Market Manipulation: An Adversarial Learning Framework for Detection and Evasion

Xintong Wang and **Michael P. Wellman**
University of Michigan, Ann Arbor
xintongw@umich.edu, wellman@umich.edu

Abstract

We propose an adversarial learning framework to capture the evolving game between a regulator who develops tools to detect market manipulation and a manipulator who obfuscates actions to evade detection. The model includes three main parts: (1) a generator that learns to adapt original manipulation order streams to resemble trading patterns of a normal trader while preserving the manipulation intent; (2) a discriminator that differentiates the adversarially adapted manipulation order streams from normal trading activities; and (3) an agent-based simulator that evaluates the manipulation effect of adapted outputs. We conduct experiments on simulated order streams associated with a manipulator and a market-making agent respectively. We show examples of adapted manipulation order streams that mimic a specified market maker's quoting patterns and appear qualitatively different from the original manipulation strategy we implemented in the simulator. These results demonstrate the possibility of automatically generating a diverse set of (unseen) manipulation strategies that can facilitate the training of more robust detection algorithms.

1 Introduction

Market manipulation is defined by the US Securities and Exchange Commission as "intentional conduct designed to deceive investors by controlling or artificially affecting the market for a security". Though it has long been present, manipulation practice has evolved in its forms, and is of increasing concern with the automation of trading and the interconnectedness of financial markets [Lin, 2015]. Automated programs are employed to inject deceitful information, as other traders make extensive uses of machine learning techniques to extract information from all possible sources (including the misleading ones) and execute decisions.

We focus on a specific but common form of manipulation, called *spoofing*, which is applied through a series of direct trading actions in a market. Traders interact with the market by submitting orders to buy or sell; we refer to the sequence of such actions taken by an individual trader over a period of time as the trader's *order stream*. Orders that do not execute immediately rest in the *order book*, a repository for outstanding offers to trade. At any given time, the order book for a particular security reflects the market's expressed supply and demand for that security. A manipulative order stream can be viewed as a *targeted attack* [Huang *et al.*, 2011], corrupting the order book's signal on supply and demand. False expressions in the manipulative order are designed to fool traders about the market state, leading them to alter trading behavior in a way that will directly move the price and benefit the manipulator. Fig. 1 illustrates an alleged spoofing order stream.

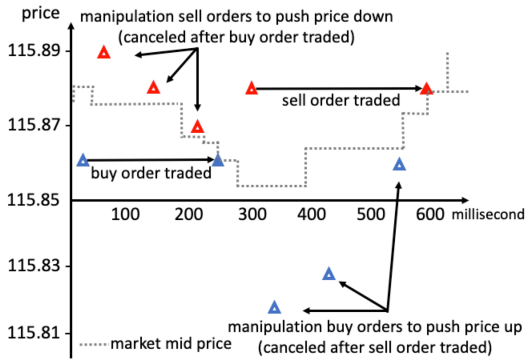


Figure 1: An example of spoofing activities conducted over the course of 0.6 seconds. A series of large out-of-the money manipulation sell orders (red triangles) are first placed to drive prices down and make the buy order accepted (the filled blue triangle). The deceptive sell orders are immediately replaced with large buy ones (blue triangles) to push the price up and profit from the sale at a higher price (the filled red triangle). Source: UK Financial Conduct Authority.

The automated nature of many manipulative strategies has also spurred efforts to automate detection. Nasdaq recently announced an AI-based surveillance system trained with historical data and spotted patterns of market-abuse techniques to detect suspect equities trading practices [Rundle, 2019]. Despite recent advances, developing high-fidelity detection systems faces the challenge that an adversary often takes steps to obfuscate their strategies in effort to escape detection (e.g., manipulating in a way that appears as normal trading activity). This causes regulators to play a costly game of cat-and-mouse with manipulators who constantly innovate to evade.

We propose an adversarial learning framework to reason about how a manipulator might mask its behavior (represented

as the order stream) to evade detection of a given discriminative model. The idea is to let a generative model learn to adapt existing manipulation strategies to resemble characteristics of normal trading, while preserving a comparable manipulation effect. A history of adapted order streams that effectively manipulate are further used to improve the robustness of the detector. We apply such adversarial reasoning recursively, updating the generator and the discriminator level-by-level, and characterize the evolution of adapted manipulation strategies.

Our generative model adopts the sequence-to-sequence paradigm [Sutskever *et al.*, 2014], and takes a manipulation order stream as source and a paired benign trader's order stream as target. It learns to adapt the source by minimizing the combination of an adversarial loss and a self-regularization loss. The adversarial loss is calculated by a discriminator that classifies an order stream as adapted from manipulation or target, minimized as the output becomes indistinguishable from a benign trader's order stream. The self-regularization loss is a feature-wise distance between the source and the adapted stream, penalizing large changes between the two to preserve the manipulation effect.

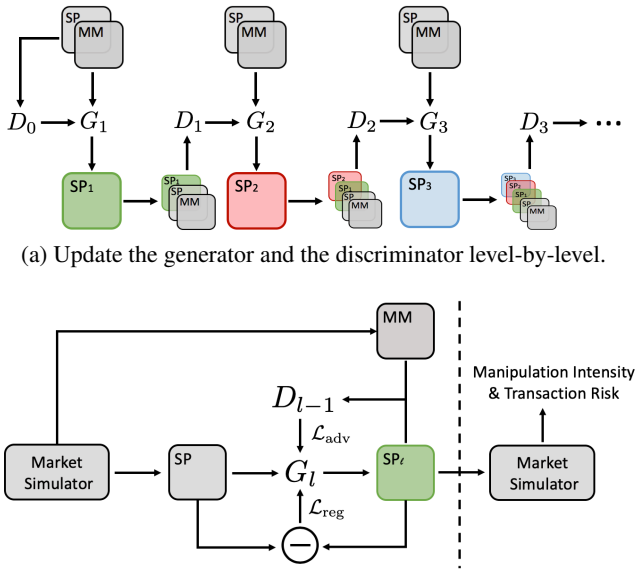
We conduct experiments and evaluate the proposed approach using order streams generated by an agent-based market simulator.[1] The simulator models simple manipulation strategies similar to Fig.1 [Wang and Wellman, 2017], and can practically produce a large set of order streams associated with each agent across a variety of market conditions. We run controlled simulations to acquire order streams associated with a manipulator (SP) as *source*, and as *target* the order streams that a market-making agent (MM) would have placed under corresponding market conditions. To help quantify the manipulation effect, we decompose the SP behavior into manipulation and exploitation orders, and define a *baseline* order stream (EXP) that omits the manipulation orders (i.e., those not intended to execute). Our goal here is to adapt manipulation order streams to resemble market-making, a legitimate trading role with generally positive influence on market efficiency [Schwartz and Peng, 2012; Wah *et al.*, 2017]. Fig. 2 gives an overview of our approach.

Experimental results show that our proposed framework can generate adapted manipulation order streams that resemble quoting patterns of a market maker and appear qualitatively different from the original manipulation strategy we implemented in the simulator. This adaptation evades detection, but at the cost of compromising effectiveness in manipulation. After a few iterations of evolving and evading the detector, the strategy has sacrificed almost all of its manipulation capability. Though it is likely impossible to develop a detector immune from adversarial attacks, modeling the evasion can be a useful step toward more robust detection of market manipulation.

2 Related Work

Agent-Based Models of Market Manipulation To study the effects of particular trading practices, researchers classify market participants into different roles based on their trading

[1]Learning from real market data is infeasible, as actual order streams identified as manipulation do not exist in any substantial quantity.



(a) Update the generator and the discriminator level-by-level.

(b) Given a fixed detector D_{l-1}, train G_l to generate SP_l.

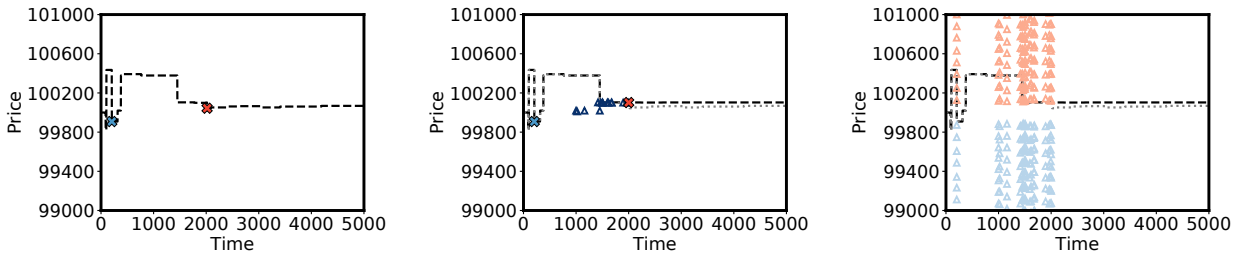
Figure 2: Overview of our approach. We start with a classifier D_0 that discriminates between SP and MM order streams. In response, a generator G_1 learns to adapt SP order streams, producing SP_1 that can evade detection by D_0. SP_1 order streams are then incorporated to train the next-level discriminator D_1. We apply such adversarial reasoning recursively, producing a sequence of adapted manipulators and corresponding increasingly robust detectors.

intent and activity patterns (e.g., trading volume, frequency, position). An agent-based market simulator designs agents around such roles, and reproduces "stylized facts" observed in real financial markets through strategic interactions of these agents [LeBaron, 2006; Kirilenko *et al.*, 2017].

In prior work, we developed an agent-based model of market manipulation [Wang and Wellman, 2017], demonstrating settings where a manipulation agent can effectively deceive approximately rational background traders through spoofing. Specifically, in markets populated with background learning traders who bid based on beliefs induced from market observations including the malicious activities, the manipulator is able to push prices significantly higher than they would be otherwise, and profit from this manipulation. Since background trading agents react to different market conditions according to their codified strategies, the model can verify manipulation intent and quantify its impact by conducting controlled experiments of markets with and without a spoofing agent.

Learning via Adversarial Training There is a substantial body of work on *adversarial training* [Goodfellow *et al.*, 2015; Tzeng *et al.*, 2017; Volpi *et al.*, 2018; Sinha *et al.*, 2018], investigating a variety of training procedures designed to learn models robust to (adversarial) perturbations in the input. Many of these approaches involve augmenting training dataset with examples from a target domain that is considered "hard" under the current model. A key issue addressed in some but not all of this work is to preserve specified properties of the source domain while generating adversarial examples to improve robustness.

Our approach draws particular inspiration from Shrivastava



(a) EXP (baseline). (b) SP (source). (c) MM (target).

Figure 3: Order streams associated with EXP, SP, and MM in a set of controlled simulations. During the execution stage (time before 1000), both EXP and SP bought one share of the security at price 99,908. Then, SP maintained manipulation buy orders at a tick behind the best bid to push the price up. As a result, SP managed to sell the share at price 100,102, whereas EXP sold the share at 100,044.

et al. [2017], who proposed Simulated + Unsupervised (S+U) learning. The idea is to train a generative model to improve the realism of simulated images using unlabeled real ones, while preserving the annotation information from the simulator. A pixel-level loss is further imposed between the simulated input and the generated image to enforce annotation. Experimental results show that S+U learning enables the generation of highly realistic images with reliable labels and helps to improve learning models' performance on classification tasks, including gaze estimation and hand pose estimation. Our work extends the approach to adapt simulated order streams while preserving the intent behind the original sequence of actions.

3 Formulation and Model

3.1 Trading Strategies and Representations

We follow prior work [Wang and Wellman, 2017; Wang *et al.*, 2018; Wah *et al.*, 2017] in the design of manipulation and market-making strategy, extending each with a bit of flexibility to reduce overfitting to artifacts. We describe the trading strategies and their representations as order streams below.[2]

Manipulation Strategy (SP) During each simulation run, the manipulator aims to maneuver prices either up or down with equal probability. We elaborate the case of manipulating prices up, and the other applies vice versa. The strategy includes three stages. During the first execution stage, the agent buys by accepting any sell order at price lower than the fundamental mean $\bar{r}$. In the next manipulation stage, it stops buying and instead maintains large manipulation buy limit orders at price one tick below the best bid. The goal is to falsely signal demand to push price up so that the units bought earlier can be sold at higher prices later. During the last stage, the manipulator starts to sell the units by accepting any buy orders at a price higher than $\bar{r}$. The agent continues to manipulate until the trading period ends or all the bought units are sold.

Market-making Strategy (MM) Upon each arrival, the market maker submits a quote ladder centered around an estimate of the terminal fundamental value of the underlying security, denoted by $\hat{r}_t$. Specifically, the quote ladder is decided by three strategic parameters ω, K, ζ that respectively control the quote spread, number of price levels, and the number of ticks between two adjacent prices:

$$\begin{cases} [B_t - K\zeta, \ldots, B_t - (K-\beta)\zeta] & \text{for buy orders} \\ [S_t + (K-\alpha)\zeta, \ldots, S_t + K\zeta] & \text{for sell orders,} \end{cases} \quad (1)$$

where $B_t = \hat{r}_t - \omega/2$, $S_t = \hat{r}_t + \omega/2$, and α and β truncate the price ladder such that limit orders do not immediately transact with the market's current best bid and ask. We add Gaussian noise around each price in Eq. (1) and its associated quantity to mitigate certain artifacts (e.g., prices separated by an equal distance). Since quote ladders are symmetrically centered around unbiased estimations of the terminal fundamental value, the MM orders in expectation do not distort learning traders' pricing beliefs. The MM agent follows the same arrival schedule as the manipulator to produce a paired target order stream, which records orders that would have placed under market conditions encountered by the manipulator.

Exploitation Strategy (EXP) The exploitation order streams serve as the control group to measure the effect of manipulation orders. The strategy executes the same buy and sell scheme as the SP strategy during the first and last stage without placing any manipulation order.

Order Stream Representation An order stream records a sequence of (hypothetical) actions associated with an agent. It is represented by a variable-length sequence with an element corresponding to each time an agent arrives and submits a *bid schedule*. A bid schedule comprises a set of limit orders, each specifying a price (expressed by distance to market quote) and a quantity. Fig. 3 shows order streams respectively associated with EXP, SP, and MM in a set of controlled simulations.

3.2 The Model

We use the market simulator to generate a dataset of labeled order streams $\mathcal{D} = \{(w_i, \text{EXP}), (x_i, \text{SP}), (y_i, \text{MM})\}_{i=1}^{N}$, where w_i, x_i, and y_i denote order streams incurred by their respective strategies under one set of controlled simulations (like those in Fig. 3). The goal here is to adapt the simulated SP order streams to become indistinguishable from the MM ones while preserving some manipulation effect.

Model Overview Our generator adopts the sequence-to-sequence paradigm [Sutskever *et al.*, 2014], which considers the interconnection between bid schedules within a sequence (e.g., a manipulator who buys first is more likely to

[2]Since an order stream is a sequence of actions incurred by a strategy, we refer to a strategy and order streams associated with that strategy interchangeably.

manipulate price up and later sell). It has an encoder-decoder structure $G_\theta = (G_{\text{enc}}, G_{\text{dec}})$, where θ denotes the function parameters. This encoder-decoder model has been widely used in tasks that require sequence-to-sequence learning, such as the statistical machine translation [Sutskever *et al.*, 2014; Cho *et al.*, 2014] and sentence generation [Logeswaran *et al.*, 2018]. The encoder adopts a recurrent neural network (RNN) that takes an order stream x as input and produces a fixed-length latent representation vector $z_x := G_{\text{enc}}(x)$. The vector contains compressed information of the input (e.g., manipulate prices up or down), and is decoded by G_{dec}, a second RNN that generates $x' \sim p_{G_{\text{dec}}(\cdot|z_x)}$ to resemble characteristics of the target domain y. The discriminator D_ϕ also uses an RNN component followed by a linear layer, and outputs the probability of an input being an adapted order stream.

A Recursive Training Procedure We propose a recursive training procedure of the generator and the detector (depicted in Fig. 2a), designed to mimic the adversarial reasoning between a manipulator and a regulator. The manipulator starts by playing the SP strategy that is codified in our market simulator, and the regulator develops detector D_0 to distinguish manipulation order streams from MM streams. The manipulator then constructs its next-level strategy SP_1 by learning a generator G_1 to adapt SP, such that the adapted order streams can evade the detection of D_0 and preserve a comparable manipulation effect. To achieve both aims, the generator is trained to minimize a combination of adversarial loss and regularization loss (depicted in Fig. 2b), which we describe in detail below. In response, a new detector D_1 is trained to identify both the original manipulation strategy SP and the evolved one SP_1. We apply such reasoning recursively to generate adversarial manipulation activities, so as to improve the robustness of a detector.

Adversarial Loss We follow the GAN setup [Goodfellow *et al.*, 2014] which models the generator and the discriminator as a two-player minimax game. During training, the level-l discriminator network D_l updates its parameters ϕ_l to minimize the following loss:

$$\mathcal{L}_D(\phi_l) = -\sum_i \log(D(x'_i;\phi_l)) - \sum_i \log(1 - D(y_i;\phi_l)), \quad (2)$$

where x'_i represents some learned (or identity) transformation of x_i, and $D(\cdot)$ denotes the probability of the input order stream either associated with or adapted from SP.

We fix the discriminator D_{l-1} and train the level-l generator G_l to maximize the probability of D_{l-1} making a mistake. Specifically, it learns θ_l by minimizing the adversarial loss:

$$\mathcal{L}_G^{\text{adv}}(\theta_l) = -\sum_i \log(1 - D_{l-1}(G(x_i;\theta_l))). \quad (3)$$

Self-Regularization Loss To preserve manipulation effect, we combine the adversarial loss with a self-regularization loss that penalizes any difference between the adapted and original order stream. This can be interpreted as a manipulator preference to adapt its original manipulation strategy as little as possible to evade detection. We define regularization loss as the mean squared error between the input and the adapted order stream:

$$\mathcal{L}_G^{\text{reg}}(\theta_l) = \frac{1}{N}\sum_i \|G(x_i;\theta_l) - x_i\|_2^2, \quad (4)$$

	Payoff	Manipulation Effect	Transaction Risk	D_{l-1} (%)	D_l (%)
SP	$411^{*,**}$	1	0	-	100
SP_1	$362^{*,**}$	0.50	0.14	0.59	100
SP_2	310^{*}	0.30	0.26	0	100
SP_3	303^{*}	0.22	0.59	0	100
MM	121	0.15	0.85	100	100
EXP	324^{*}	0	0	-	-

Table 1: Summary statistics of the respective trading strategy on test dataset. Asterisks denote statistical significance at 5% level of the paired t-test for payoffs compared to MM(*) and EXP(**).

where $\|\cdot\|_2$ is the L2 norm. The overall loss for G is $\mathcal{L}_G = \mathcal{L}_G^{\text{adv}} + \lambda\mathcal{L}_G^{\text{reg}}$, where λ is a hyperparameter.

Measuring Manipulation Effect We evaluate the manipulation effect of an adapted order stream $x'_i := G_l(x_i)$ by feeding it back to the market simulator under the same set of experimental controls. That is, we compare the effects under scenarios where background traders are guaranteed to arrive at the same time, receive identical private values, and observe the same fundamental values as in simulations that generate w_i, x_i, and y_i. Any change in background bidding behavior can therefore be attributed to the adapted order stream.

We compare market outcomes incurred by the adapted order stream to those of markets with SP and EXP, and measure the *manipulation intensity* and *transaction risk*. The manipulation intensity of x'_i, denoted by $\delta_{x'_i}$, is defined as the fraction of the price deviation realized by x'_i in that of the SP order stream:

$$\delta_{x'_i} = \begin{cases} \min\left\{\max\left\{\dfrac{P_{x'_i} - P_{w_i}}{P_{x_i} - P_{w_i}}, 0\right\}, 1\right\} & \text{if } P_{x_i} > P_{w_i} \\ \min\left\{\max\left\{\dfrac{P_{w_i} - P_{x'_i}}{P_{w_i} - P_{x_i}}, 0\right\}, 1\right\} & \text{otherwise,} \end{cases} \quad (5)$$

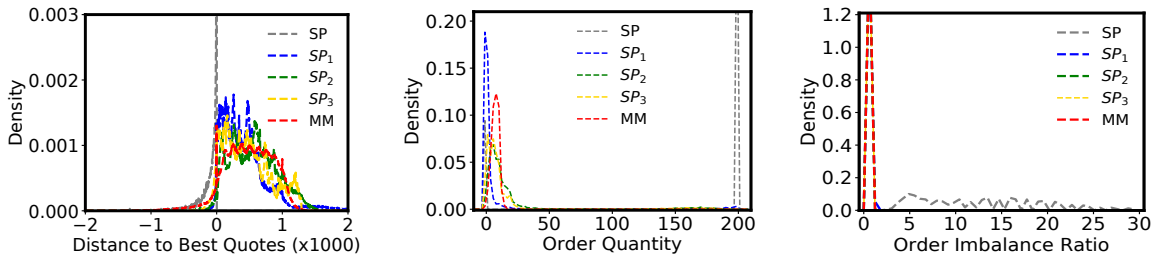
where P_{w_i}, P_{x_i}, and $P_{x'_i}$ denote the average transaction price in respective markets since the start of the manipulation stage. The higher the manipulation intensity is, the better x'_i preserves the manipulation effect. Transaction risk is defined as the ratio between the number of transactions and the number of arrivals during the manipulation phase. By definition, SP and EXP have manipulation intensity one and zero, respectively, and both exhibit transaction risk zero.

4 Experimental Results

We follow the proposed framework and generate adversarial order streams by adapting the simulated SP order streams to look like quoting patterns of a market maker. We visualize examples of adapted manipulation activities, and demonstrate the competing improvement between the adapted manipulation strategies and the detectors.

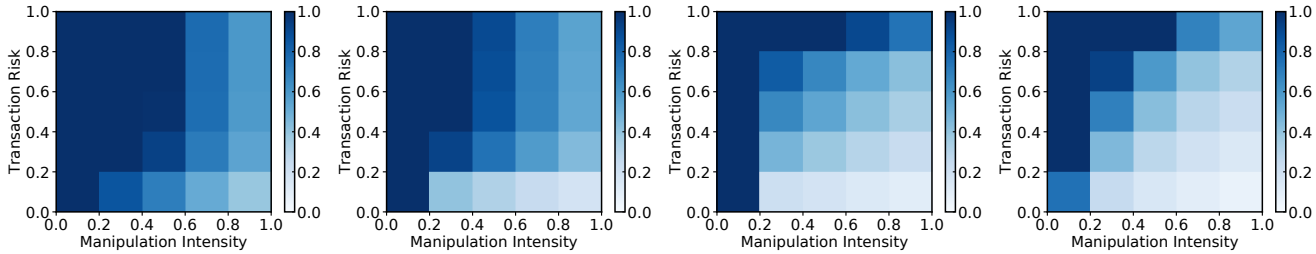
4.1 Dataset and Implementation Details

We conduct simulations using the agent-based market simulator, and generate 10,944 groups of labeled order streams $\{(w_i, \text{EXP}), (x_i, \text{SP}), (y_i, \text{MM})\}$ (out of 30,000 controlled



(a) Price distribution. (b) Quantity distribution. (c) Order imbalance distribution.

Figure 4: Comparisons of the respective statistics on the SP order streams, adapted outputs, and MM order streams.



(a) SP_1. (b) SP_2. (c) SP_3. (d) MM.

Figure 5: The manipulation effect of order streams associated with the corresponding level of SP strategy. Each color of a cell encodes the cumulative density of order streams that achieve a certain manipulation intensity and transaction risk. The closer dark blue is to the bottom right, the better adapted order streams are able to preserve higher manipulation intensity with lower transaction risk.

simulation runs).[3] Each trading session lasts 5000 time steps, and the generated order streams have lengths varying from 4 to 91. The first execution stage is from time 200 to 1000, after which the manipulation agent starts to spoof. At time 2000, it begins to liquidate previously accumulated positions. The underlying security has a fundamental mean $\bar{r} = 10^5$. Based on estimations of the final fundamental value, the MM submits a quote ladder with $\omega = 256$, $K = 8$, $\zeta \sim N(128, 10)$, and quantity $q \sim N(5, 2)$. We use 8,896 groups of order streams for training (with a 80/20 train-validation split) and the rest 2,048 groups for testing.

We use a bi-directional Gated Recurrent Unit (GRU) RNN [Cho *et al.*, 2014] with a hidden state size of 64, followed by a linear layer for both G_{enc}, G_{dec}, and D in the experiments. Since order streams are of variable lengths, we pad them to the maximum length for forward passes, and cut them back to original lengths for loss calculations and evaluations. We initialized the model parameters with the uniform distribution between -0.08 and 0.08. We use batches of 64 order streams to train the discriminator and the generator, and pick weight of the self-regularization loss $\lambda = 1$ based on the validation performance.

4.2 Generating Adapted Manipulation Examples

We evaluate the adversarially adapted order streams from three main aspects: (1) similarity to the MM quoting patterns, (2) preservation of manipulation effect, and (3) effectiveness in evading the detection of an existing discriminator. Table 1 presents summary statistics of order streams associated with their corresponding trading strategies (or generative models). We discuss each aspect in detail below.

Comparing to MM We follow prior work [Li *et al.*, 2020] in using price and quantity distributions to measure how well the generated order streams resemble the target MM streams. We further propose a domain-specific measure, the *order imbalance distribution*, defined as the ratio between the numbers of buy and sell orders submitted over a trading period (whichever value is larger on the numerator). This captures a trader's imbalance in preference between long and short positions. Fig. 4 presents comparisons of the respective distributions. We find the adapted manipulation order streams produce distributions similar to that of the MM, and are able to overcome artifacts codified in the SP strategy (e.g., large-quantity orders always at one tick behind the best quotes and severe order imbalance to deceive the market). Specifically, we find that orders are gradually adapted to cover a wider range of prices with relative small quantities, and order balance is roughly maintained throughout the trading period.

Preserving Manipulation Effect We feed adapted order streams back to the market simulator under the same set of experimental controls, and measure their manipulation effect by the manipulation intensity and transaction risk as defined in Sec. 3. Fig. 5 shows the two-dimensional cumulative density over the 2,048 adapted outputs with respect to the two proposed metrics. We find that SP_1 can preserve a comparable manipulation intensity under a reasonable transaction risk; however, as the generator adapts in response to a more robust

[3] We keep valid simulations where the manipulator successfully trades during the first stage (so that there is an incentive to spoof), and pushes prices to its desired direction by at least ten ticks.

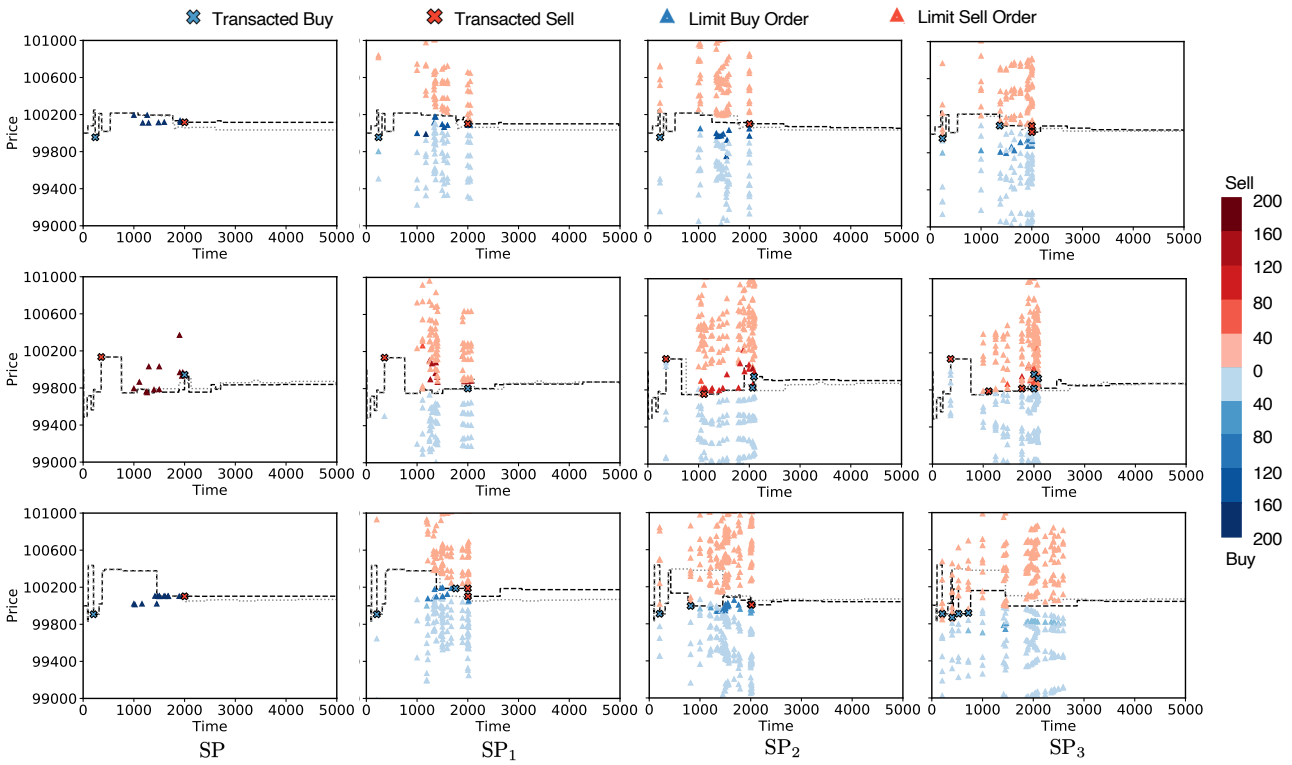


Figure 6: Examples of adapted manipulation order streams. Dashed black lines represent the latest transaction prices, whereas dashed grey lines the transaction prices if no manipulation exists.

discriminator, the adapted streams become to suffer a large degradation in manipulation intensity and an increase in transaction risk (e.g., SP_3 has a similar performance to MM). This weakened manipulation effect is further confirmed in Table 1.

Evading the Detection Table 1 shows that a generator can easily fool an existing detector with adversarially generated order streams. By learning from a history of adapted order streams, the discriminator is able to detect manipulation streams from all previous levels, and in the meantime ensures the training stability of the next-level generator.

Qualitative Evaluation Fig. 6 demonstrates examples of the original and its corresponding adapted manipulation order streams. We observe that the adapted streams become qualitatively similar to the trading patterns of a MM, and such simultaneous quoting behavior on both sides of the market has indeed been suggested as a good strategy for high frequency traders to mask their manipulative intent [Levens, 2015]. We note several other findings from the evolution of adapted manipulation strategies. First, SP_1 remains to place large orders close to the market best quote, whereas SP_2 and SP_3 choose to either largely decrease the order quantity or place large orders behind smaller ones to avoid being detected. Second, SP_2 and SP_3 tend to submit orders at more aggressive prices across market quotes, and this may cause unintended transactions during the manipulation phase.

5 Conclusion

We employ an adversarial learning framework to model the evolving game between a regulator and a manipulator, in which the regulator deploys algorithms to detect manipulation and the manipulator masks actions to evade detection. Evasion is represented by a generative model, trained by augmenting manipulation order streams with examples of market making activity traces. The intent is to produce adapted streams that are hard to distinguish from a market maker's behavior. We visualize examples of adapted manipulation order streams, and show they resemble quoting patterns of a market maker and appear qualitatively different from the original manipulation strategy we implemented in the simulator. This adaptation evades detection, but only at the cost of compromising effectiveness in market manipulation. After a few iterations of evolving and evading the detector, the strategy has sacrificed almost all of its manipulation capability.

Our results reflect the specific modeling and simulation choices adopted, and thus it remains to be seen whether a more clever form of adaptation can evade detection while retaining more effectiveness in manipulation. Whether or not it is possible to ultimately craft successful adversarial attacks, the generation and evasion process modeled here provides a way to anticipate the evolution of evasive adversaries. Such anticipation capacity provides a way to develop more robust detection methods, for market manipulation as well as other fraudulent behaviors.

Acknowledgments

This work was supported in part by the US National Science Foundation IIS-1741190.

References

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *International Conference on Neural Information Processing Systems*, pages 2672–2680, 2014.

[Goodfellow *et al.*, 2015] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[Huang *et al.*, 2011] Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I.P. Rubinstein, and J. D. Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, page 43–58, 2011.

[Kirilenko *et al.*, 2017] Andrei A. Kirilenko, Albert S. Kyle, Mehrdad Samadi, and Tugkan Tuzun. The flash crash: High frequency trading in an electronic market. *Journal of Finance*, 72:967–998, 2017.

[LeBaron, 2006] Blake LeBaron. Agent-based computational finance. *Handbook of Computational Economics*, 2:1187–1233, 2006.

[Levens, 2015] Tara E. Levens. Too fast, too frequent? High-frequency trading and securities class actions. *University of Chicago Law Review*, 82:1511–1558, 2015.

[Li *et al.*, 2020] Junyi Li, Xintong Wang, Yaoyang Lin, Arunesh Sinha, and Michael P. Wellman. Generating realistic stock market order streams. In *34th AAAI Conference on Artificial Intelligence*, 2020.

[Lin, 2015] Tom C. W. Lin. The new market manipulation. *Emory Law Journal*, 66:1253–1314, 2015.

[Logeswaran *et al.*, 2018] Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. Content preserving text generation with attribute controls. In *International Conference on Neural Information Processing Systems*, pages 5108–5118, 2018.

[Rundle, 2019] James Rundle. Nasdaq deploys AI to detect stock-market abuse. *Wall Street Journal*, 2019.

[Schwartz and Peng, 2012] Robert A Schwartz and Lin Peng. Market makers. *Encyclopedia of Finance*, 2012.

[Shrivastava *et al.*, 2017] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2242–2251, 2017.

[Sinha *et al.*, 2018] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. In *International Conference on Learning Representations*, 2018.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *International Conference on Neural Information Processing Systems*, pages 3104–3112, 2014.

[Tzeng *et al.*, 2017] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2962–2971, 2017.

[Volpi *et al.*, 2018] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *International Conference on Neural Information Processing Systems*, pages 5339–5349, 2018.

[Wah *et al.*, 2017] Elaine Wah, Mason Wright, and Michael P. Wellman. Welfare effects of market making in continuous double auctions. *Journal of Artificial Intelligence Research*, 59:613–650, 2017.

[Wang and Wellman, 2017] Xintong Wang and Michael P. Wellman. Spoofing the limit order book: An agent-based model. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 651–659, 2017.

[Wang *et al.*, 2018] Xintong Wang, Yevgeniy Vorobeychik, and Michael P. Wellman. A cloaking mechanism to mitigate market manipulation. In *International Joint Conference on Artificial Intelligence*, pages 541–547, 2018.