
11328	0.00041408,-0.00248447,0.00041408	0.0003, -0.0003, 0.0003	0	1
17733	0.00362319,-0.00072464, 0.00362319	0.0003, -0.0003, 0.0003	0	1
20145	0.00102881,0.0,0.00102881	0.0002, 0.0000, 0.0002	0	1
26787	0.00569106, 0.0, 0.00569106	0.0003, 0.0000, 0.0003	2	1

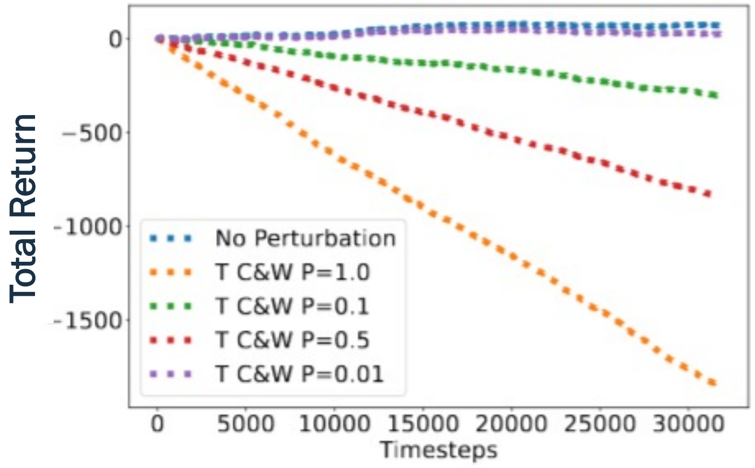
Table 12: Successful Basic DQN Target C&W Observations Samples

Performance Impact

BasicDQN

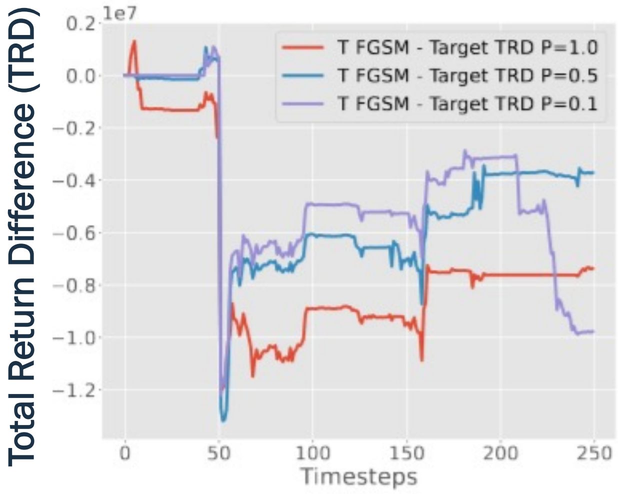


(a) Targeted FGSM Attack

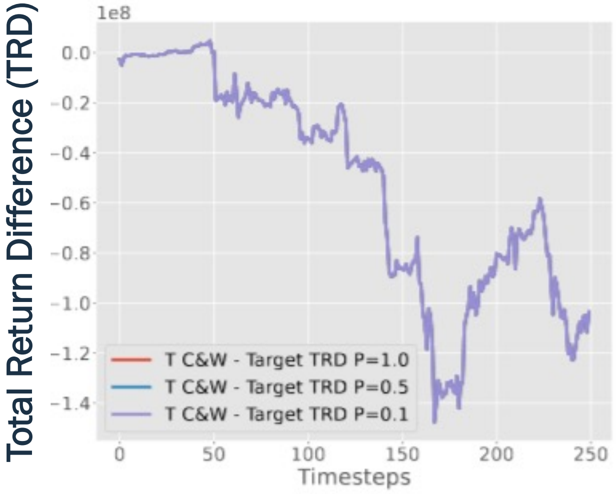


(b) Targeted C&W Attack

TensorTrade DQN

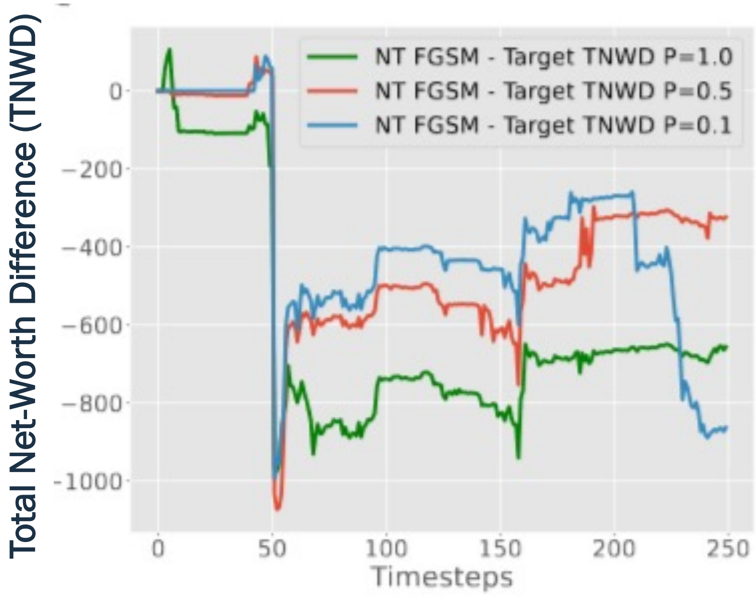


(a) Reward Difference Targeted FGSM Attack

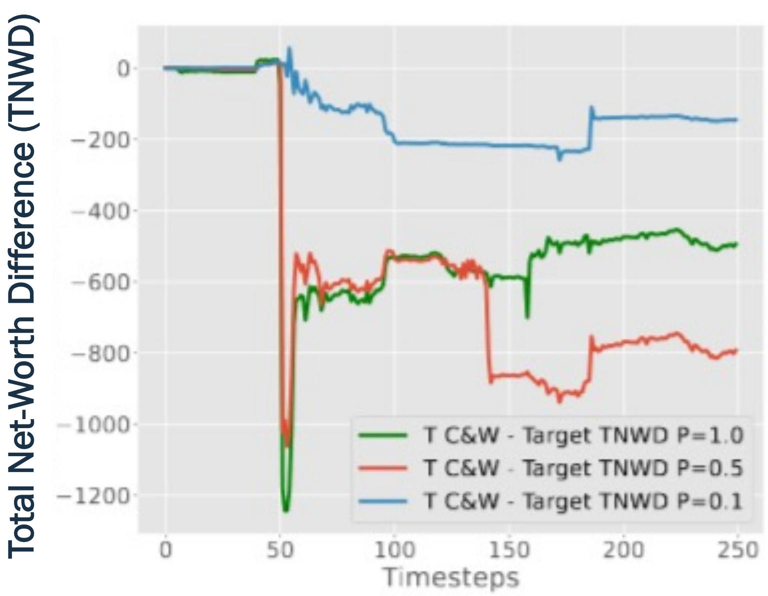


(b) Reward Difference Targeted C&W Attack

Net-Worth Impact Targeted - TensorTrade



(a) Net-Worth Difference Targeted FGSM Attack



(b) Net-Worth Difference Targeted C&W Attack

Passive, Test-Time Attacks

Man-In-The-Middle (MiTM) Attack is a passive, test-time attack. How can observed trajectories be used by an adversary?

- If observation contains sensitive data, information is valuable.

→ Differential Privacy? Can people's information be compromised? Yes, e.g. language models can compromise sensitive information from the data it was trained on.

- If experiences can be used to train another policy that is like the target policy, can lead to more whitebox-like future attacks.

→ Gain information on target policy architecture for stronger attacks.

- If experiences can be used to train another policy that an adversary can directly use to make profit.

→ Explicitly policy imitation. But, there is a branch of learning called **Imitation Learning** that addresses leveraging demonstrations, we follow the perspective of adversarial use of IL methods from Behzadan, V.; and Hsu [2].

Imitation Learning

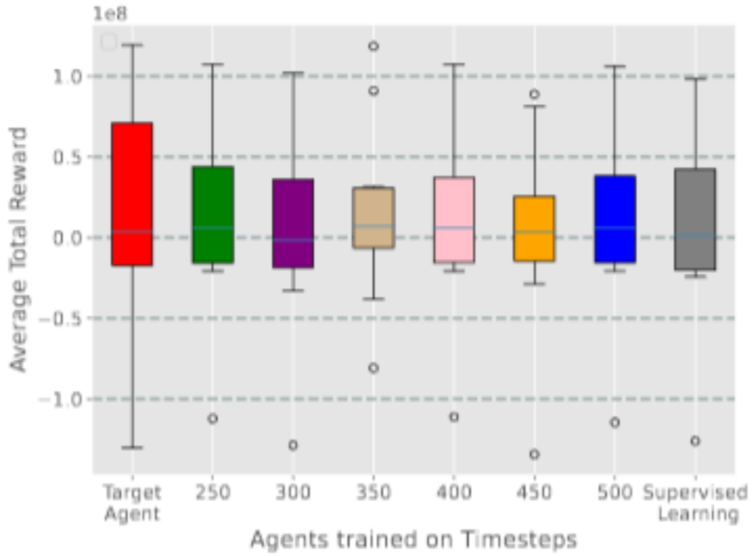
Imitation Learning (IL) is learning to imitate an expert policy through demonstrations. Behaviorial Clones (BC) are Supervised Learners that match demonstrations. IL + RL's objective is to learn a policy that is as good or out-preforms the expert demonstrations.

- Deep Q-Learning from Demonstration
  - *Imitation Learning variant*
  - *Intended to mitigate early stages of training.*
    - Off-Policy agents (or with a stochastic behavior policy) are often destructive during optimistic initialization.
  - *Basic Idea*

    1. Train an agent on demonstrations we assume are optimal/expert called the pretraining phase.

    2. After the pretraining phase, let the agent interact with the environment to generate its own experiences. Prioritize the expert's experience but also learn on self-generated experiences.

Imitated Agents vs. Target Agent



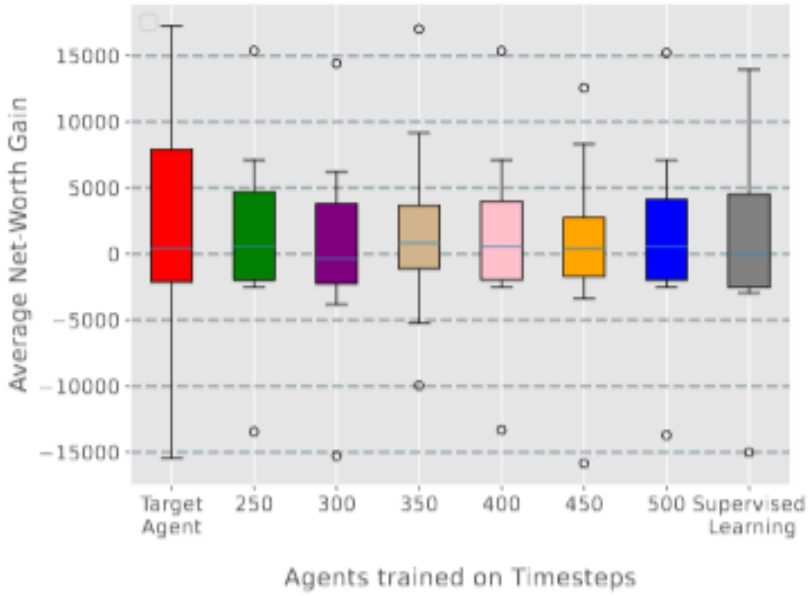


Figure 3: Average Total Reward over 10 Randomized Starts

	Training Evaluation			Randomized Evaluation		
Agent	Exact	Action Type	Length	Exact	Action Type	Length
250	19	207	250	207	1797	2490
300	27	162	300	146	1305	2490
350	26	61	350	233	251	2490
400	27	322	400	207	1799	2490
450	32	353	450	220	1772	2490
500	31	399	400	188	1819	2490
B.C.	345	349	350	341	1431	2490

Table 2: DQfD Agent Policy Action Match with Target Agent

- Passive Budget would be the length of the expert demonstration,
- Active Budget would be the number of evaluations

Imperfect Demonstrations

- DQfD assumes a continuous set of experiences can be provided but what if we cannot? How useful is imperfect demonstrations? How useful is approximating demonstrations?
- We cannot expect it to imitate the target agent.
- Can we still have competitive agents?

Figure 5: Average Total Reward over 10 Randomized Starts

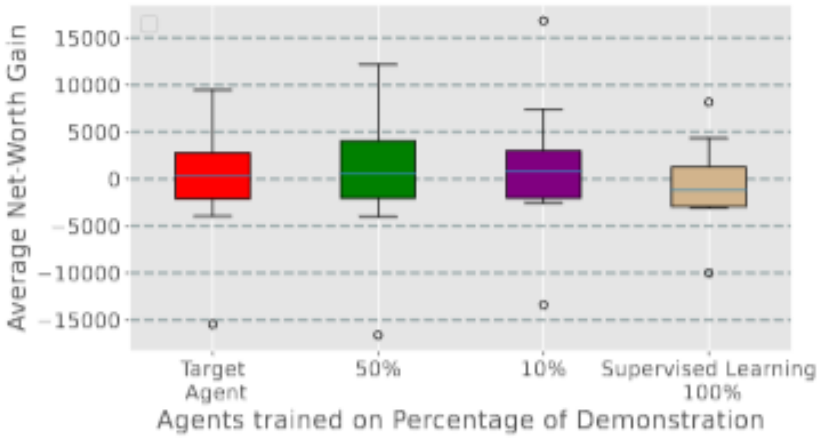


Figure 6: Average Net-Worth Gain over 10 Randomized Starts

Transferability

- Is an active, test-time attack, requires interception like MiTM.
- How susceptible is the target agent to whitebox optimization attacks that were successful to the imitated agents?
- Budget includes the passive budget + active budget.

Randomized Evaluation

agent	Successful Attack	Successful Transfer	No Attack Needed
250	7	1	10
300	166	44	253
350	15	5	17
400	2	1	10
450	10	2	11
500	88	24	75
250 (pre)	422	166	427
300 (pre)	195	70	245
350 (pre)	169	57	216
400 (pre)	938	500	702
450 (pre)	230	103	291
500 (pre)	228	98	260
B.C.	1545	979	544

Table 3: Successful Transferred FGSM Non-Targeted Attacks

Expectations for RL and DQfD

- Using reward functions like Sharpe Ratio is human-interpretable, but what are its caveats? Does this transfer to adversarial attacks that target these caveats?
- DQfD can produce competitive agents at less cost, but it is difficult to measure similarity to target agent unless under whitebox settings where we may query the target policy to account for distributional shift.
- Both DQfD agents and Behaviorial Cloning (through Supervised Learning) agent provide an adversary information which can be used for stronger, whitebox attacks.

Commentary on Trading DRL Agents

- We observe from the investigation that trading DRL Agents are susceptible:
    - Moving window of past tuples allow perturbations to stay in succeeding timesteps.
    - Some dimensions may be more sensitive to perturbation.
- If given a successful MiTM attack, the presence of the perturbation tuple in the observation space is enough to impact future timesteps.

What can be done?

- Threat model trading DRL (this presentation)
- Mitigation algorithms or defenses against active and passive adversarial attacks can apply to DRL trading algorithms.
  - *Active, test-time attack?* → *(specified) adversarial training*
  - *Passive, test-time attack?* → *Mitigation algorithms eg. our work Constrained Randomization of Policy (CRoP)[5].*
  - *Sensitive information?* → *There are works on Differential Privacy in RL through noise.*
  - *SL poisoning attacks transfer?* → *Yes, most norm-based attacks can apply to most domains.*
  - *Are there domain specific attacks?* → *Possibly, reward functions can be exploited;*

Our investigated models show implementation

and practicality of two well-known whitebox optimization attacks.

Conclusion

- We have shown that DRL trading agents are susceptible to adversarial attacks and can fool humans who manage these algorithms.
- We have outlined a threat model structure for trading agents based on a DRL Threat Model by Behzadan[3].
- We address the usefulness of passive attacks and how demonstrations can be leveraged for adversarial gain.
- We discuss existing areas of research that can be used for threat modeling trading agents for current and future areas of research.

References

- [1] GOODFELLOW, I. J., SHLENS, J.,ANDSZEGEDY, C. Explaining and harnessing adversarial examples. arXiv preprint *arXiv*:1412.6572(2014).
- [2] Carlini N. and Wagner, D. Towards evaluating the robustness of neural networks. Symposiumon Security and Privacy, (2017).
- [3] Behzadan, V.: Security of deep reinforcement learning. *Thesis*, (2019).
- [4] Faghan, Y.; Piazza, N.; Behzadan, V.; and Fathi, A. 2020.Adversarial Attacks on Deep Algorithmic Trading Policies.https://arxiv.org/pdf/2010.11388.pdf.
- [5] Behzadan, V.; and Hsu, W. 2019. Adversarial exploitation of policy imitation.arXiv preprint arXiv:1906.01121.

AUXILIARY

DQfD (A little More Detail)

- Sum of 4 losses: $J(Q) = J_{DQ}(Q) + \lambda_1 J_n(Q) + \lambda_2 J_E(Q) + \lambda_3 J_{L2}(Q)$
  - *Double DQN 1-step look-ahead loss*
  - *Double DQN 10-step look-ahead loss*
  - *Supervised Margin Loss* $J_E(Q) = \max_{a \in A} |Q(s,a) - l(a_E, a)| - Q(s, a_E)$
  - *L2 Regularization*
- Lambdas are scalars. L2 Regularization can be applied to the weights in the neural network.
- Double DQN loss is used to mitigate maximization bias, which occurs when the neural network is used to generate both the action and the Q-value.

The double DQN loss is defined as:

$$J_{DQ}(Q) = (R(s,a) + \gamma Q(s_{t+1}, a_{t+1}^{\max}; \theta') - Q(s,a;\theta))^2$$

where

$$a_{t+1}^{\max} = \arg\max_a Q(s_{t+1}, a; \theta)$$