Learning who is in the market from time series: market participant discovery through adversarial calibration of multi-agent simulators.

Victor Storchan [1]   Svitlana Vyetrenko [2]   Tucker Balch [2]

Abstract

In electronic trading markets often only the price or volume time series, that result from interaction of multiple market participants, are directly observable. In order to test trading strategies before deploying them to real-time trading, multi-agent market environments calibrated so that the time series that result from interaction of simulated agents resemble historical are often used. To ensure adequate testing, one must test trading strategies in a variety of market scenarios – which includes both scenarios that represent ordinary market days as well as stressed markets (most recently observed due to the beginning of COVID pandemic). In this paper, we address the problem of multi-agent simulator parameter calibration to allow simulator capture characteristics of different market regimes. We propose a novel two-step method to train a discriminator that is able to distinguish between "real" and "fake" price and volume time series as a part of GAN with self-attention, and then utilize it within an optimization framework to tune parameters of a simulator model with known agent archetypes to represent a market scenario. We conclude with experimental results that demonstrate effectiveness of our method.

1. Introduction

Multi-agent simulation is widely used in financial markets to study macroscopic structural effects in counterfactual scenarios. It is especially useful as a natural bottom-up approach in situations where the environment needs to be responsive to experimental agent's actions such as trading strategy testing. Thorough trading strategy testing must include exposure to a variety of market scenarios in order to gain understanding how experimental agents will behave when market dynamics shifts. For example, in March 2020 markets became significantly more volatile and illiquid due to the start of COVID pandemic, which affected performance of trading strategies. Therefore, there is a practical need to be able to calibrate multi-agent simulators to a variety of scenarios for proper risk management.

[1]JP Morgan, Palo Alto, USA [2]JP Morgan AI Research, New York, USA. Correspondence to: Victor Storchan <victor.storchan@jpmchase.com>, Svitlana Vyetrenko <svitlana.s.vyetrenko@jpmchase.com>, Tucker Balch <tucker.balch@jpmchase.com>.

Explainability is an additional benefit of multi-agent simulation. One can provide insightful answers to "what if" questions by explicitly changing agent composition of simulated markets. When individual agent-specific data is available to the researcher, it can be used for the learning the market agent policy (e.g., (Vyetrenko & Xu, 2019), (Darley & Outkin, 2007)); however, most frequently such labeled data is proprietary and only the price or volume time series that result from interaction of multiple market participants are directly observable. Some market participants, such as exchanges or over-the-counter market markers, however, have additional knowledge of parametrized agent archetypes that play in the market (momentum traders, value traders, noise (retail) traders, etc.) – such participants are interested in tuning parameters of the known agent archetypes so that the result of their interaction resembles historical time series for counterfactual scenario calibration. Explicit optimization objectives for simulator parameter tuning are, however, difficult to construct due the complex nature of time series that results from agent interaction. Therefore, we propose to learn a discriminator (i.e. a classifier) that can distinguish synthetic time series from real, and use it as an implicitly written objective function for the calibration task.

2. Our contributions

In this paper, we propose MAS-GAN – a two-step method for multi-agent market simulator calibration. A discriminator is trained in competition with the generator as a part of GAN to distinguish synthetic time series from real (i.e. historical). Both generator and discriminator are enhanced by self-attention layers for better time series resolution quality. The input of trained discriminator is a synthetic time series, and the output uses sigmoid activation which allows to interpret it as probability of synthetic time series being real. Similar to (Liang et al., 2018), the discriminator score is higher if the synthetic time series shares more features with the historical dataset. One can then use the discriminator score as an implicit optimization objective, and optimize simulated model parameters to determine the set that produces most realistic time series (see Figure 1 for schematic).

To the best of our knowledge, MAS-GAN is the first method that proposes to use adversarially trained discriminator as an objective function for multi-agent system optimization.
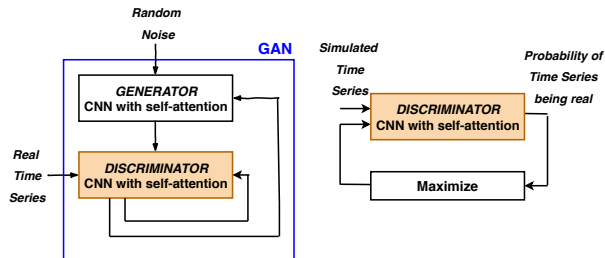
Figure 1: MAS-GAN approach schematic.

Unlike (Alonso-Monsalve & Whitehead, 2018), our method is model-agnostic and does not require learning a simulator approximation which can be expensive. Because MAS-GAN learns from historical dataset and is unbiased toward synthetic data coming from the simulator, once the discriminator is trained – it can be used to calibrate any time series model (not necessarily multi-agent) that is intended to model that historical dataset. In contrast with (Li et al., 2020), our method retains full explainability of multi-agent simulations as GANs in our case are not used for black-box data generation, but rather only for learning a calibration objective given by the discriminator.

Let $S_R(v)$ be the simulated mid price time series obtained using the parameter vector v and random seed R. Since the discriminator D outputs probability of time series $S_R(v)$ being real, we are interested in finding parameter vector v^* that maximizes this probability. Formally, we are interested in finding $v^* = \arg\max_v E_R[D(S_R(v))]$. We then perform grid-based optimization with respect to the discriminator-defined objective to find most realistic simulation parameters out of the given grid.

3. Training details

Most of the GAN literature focuses on training GANs for the purpose of subsequently using only data generator, and discriminator is only viewed as an auxiliary agent that assists with generator training. To complement this, using GAN-trained discriminator for the purpose of time series anomaly detection is discussed in (Schlegl et al., 2017), (Mattia et al., 2019), (Li et al., 2018).By learning statistical similarities between non-anomalous time series, one can learn to determine what constitutes an anomaly. In this paper, we rely on a similar idea and introduce an application of adversarially trained discriminators to multi-agent simulator parameter optimization.

Using GANs to generate synthetic time series has been widely studied – however, only LSTMs and recurrent neural network architectures have been previously considered (e.g., (Esteban et al., 2017)). To the best of our knowledge, we are innovating by using both conv1D and self-attention without recurrence in the architecture to generate synthetic time series. We use conv1D to encode the local correlations in the data. The deeper the network is, the more global encoding will happen. We add self-attention to provide a more targeted way to encode global correlations with a single layer. A 2D version of the self-attention layer was described in (Zhang et al., 2019). For this work, we use a similar implementation of a 1D version of the self-attention layer (see Figure 8 in Appendix A.2 for schematic).

3.1. Self-Attention

Transformer's head mechanisms, introduced in (Vaswani et al., 2017) have proved their efficiency at capturing global dependencies in multiple domains of applications such as translation, language modeling or object detection. Among the different attention mechanisms, self attention is computing a representation of the next position in a sequence by looking at all the position in the sequence and learning regions of interest. In self-attention, we route information according to the information itself, so according to the incoming information. We do that by expressing queries expressing the kind of information needed from the previous layer and keys exposing the kind of information the nodes of the next layer are containing. To aggregate the results, we use the softmax function which is a kind of soft routing: all the nodes of the previous layer are contributing to the expression of the node of the next layer but the majority goes to the places where the inner product is high (see Figure 8 in Appendix A.2).

3.2. WGAN with gradient penalty

In the original paper (Goodfellow et al., 2014), GANs are trained using the Jensen-Shannon Divergence (JSD), essentially to solve the problem of KL divergence in Variationnal Autoencoders (VAE) which is note define in the real world scenario where the model manifold and the true distribution's support have not a non-negligible intersection (Arjovsky & Bottou, 2017). However, in (Arjovsky et al., 2017) example 1 shows that it is still not a good notion of distance. The authors introduce Wasserstein GANs which use the earth mover distance as a distance between distribution. It has the advantage of being a true metric: a measure of distance in a space of probability distributions. Instead of having a discriminator evaluating the probability of a sample to be real, we train a critics that is computing a realism score for each sample. WGANs are less vulnerable to getting stuck than minimax-based GANs, and avoid problems with vanishing gradients. However, the Kantorovitch-Rubinstein duality is stating that the critics should lie in the set of K-lipschitz functions. To implement this constraint, the original WGAN paper is clipping the weights of the critics. But it biases it towards learning much simpler functions. We chose to implement the constraint proposed by (Gulrajani et al., 2017) where the norm of gradient of the critic with re-
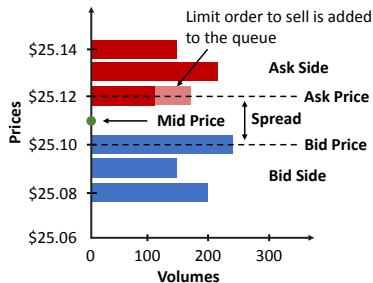
Figure 2: Visualization of the LOB structure.

spect to its input is penalized. See Figure 9 in Appendix A.3 for the detailed schematic of GAN architecture.

4. Experimental results

4.1. Simulated environment

Public exchanges and over-the-counter market makers facilitate the buying and selling of assets by accepting and satisfying buy and sell orders from multiple market participants via limit order book (LOB) mechanism, which is an electronic record of the queues of outstanding buy and sell limit orders organized by price levels (Bouchaud et al., 2018). See Figure 2 for visualization of the LOB structure.

We apply MAS-GAN to calibrate the LOB simulation environment [citation redacted] that provides a selection of basic agent archetypes (such as market makers, noise agents, value agents, etc.), a NASDAQ-like exchange agent which lists any number of securities for trade against an LOB with price-then-FIFO matching rules, and a simulation kernel which manages the flow of time and handles all inter-agent communication.

We use agent archetype implementation introduced in (Vyetrenko et al., 2020). *Value agents* are designed to simulate the actions of fundamental traders that trade according to their belief of the exogenous value of a stock (also called fundamental price – which in our model follows a mean reverting price time series). Value traders arrive to the market at a Poisson rate λ and choose to buy or sell a stock depending on whether it is cheap or expensive relative to their noisy observation of a fundamental price. The fundamental follows a discrete-time mean-reverting Ornstein-Uhlenbeck process (Byrd, 2019). *Noise agents* are designed to emulate the actions of retail players who trade on demand without any other considerations; N noise agents arrive to the market at times that are uniformly distributed throughout the trading day and place an order of random size in random direction. Finally, *market makers* act as liquidity providers by placing limit orders on both sides of the LOB with a constant arrival rate.

4.2. Training details

In real limit order book markets, multiple correlated data streams are produced as a result of participant interaction (e.g., prices and quotes at multiple limit order book levels and traded volumes). MAS-GAN allows to capture temporal autocorrelations and cross-correlations of multiple limit order book time series distributions. Specifically, we train a generative adversarial network on T-second mid price returns and cumulative T-second traded volumes. To produce the input vector that is passed to the GAN discriminator, we concatenate T-second mid price returns and volumes. It is done in order for the neural network with self-attention to encode correlations between price and volume time series and to retranscribe them when generating time series from noise.

We note that in order to train a high quality discriminator, the generator needs to be capable to generate a diverse set of realistic time series. We train the GAN iteratively until the following three conditions are satisfied: (1) Generated time series visually show diversity – see Figure 5 in Appendix A for visual inspection. We are aware of the issues raised by (Ryan et al., 2019) that visual inspection might not be sufficient to demonstrate diversity as time series can be simply memorized by GANs and leave this question for further work; (2) Asset return distributions and volume/volatility correlation of the generated time series converge to historical – see Figure 7 in Appendix A for the training progression; (3) The discriminator cannot distinguish generated time series from real – see Figure 6 in Appendix A for distribution of discriminator scores during training.

4.3. Configuration 'recovery' experiment

We test MAS-GAN on synthetic configurations that are designated as "ground truth". We consider a family of synthetic market configurations with a single market maker, 100 value agents that arrive to the market at Poisson rate λ and N noise agents (see agent descriptions in Section 4.1) - with λ and N being calibration parameters. In Figure 3 one can see simulated mid price time series generated for different values of parameters λ and N. Different parameter values can be used to model different liquidity regimes. For example, parameter values from Figure 3(a) can used to simulate "ordinary" liquid markets, whereas, parameter values from Figure 3(b) model a "bumpy" illiquid markets – such markets can be can be observed during shock periods such as those that were seen in the beginning of COVID pandemic. We fix λ^* and N^* that correspond to different liquidity configurations as per charts Figure 3(a) and (b). For different seeds for initialization of pseudo-random number generator, one can produce multiple time series samples — that is, simulate trading days with fixed agent participation.
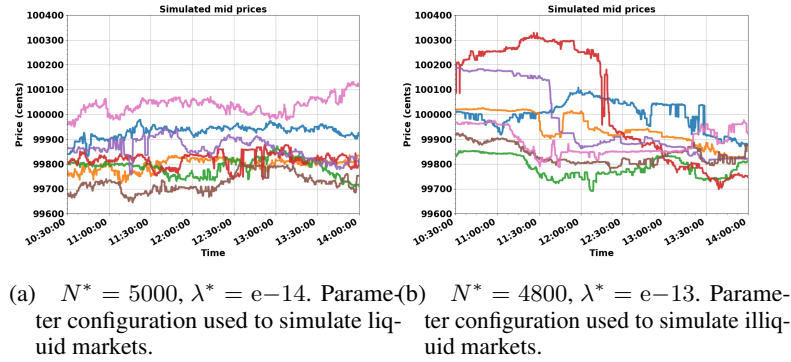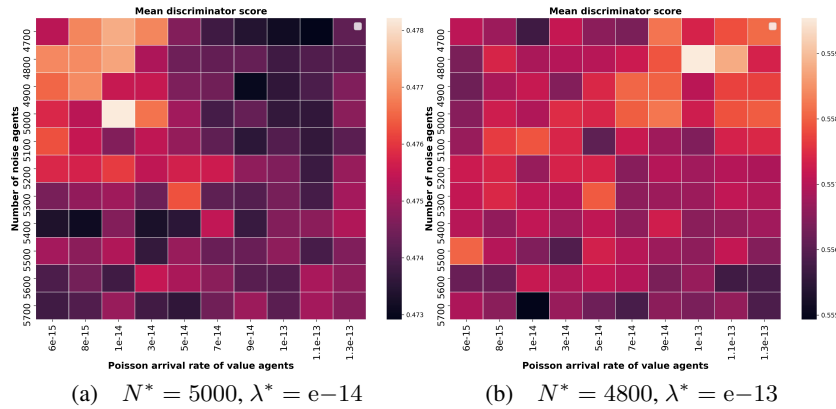
(a) $N^* = 5000$, $\lambda^* = \mathrm{e}{-}14$. Parameter configuration used to simulate liquid markets.

(b) $N^* = 4800$, $\lambda^* = \mathrm{e}{-}13$. Parameter configuration used to simulate illiquid markets.

Figure 3: Mid price time series.

(a) $N^* = 5000$, $\lambda^* = \mathrm{e}{-}14$

(b) $N^* = 4800$, $\lambda^* = \mathrm{e}{-}13$

Figure 4: Mean discriminator score heatmap with respect to the noise agent number N^* and value agent arrival rate λ^*. For each grid configuration, we run 20 simulations with different seeds for initialization of pseudo-random number generator.

We run the following experiments to verify the validity of MAS-GAN: first, train a discriminator as a part of GAN with self-attention to recognize time series samples for fixed λ^* and N^* and varying random seeds; second, for each choice of λ and N over the grid, apply the trained discriminator to "recognize" λ^* and N^* — a configuration that a discriminator was trained against. Figure 4 illustrates the outcome of the above experiment for the respective values of λ^* and N^* as in Figure 3. We observe that applying discriminator to the rectangular grid can identify the configuration that the discriminator was trained against. We also notice that configurations in the immediate neighborhood of λ^* and N^* on the grid are identified as more likely to have trained the discriminator than those that are further away.

5. Conclusion

In summary, we propose a novel GAN-based method for multi-agent simulator parameter calibration and demonstrate its effectiveness experimentally. Our method relies on the basic knowledge of market agent archetypes, and can be ef-

fectively used for agent micro-parameter tuning to different market regimes by market participants such as exchanges or market makers to whom knowledge of basic archetypes is available.

6. Acknowledgements

References

Alonso-Monsalve, S. and Whitehead, L. H. Image-based model parameter optimisation using model-assisted generative adversarial networks. *CoRR*, 2018. URL http://arxiv.org/abs/1812.00879.

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks, 2017.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein gan, 2017.

Bouchaud, J.-P., Bonart, J., Donier, J., and Gould, M. *Trades, quotes and prices: financial markets under the microscope*. Cambridge University Press, Cambridge, 2018.

Byrd, D. Explaining agent-based financial market simulation. *arXiv preprint arXiv:1909.11650*, 2019.

Darley, V. and Outkin, A. V. *A NASDAQ Market Simulation:Insights on a Major Market from the Science of Complex Adaptive Systems*. World Scientific Publishing Co. Pte. Ltd., December 2007.

Esteban, C., Hyland, S., and Rätsch, G. Real-valued (medical) time series generation with recurrent conditional gans. 06 2017.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks, 2014.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. Improved training of wasserstein gans, 2017.

Li, D., Chen, D., Goh, J., and Ng, S. Anomaly detection with generative adversarial networks for multivariate time series. *CoRR*, abs/1809.04758, 2018. URL http://arxiv.org/abs/1809.04758.

Li, J., Wang, X., Lin, Y., Sinha, A., and Wellman, M. P. Generating realistic stock market order streams. In *To Appear in the 34th AAAI Conference on Artificial Intelligence*, 2020.

Liang, S., Li, Y., and Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1VGkIxRZ.

Mattia, F. D., Galeone, P., Simoni, M. D., and Ghelfi, E. A survey on gans for anomaly detection, 2019.

Ryan, W., Rabin, J., Simon, L., and Jurie, F. Detecting overfitting of deep generative networks via latent recovery. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/papers/Webster_Detecting_Overfitting_of_Deep_Generative_Networks_via_Latent_Recovery_CVPR_2019_paper.pdf.

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., and Langs, G. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Vyetrenko, S. and Xu, S. Risk-sensitive compact decision trees for autonomous execution in presence of simulated market response. In *ICML 2019 Workshop on AI in Finance*, 06 2019.

Vyetrenko, S., Byrd, D., Petosa, N., Mahfouz, M., Dervovic, D., Veloso, M., and Balch, T. H. Get real: Realism metrics for robust limit order book market simulations. In *International Conference on Artificial Intelligence in Finance*, 2020.

Wang, Z., Yan, W., and Oates, T. Time series classification from scratch with deep neural networks: A strong baseline. *CoRR*, 2016. URL http://arxiv.org/abs/1611.06455.

Zhang, H., I., G., Metaxas, D., and Odena, A. Self-attention generative adversarial networks. *International Conference on Machine Learning*, 2019. URL https://arxiv.org/pdf/1805.08318.pdf.

A. Appendix

A.1. Evaluating the quality of generated time series

A.2. 1D self-attention layer

In Figure 8 we present in more detail the attention mechanism that is accountable for encoding the global correlations of volumes and mid price returns feature maps.

A.3. GAN architecture

A.4. Ablation study and evaluation

To justify the complexity of our WGAN with gradient penalty and self-attention architecture in Figure 9, we compare it to a simpler GAN without self-attention that takes same latent dimension as an input, but whose both generator and discriminator are given by feed-forward neural networks with dropout layers and ReLU activation in intermediate layers. This discriminator architecture was studied in (Wang et al., 2016) and is depicted in Figure 10. We train both GANs on mid price return time series and observe that GANs with self-attention produce discriminators with better recognition capabilities, as evident from Figures 11a and 11b. One can see that adding self-attention to GANs reduces the variance of discriminator score distribution and makes it more concentrated around $\frac{1}{2}$ both for real and generator-produced time series. We further observe that by using both mid price return and volume time series for training a GAN with self-attention, we achieve sufficient discrimination accuracy quicker than by using only mid price returns.

To measure the performance of our model more quantitatively and to study the impact of the latent dimension size, we also conducted analysis using two-sample Kolmogorov-Smirnov test. The historical and generated discriminator scores distributions are compared. The null hypothesis (H0) is that "the two samples are drawn from the same underlying distribution". From Figure **??**, p-values indicates that after 10k iterations, for latent dimensions equal to 100 and 200 we can't reject the null hypothesis. This means that our generator is producing a distribution of time series that follows the same underlying distribution that the historical. For latent dimension equal to 400, the system requires at least 20k iterations to generalize well and to produce realistic distribution.
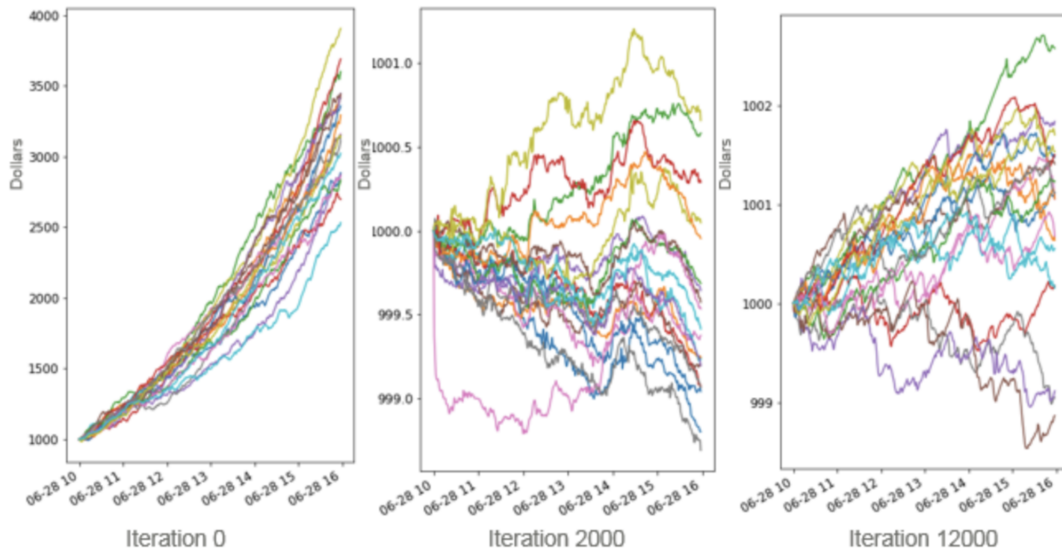
Figure 5: Training progression of generated mid price returns. We visually observe diversity of generated price time series with training progression.
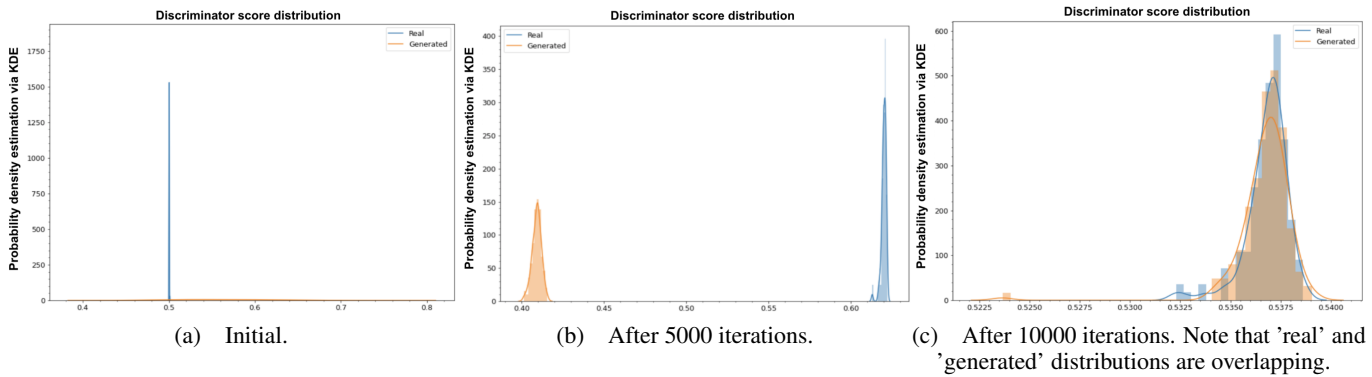

Figure 6: Training progression of distributions of discriminator scores (estimates of the probability density functions from the Kernel Density Estimation). At convergence, we observe an overlap between 'real' and 'generated' data discriminator scores distributions, centered around 0.5 with low discrimination variance. We also passed random noise to the discriminator to validate it - 'random' label.

(a) Historical.

(b) GAN-generated after 1000 iterations.

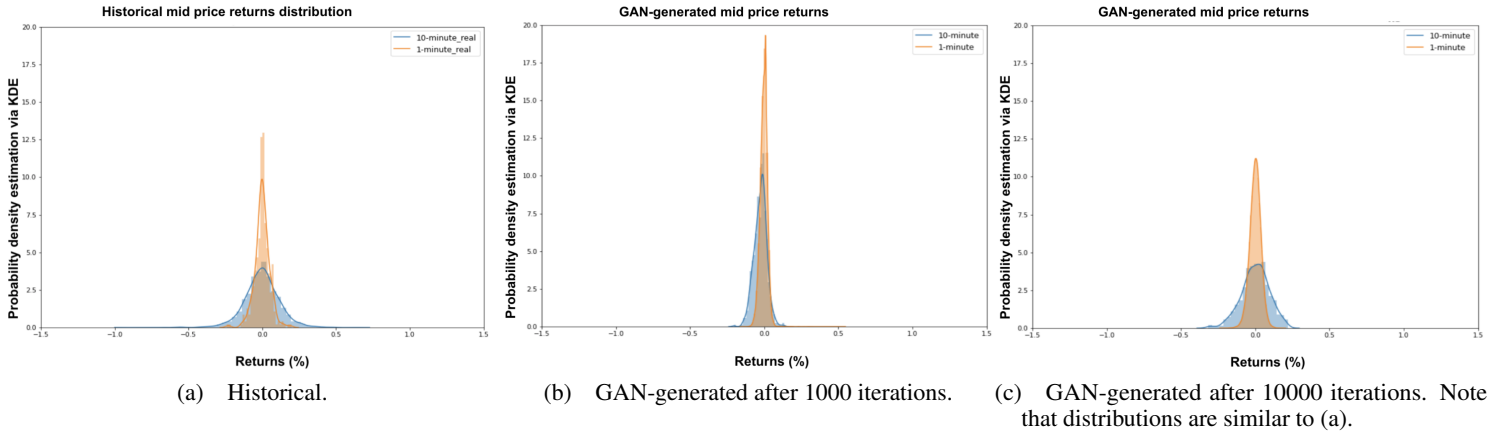(c) GAN-generated after 10000 iterations. Note that distributions are similar to (a).

Figure 7: (a) Historical distributions of mid price returns. (b,c) Training progression of 1- and 10-minute mid price return distributions.
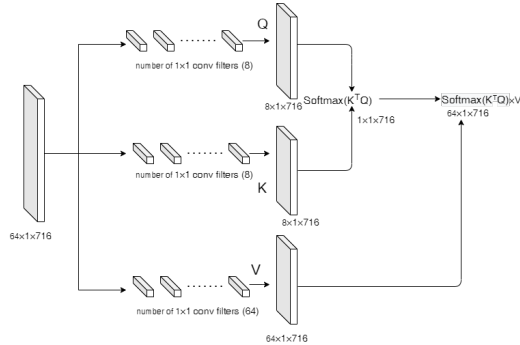

Figure 8: The 1D self-attention mechanism first starts with 1 by 1 convolutional layers along the channel dimension for memory efficiency. Then it follows the standard key, query, values scheme applied to the feature map. Kernel size for the feature spaces f, g is 8 and 8 for feature space h.
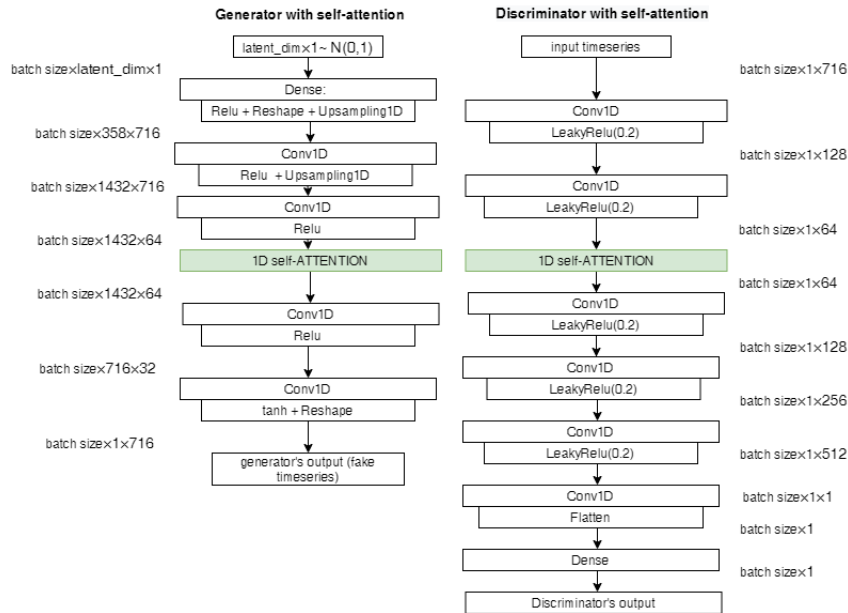

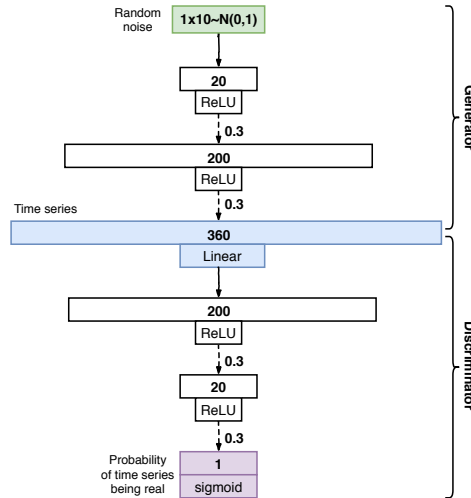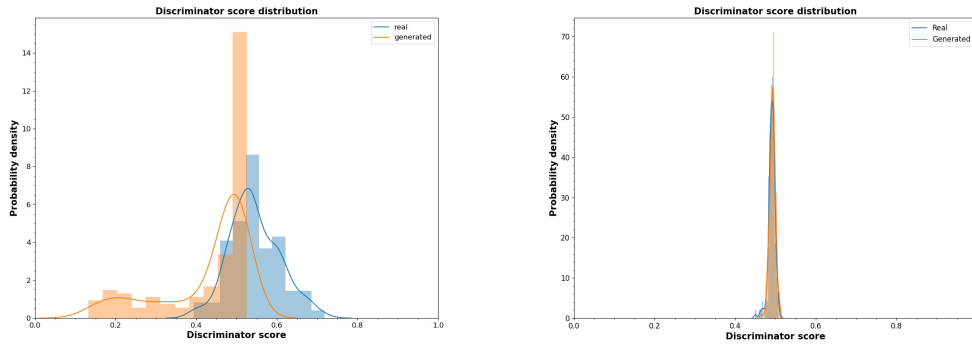Figure 9: Generative adversarial network architecture with self-attention.

Figure 10: Generative adversarial network architecture without self-attention (used for the ablation study). The punctured lines represent dropout layers.

(a) Ablation study: GAN without self-attention. Trained with latent dimension 10 using mid-price returns time series only.

(b) Ablation study: GAN with self-attention. Trained with latent dimension 10 using mid-price returns time series only.

Figure 11: Adding self-attention to GANs reduces the variance of discriminator score distribution and makes it more concentrated around $\frac{1}{2}$ both for real and generator-produced time series

Epoch Number	Latent Dim	K-S Stat	P-Value
20k	100	0.10	0.37
	200	0.18	0.05
	400	0.15	0.06
10k	100	0.10	0.40
	200	0.23	0.06
	400	0.10	0.03
1k	100	0.72	10^{-12}
	200	0.44	10^{-33}
	400	0.65	10^{-20}

Figure 12: Two sample Kolmogorov-Smirnov test. At 10k iterations for a latent dimension of 100, we cannot distinguish between historical discriminator scores distribution and generated one based on the two sample test.