Market-GAN: Adding Control to Financial Market Data Generation with Semantic Context

Haochong Xia, Shuo Sun, Xinrun Wang[*], Bo An[*]

Nanyang Technological University, Singapore
{haochong001, shuo003}@e.ntu.edu.sg, {xinrun.wang, boan}@ntu.edu.sg

Abstract

Financial simulators play an important role in enhancing forecasting accuracy, managing risks, and fostering strategic financial decision-making. Despite the development of financial market simulation methodologies, existing frameworks often struggle with adapting to specialized simulation context. We pinpoint the challenges as i) current financial datasets do not contain context labels; ii) current techniques are not designed to generate financial data with context as control, which demands greater precision compared to other modalities; iii) the inherent difficulties in generating context-aligned, high-fidelity data given the non-stationary, noisy nature of financial data. To address these challenges, our contributions are: i) we proposed the Contextual Market Dataset with market dynamics, stock ticker, and history state as context, leveraging a market dynamics modeling method that combines linear regression and Dynamic Time Warping clustering to extract market dynamics; ii) we present Market-GAN, a novel architecture incorporating a Generative Adversarial Networks (GAN) for the controllable generation with context, an autoencoder for learning low-dimension features, and supervisors for knowledge transfer; iii) we introduce a two-stage training scheme to ensure that Market-GAN captures the intrinsic market distribution with multiple objectives. In the pertaining stage, with the use of the autoencoder and supervisors, we prepare the generator with a better initialization for the adversarial training stage. We propose a set of holistic evaluation metrics that consider alignment, fidelity, data usability on downstream tasks, and market facts. We evaluate Market-GAN with the Dow Jones Industrial Average data from 2000 to 2023 and showcase superior performance in comparison to 4 state-of-the-art time-series generative models.

1 Introduction

Financial simulations play a pivotal role in navigating the complexities of the economic landscape, enabling stakeholders to anticipate market fluctuations, manage risks, and optimize investment strategies (Staum 2002; Chan and Wong 2015; Lopez-Rojas and Axelsson 2016). The growing trend of applying machine learning on price feature data in financial research (Rundo et al. 2019) leads to the need for a simulator that can generate the data with high-fidelity market features. On the other hand, the existing research indicates that there is a non-stationary context in the market (Michael and Johnson 2003; Hens and Schenk-Hoppé 2009). Leveraging the semantic context as control, the market simulation could be more explainable, controllable, and diverse, benefiting downstream applications (Purkayastha, Manolova, and Edelman 2012). This further leads to the need for a context-aligned, high-fidelity market feature generator. However, existing simulators focus on generating Limit Order Book with agent-based model (Samanidou et al. 2007; Axtell and Farmer 2022) and generative model (Takahashi, Chen, and Tanaka-Ishii 2019). While ensemble methods, for example, a multi-expert system can be combined with time-series generative models (Mogren 2016; Yoon, Jarrett, and Van der Schaar 2019; Ni et al. 2020) to generate aligned market feature data, this solution could be complex and inefficient (Mienye and Sun 2022).

To address the problem, we proposed Market-GAN, a controllable generator with semantic context for financial simulation of the market features. Our key contributions are: i) we construct the Contextual Market Dataset with the stock ticker, history state, and market dynamics extracted by a market dynamics modeling algorithm as semantic context, addressing the absence of a financial dataset with context; ii) we propose Market-GAN, an innovative hybrid architecture which is the first contextual generative model for financial market features; iii) we design a two-stage training scheme of pre-training and adversarial training for a better initialization of the generator, to address the mode collapse (Wiatrak, Albrecht, and Nystrom 2019) observed in training complex GAN networks; iv) with a discussion of the evaluation for financial market simulation, we conduct comprehensive experiments on the generated data with the metrics from the perspective of context alignment, fidelity, data usability, and market facts. Market-GAN showcases superior performance compared with 4 benchmark methods.

2 Background and Related Works

Financial Market Simulator. Over the years, simulators have emerged as a valuable tool for studying the behavior of financial markets in a controlled environment. Agent-based model methods are widely applied to simulate the Limit Order Book of financial markets (Samanidou et al.

[*]Corresponding authors

2007; Axtell and Farmer 2022). Recent progress has combined the agent-based model with stochastic models (Shi and Cartlidge 2023). While agent-based models offer insights by simulating individual agent behaviors, they rely heavily on behavior models of agents and empirical market models, which sheds some doubts on the plausibility of using this method to simulate complex market (Gould et al. 2013; Preis et al. 2007; Vyetrenko et al. 2020). While the price feature is an important data source, especially in fundamental and technical analysis as illustrated by (Petrusheva and Jordanoski 2016; Dechow et al. 2001; Gite et al. 2021; Miao 2014), its simulation is also essential.

Time-Series Data Generation. Generated time-series data can be useful in data augmentation or when real data is scarce or sensitive, especially in financial applications, showing its potential for market simulation. Among all the methods, solutions based on GAN gain popularity in recent years. RCGAN (Esteban, Hyland, and Rätsch 2017) introduces RNN with conditional inputs to multi-variant time-series generation via GAN architecture. TimeGAN (Yoon, Jarrett, and Van der Schaar 2019) utilizes a two-stage autoencoder and GAN training scheme to learn goal and local goals together. SigCWGAN (Ni et al. 2020) combines continuous-time stochastic models with its signature metric. Stock-GAN (Takahashi, Chen, and Tanaka-Ishii 2019) is a generative model that generates the order stream instead of the market features. While FIN-GAN (Takahashi, Chen, and Tanaka-Ishii 2019) introduces GAN to generate price features, the use of vanilla GAN is rudimentary compared to the benchmark methods of time-series generative models.

Contextual Generation. Contextual generation refers to the generation of content that is highly relevant given a certain context. Unlike generic content generation, contextual generation ensures that the produced content aligns with the given semantics of the context. This capability is vital for tasks where precision and relevance are paramount, for example, financial simulation. Conditional GAN (Mirza and Osindero 2014) introduces a method to direct the generation process with conditions in GAN architecture. CGMMN (Ren et al. 2016) enables contextual generation based on GMMN (Li, Swersky, and Zemel 2015). More recent works show a greater ability to use semantic context as conditions in various domains, including natural language processing (Platanios et al. 2018; OpenAI 2023), and image generation (Karras et al. 2020; Zhang and Agrawala 2023). While there is a blooming trend in the multi-modality generation using text as context (Saharia et al. 2022; Ramesh et al. 2021; Crowson et al. 2022; Rombach et al. 2022), this paradigm has not been introduced to financial data generation.

3 Contextual Market Dataset

To build the Contextual Market Dataset that addresses the lack of financial datasets with semantic context, we first define the plausible context for the financial market. In the spirit of combining fundamental and technical analysis methods in financial research, we propose a hybrid asset price model $\mathbf{X}_{t,\epsilon} = f(C_{\text{lt}}(t), C_{\text{mt}}(t), C_{\text{st}}(t,\epsilon))$. This model takes into account different time scales and viewpoints to estimate asset prices, which are represented as $\mathbf{X}$ based on three types of context: i) long-term fundamentals $C_{\text{lt}}(t)$ with a time range of multiple years; ii) mid-term market dynamics $C_{\text{mt}}(t)$ with a range of 2 to 6 months; and iii) short-term history with volatility $C_{\text{st}}(t,\epsilon)$ with a range less than 2 months.

Consider a financial market with a set of financial instruments $\mathcal{L}$ and a historical time range of T. We define a consecutive batch of price features in a stock market as $\mathbf{X} \in \mathbb{R}^{T\times F}$, where T is the length of the batch $\mathbf{X}$ and F is the number of features. For a stock market, $\mathbf{X}$ carries the following attributes: i) market dynamics $C_{\text{mt}}(t) = d \in \mathcal{D}$, with $\mathcal{D}$ being the market dynamics space ii) stock ticker $C_{\text{lt}}(t) = l \in \mathcal{L}$, where $\mathcal{L}$ is the stock ticker space iii) near history $C_{\text{st}}(t,\epsilon) = \mathbf{H}_{t,\epsilon} \in \mathbb{R}^{T_H\times F_H}$, with T_H being the length of the history batch and F_H being the number of features, and ϵ is a noise vector sampled from a normal distribution that models the latent volatility in the market. In the Contextual Market Datatset, the real market data R is:

$$R = \{(\mathbf{X}_{t,\epsilon_{X_t}}, \mathbf{H}_{t,\epsilon_{H_t}}, d_t, l_t) | \\ \mathbf{H} \in \text{history of } \mathbf{X}, d_t = d_{X_t}, l_t = l_{X_t}\}, \quad (1)$$

where $\mathbf{X}$ is a batch from the historical data stream maintaining the same dynamics d and stock ticker l.

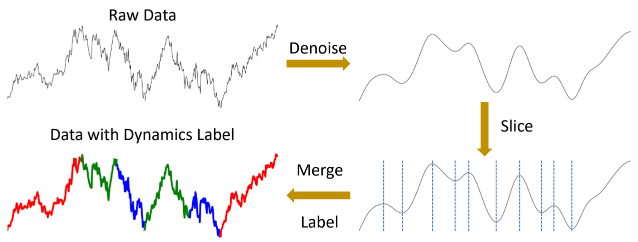


Figure 1: Overview of the market dynamics modeling

3.1 Market Dynamics Modeling

While $\mathbf{H}$ and l have ground truth values associated with each $\mathbf{X}$, the market dynamics d of $\mathbf{X}$ still require mining.

Algorithm 1: Market Dynamic Modeling

Input: Time-series x
Parameter: Number of Dynamics $|\mathcal{D}|$, Minimum Length l_{min}, Threshold θ, Distance Constraint c
Output: Dynamics Label d

```
1: Denoise $x$ with a low-pass filter into $x'$.
2: Slice $x'$ into batches set $B$ by extremums.
3: For each batch $b_i$ in $B$, merge $b_i$ with $b_{i+i}$ length of $b_i < l_{min}$
4: while Clustering not converge do
5:     Calculate the slope of each batch $b_i$ with linear regression models and label their dynamics $d$ into $D$ categories by the percentile of their slope. Merge $b_i$ with $b_{i+1}$ if distance$(b_i, b_{i+1}) < \theta$ and label distance $|d_i - d_{i+1}| \le c$
6: end while
```

In Algorithm 1, the Market Dynamics Modeling (MDM) method integrates the clarity of linear regression with the flexibility of clustering which effectively measures the similarity between two time sequences.

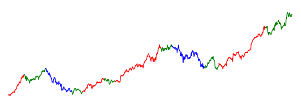
As described in Algorithm 1, the data is first denoised by a low-pass filter, a signal processing technique used for the estimation of a desired signal from an observed noisy signal, to filter out the high-frequency volatility. After the data is denoised, we slice it by extremums into batches B of the shortest market trend that is not dividable. Then, they are merged to reach l_{min}, which is the minimum expected length of a short-term market structure. With the short-term structures, we cluster them into mid-term dynamics d by using the normalized Euclidean distance to measure the similarity of adjacent batches to find the short-term structures while using the slope calculated from the linear regression model to decide the dynamic label of each b_i. An overview of the process is illustrated in Fig 1. Detailed hyper-parameters are in Appendix A.

3.2 Dynamics Modeling Result Analysis

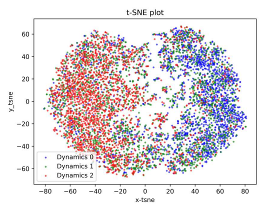
Dynamic	$length$	s	ul	dl
0(bear)	$111(\pm 55)$	$-2.0(\pm 1.6)$	$23(\pm 17)$	$80(\pm 42)$
1(flat)	$145(\pm 132)$	$1.7(\pm 1.0)$	$108(\pm 125)$	$25(\pm 12)$
2(bull)	$175(\pm 136)$	$4.8(\pm 1.8)$	$156(\pm 134)$	$19(\pm 17)$

Table 1: Modeling result of AAPL. Unit of slopes are e^{-3}

The modeling result of AAPL from 2000 to 2023 is shown in Table 1 with the number of dynamics as 3, minimum length of 50, threshold of 0.03, and distance constraint of 1. We evaluate the average length $length$, average slope s by the linear regression model, average maximum uptrend length ul, and maximum downtrend length dl of the three dynamics (bear, flat, bull) we labeled. Results show that i) l falls in the range of a mid-term context of 2 to 5 months; ii) s is aligned with the bear, flat, bull semantic; iii) ul of dynamics 0 and dl of dynamics 2 is far below $l = 50$, indicating that the dynamics should not be further segmented, thus they are correctly labeled.



(a) Data with Dynamics



(b) t-SNE plot

Figure 2: Visulizations result of dynamics modeling where dynamics 0,1,2, is marked as blue, green, and red. (a) illustrates a segment of the Adj close feature of AAPL (1473 days); (b) illustrates the t-SNE plot of 29 stocks (except DOW) in DJI.

Fig 2(a) shows an example of data with the dynamics labeled. The t-SNE plot in Fig 2(b) shows data from dynamics 0 and 2 are in distinct clusters, while data from dynamic 1 spreads in between, representing the bear, flat, and bull markets.

4 Market-GAN

With the Contextual Market Dataset, we describe the generation goal of Market-GAN and its hybrid architecture.

4.1 Problem Formualtion

As semantic context $\mathbf{H}$, d, l, and $\mathbf{X}$ form a one-to-one data-context pair, the limited real data $\mathbf{X}$ would lead to a limited context in R. Given the non-stationary nature of the financial market, the context distribution $p(\mathbf{H}_{t,\epsilon}, d_t, l_t)$ may shift over time, leading to Out Of Distribution (OOD) problem with the context of R only encompasses a subset of all the possible context sets. By utilizing the generative model G, we aim to augment R with the generated data

$$F = \{(\hat{\mathbf{X}}_\epsilon, \mathbf{H}, d, l) | \mathbf{H} \in \mathbb{R}^{T_H \times F_H}, d \in \mathcal{D}, l \in \mathcal{L}, \epsilon \in \mathcal{N}\}, \quad (2)$$

where $\hat{\mathbf{X}}_\epsilon = G(\mathbf{Z}, \mathbf{H}, d, l)$, and F is generalized to any plausible context $\mathbf{H}, d, l$. For simplicity, we omit t and ϵ.

We aim to learn a generative model $G(\mathbf{Z}, \mathbf{H}, d, l) = \mathbf{X}_{t,\epsilon}$ that uses semantic context to control the simulation of the market. G should learn the distribution $\hat{p}(\mathbf{X}, \mathbf{H}, d, l)$ that approximates $p(\mathbf{X}, \mathbf{H}, d, l) = p(\mathbf{H}, d, l) \cdot p(\mathbf{X}|\mathbf{H}, d, l)$ so that the generated F can be controlled by semantic context $\mathbf{C} = (\mathbf{H}, d, l)$. Given the context $\mathbf{C}$ in the generation, the model needs to learn $\hat{p}(\mathbf{X}|\mathbf{H}, d, l)$. With the assumption that $\mathbf{H}$, d, l are disentangled in the representation space and thus independently conditioned on $\mathbf{X}$, the objective can be represented as $p(\mathbf{X}|\mathbf{H}, d, l) \propto p(\mathbf{H}, d, l|\mathbf{X}) = p(\mathbf{H}|\mathbf{X}) \cdot p(d|\mathbf{X}) \cdot p(l|\mathbf{X})$.

As most downstream machine learning tasks assume that financial time-series features follow the Markov property, our model also learns the auto-regressive transaction distribution $p(\mathbf{X}_t|\mathbf{X}_{0:t-1})$, where $\mathbf{X}_t$ is the t-th tick of $\mathbf{X}$ and $\mathbf{X}_{0:t-1}$ is its preceding feature batch. Specifically, since d and l have different semantics compared with $\mathbf{X}$ and $\mathbf{H}$, the auto-regressive transaction distribution can be expressed as $p(\mathbf{X}_t|\mathbf{X}_{0:t-1}, \mathbf{H}, d, l)$.

4.2 Architecture

We follow the autoencoding and GAN structure of TimeGAN (Yoon, Jarrett, and Van der Schaar 2019) and enhance it with data transformation, supervisor teachers, and C-TimesBlock. The raw data $\mathbf{X}$ and context $\mathbf{C}$ is transformed into the encoding space $\mathbf{X}_e$ and $\mathbf{C}_e$ by the transformation layer and then embedded into the latent space $\mathbf{X}_h$ with an embedding network e. We train context classifiers s_d, s_l in encoding space and es_d, es_l in the latent space to capture $p(d|\mathbf{X}_e)$, $p(l|\mathbf{X}_e)$, $p(d|\mathbf{X}_h)$, $p(d|\mathbf{X}_h)$ and transfer this knowledge to the generator g. The generator g is a combination of embedding generator g_e and auto-regression generator g_{ar} where $g(\mathbf{Z}, \mathbf{C}) = g_{ar}(g_e(\mathbf{Z}, \mathbf{C}_e), \mathbf{C}_e)$. With discriminator d, the generator learns to generate the latent $\hat{\mathbf{X}}_h$, while r transforms it to $\hat{\mathbf{X}}_e$ and the inverse transformation

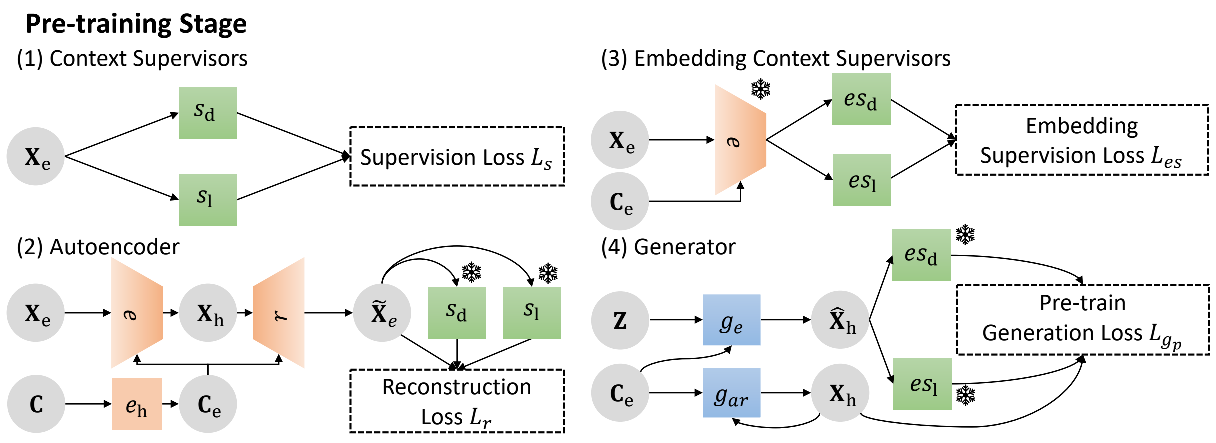


Figure 3: Training scheme of Market-GAN in the pre-training stage. The snowflake indicates the network parameters are frozen in the stage.

layer transforms the encoding back to the market features $\hat{\mathbf{X}}$.

4.3 Data Transformation

The Data Transformation layer transforms the input $\{(\mathbf{X}, \mathbf{H}, d, l)\}$ to an encoded feature $\{(\mathbf{X}_e, h_e, d_e, l_e)\}$. The price features typically follow these patterns: i) key information is carried by deviation, which is relatively small compared to the feature value; ii) the distribution shift accumulates over time, and iii) price features have correlations, causing the following issues in learning: i) the immersion of deviation in the raw data; ii) ineffective data normalization on non-Gaussian long periods of financial data; iii) With $Open$, $High$, Low, $Close$ (OHLC) as price features, violation of the $Low \leq (Open, Close) \leq High$ constraint would make data harmful to downstream tasks. We propose a data transformation layer to resolve these issues. The feature encoding layer reparameters the OHLC features into Low,$Open-Low$,$Close-Low$,$High-\max(Open, Close)$ such that all features of $\mathbf{X}_e$ and $\mathbf{H}_e$ have non-negative deviations. Then, the Low feature is processed into step-wise differential Low_d. The near history $\mathbf{H}_e$ is then used to normalize itself and $\mathbf{X}_e$ to mitigate the distribution shift and simultaneously prevent the leakage of future information to each sample. The embedding network e_h further compress $\mathbf{H}_e$ into h_e. For simplicity, we let $\mathbf{C}_e = \{h_e, d_e, l_e\}$.

4.4 C-TimesBlock

We develop a C-TimesBlock (CTB) that balances time series representation learning and condition modulation. While RNN performs well in a wide range of GAN applications, solely using the RNN in our contextual generation task will result in mode collapse (Wiatrak, Albrecht, and Nystrom 2019), a known challenge in training GANs where the generator produces a limited variety of samples. The TimesBlock (Wu et al. 2023) casts the input into a 2D spectrum and learns with the Inception Blocks, capturing multi-scale patterns of different time periods. By incorporating RNN with TimesBlock, the C-TimesBlock captures the temporal dependency in both 1D and 2D space, mitigating mode collapse. The context alignment scores in our experiment show the superior performance of C-TimesBlock over RNN on context alignment.

5 Training Scheme

Training the generator g from scratch for both context alignment and fidelity generation is challenging, leading to a mode collapse. To alleviate this issue, we design a two-stage training scheme for Market-GAN.

5.1 Pre-training Stage

Using an autoencoder to extract representation and context supervisors as teachers for alignment, the pre-training prepares the generator with a better initialization in adversarial training with discriminator d as shown in Fig 3.

Context Supervisors. We adopt TimesNet (Wu et al. 2023) for context supervisors s_d and s_l. The objective is to reduce the supervision loss $L_s(\mathbf{X}_e)$:

$$\min_{\theta_{s_d}, \theta_{s_l}} CE(d_e, s_d(\mathbf{X}_e)) + CE(l_e, s_l(\mathbf{X}_e)), \tag{3}$$

where CE is the cross-entropy loss.

Autoencoder. We undertake the training of the embedding network e, the embedding reconstruction r, and the history embedding network e_h during this phase. While the embedding network learns the latent representation $\mathbf{X}_h = e(\mathbf{X}_e, \mathbf{C}_e)$, the encoded data is reconstructed by r with $\tilde{\mathbf{X}}_e = r(\mathbf{X}_h, \mathbf{C}_e)$. The objective of this phase is to minimize the reconstruction loss $L_r(\mathbf{X}_e)$:

$$\min_{\theta_e, \theta_r, \theta_{e_h}} ||\tilde{\mathbf{X}}_e - \mathbf{X}_e|| + \gamma(CE(d_e, s_d(\tilde{\mathbf{X}}_e)) + CE(l_e, s_l(\tilde{\mathbf{X}}_e))), \tag{4}$$

where γ is the context alignment weight. By incorporating classification loss into the training objective of the autoencoder, e derives a latent space that emphasizes a robust contextual representation. While e and r together constitute an autoencoder that transform $\mathbf{X}_e$ into the latent $\mathbf{X}_h$, considering that $\mathbf{X}_e$ is a 2D return of the Low feature, the autoencoder can be considered as a factor model (Duan et al. 2022).

We train e_h from scratch in this phase instead of sharing their parameter because the outputs of e_h and e possess different semantic levels. To curb training instability, we freeze e_h in the subsequent training process.

To achieve data reconstruction within the re-parameterized range, we apply $prelu$ activation to the Low_d and $relu$ activation to the remaining three features.

Embedding Context Supervisors. With the $\mathbf{X}_h$ from pre-trained e, we train the embedding dynamics supervisor es_d, embedding stock ticker supervisor es_l. The objective is to minimize the embedding supervision loss $L_{es}(\mathbf{X}_h)$:

$$\min_{\theta_{es_d},\theta_{es_l}} CE(d_e, es_d(\mathbf{X}_h)) + CE(l_e, es_l(\mathbf{X}_h)). \tag{5}$$

Generator. The two components of the generator g are trained with their respective objective. With $\hat{\mathbf{X}}_h = g_e(\mathbf{Z}, \mathbf{C}_e)$, the embedding generator g_e is trained with the supervision of es_d and es_l to align with the context. The auto-regression generator g_{ar} is tasked with learning the distribution $p(\mathbf{X}_{h[t]}|\mathbf{X}_{h[0:t-1]}, \mathbf{C}_e)$ with the real $\mathbf{X}$. The objective is to minimize the pre-train generation loss L_{g_p}:

$$\begin{aligned}\min_{\theta_{g_e},\theta_{g_{ar}}} & CE(d_e, es_d(\hat{\mathbf{X}}_h)) + CE(l_e, es_l(\hat{\mathbf{X}}_h)) \\ & + MSE(\mathbf{X}_{h[1:t]}, g_{ar}(\mathbf{X}_{h[0:t-1]}, \mathbf{C}_e)),\end{aligned} \tag{6}$$

where MSE is the mean-squared error loss.

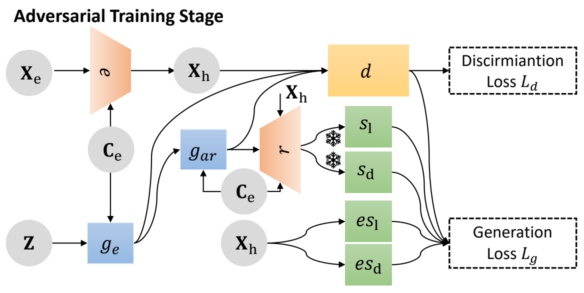


Figure 4: Training scheme of Market-GAN in the adversarial training stage. The snowflake indicates the network parameters of s_l, s_d are frozen in the stage.

5.2 Adversarial Training Stage

As shown in Fig 4, we train the generator g and discriminator d adversarially in this phase with the autoencoder and the supervisors. The discrimination loss is

$$L_d = CE(\mathbf{1}, \mathbf{X}_h) + CE(\mathbf{0}, \hat{\mathbf{X}}_h) + CE(\mathbf{0}, \hat{\mathbf{X}}_{h[T+1]}), \tag{7}$$

where $\hat{\mathbf{X}}_{h[T+1]} = g_{ar}(\hat{\mathbf{X}}_h, \mathbf{C}_e)$. The generation loss is

$$\begin{aligned}L_g = & CE(\mathbf{1}, d(\hat{\mathbf{X}}_h)) + CE(\mathbf{1}, d(\hat{\mathbf{X}}_{h[T+1]})) + \\ & \gamma(L_s(\hat{\mathbf{X}}_e) + L_{es}(\mathbf{X}_h)) + L_r(\mathbf{X}_e),\end{aligned} \tag{8}$$

where we reconstruct $\hat{\mathbf{X}}_e$ by $\hat{\mathbf{X}}_e = r(\hat{\mathbf{X}}_h, \mathbf{C}_e)$, training the Market-GAN with multiple losses jointly. The training objectives for generation and discrimination are

$$\min_{\theta_{g_e},\theta_{g_{ar}},\theta_r,\theta_e,\theta_{es_l},\theta_{es_d}} \max_{\theta_d} L_g, \quad \min_{\theta_d} L_d. \tag{9}$$

While adversarial training is challenging with the complex architecture and multi-tasking objectives, our C-TimesBlock and the multi-stage training scheme help to stabilize the process. With the two-stage training scheme, we train the generative model $G(\mathbf{Z}, \mathbf{H}, d, l) = I(r(g_{ar}(g_e(\mathbf{Z}, \mathbf{C}), \mathbf{C}), \mathbf{C}))$, where I is the inverse transformation layer.

6 Evaluating Generated Data with Context

The main challenge for proposing a robust evaluation metric for financial data is that, unlike images that have an explicit and almost one-to-one explicit, discrete, and fixed semantic of context, the semantics of financial context can be implicit, continuous, and non-stationary. We discuss the evaluation metrics for a contextual generative model of financial data with these research questions:

1. Is generated F aligned to the given context $\mathbf{C}$ while resembling real data R?
2. Can F enhance the performance of downstream tasks?
3. Will F be identified as fake data with definiteness?

Experiments Setting. For the real data R, we generate a corresponding synthetic dataset F, maintaining the same context $\mathbf{C}$. For each instance of $\mathbf{X}$, we generate a $\dot{\mathbf{X}}$ with a random $\mathbf{Z}$, ensuring that R and F have equivalent sizes. While F could be configured to any size, we opt to maintain equality in size between $\mathbf{X}$ and F during evaluation to uphold the fairness of the contrast experiment. We conducted the experiments on a 4090 GPU. Detailed descriptions of the training setups can be found in the Appendix.

Context Alignment. Following the spirit of using CLIP score (Hessel et al. 2021) for evaluation of context alignment, we utilized pre-trained condition supervisors s_d and s_l to classify the d and l of F with cross-entropy loss $L_d = CE_{s_d}(F)$ and $L_l = CE_{s_l}(F)$ as metrics.

Generation Fidelity. Following the spirit of using FID (Heusel et al. 2017) to evaluate the fidelity of generated images, the fidelity of F is evaluated using discriminators, d_e, trained via TimeNets. The objective is defined as:

$$\min_{\theta_{d_e}} CE(\mathbf{1}, R) + CE(\mathbf{0}, F). \tag{10}$$

The accuracy of d_e should be 50% on the test set if it is not distinguishable. Hence, the discrepancy between R and F is evaluated by $L_D = |accuracy_{d_e} - 50|$. To give d_e stronger discrimination ability, we train multi-expert $d_{e(i,j)}$ where each discriminator is trained to classify a subset of data where $d = i$ and $l = j$ and use the average L_D of all experts as the metric with 50 training epochs.

Data Usability. With the real set R and the generated set F, we have the augmented set $R+F$. In the spirit of TSTR (Esteban, Hyland, and Rätsch 2017), the usability of the data is assessed by examining the performance of the one-step prediction task using the augmented training set, where we applied four prominent time-series forecasting models: TimesNet, TCN, LSTM, and GRU. For a fair comparison, we half the training epoch when using $R + F$ as a training set compared with training only with R. We train multi-expert predictors $p_{e(i,j)}$ where each predictor is trained on a subset of data where $d = i$ and $l = j$, and test $p_{e(i,j)}$ on the respective test set of real data. We calculate the Symmetric Mean Absolute Percentage Error (SMAPE) losses of prediction value to evaluate the usability of F in improving the downstream task's performance. The TimesNet predictor is trained with 50 epochs, while TCN, LSTM, and GRU are trained with 200 epochs.

Market Facts. We evaluate if the generated data adheres to the facts of $OHLC$. L_f is the percentage of generated data violating the $Low \leq (Open, Close) \leq High$. Any data that contradicts this fact is definitively identified as fake.

7 Experiments

7.1 Benchmarks and Datasets

We utilized the daily OHLC features of 29 stocks (excluding DOW due to insufficient data) from the DJI index, from January 2000 to June 2023. This is a large dataset that guarantees generalization. We compared our model with representative generative models for time-series generation tasks including TimeGAN, SigCWGAN, CGMMN, and RCGAN. In the experimental setup, $\mathbf{X}$ represents a continuous segment of the price feature possessing identical d and l attributes and a length of 30 while $D = 3$ and $L = 29$. $\mathbf{H}$ is the preceding segment of features relative to $\mathbf{X}$, also with a length of 30.

Multi-expert Benchmarks. As not all benchmarks are designed for contextual generation, we applied a multi-expert paradigm when training the benchmark models for fairness. We train and evaluate a respective model $M_{i,j}$ of each benchmark method with the sub-dataset $R_{i,j} = \{(\mathbf{X}, \mathbf{H}, d, l) | d = i, l = j\}$, with $\mathbf{H}$ as the conditional input for the benchmarks. We show the result as the average score of all the experts of a model.

7.2 Ablation Study

Data Transformation Layer. We evaluate the benchmarks with the application of the data transformation layer and inverse transformation in the pipeline. The results are included in the Table 4 and the brackets of Table 2, 3, and 5.

C-TimesBlock. We compare the performance of Market-GAN using RNN as building blocks in networks versus using C-TimesBlock (CTB).

7.3 Quantitative Results

Model	L_d	L_l
Real Data	0.055	0.023
SigCWGAN	2.758(2.370)	0.594(0.468)
TimeGAN	2.847(1.839)	0.533(0.490)
CGMMN	2.415(2.181)	0.560(0.489)
RCGAN	3.276(2.279)	0.574(0.492)
Market-GAN(CTB)	**0.023**	**0.100**
Market-GAN(RNN)	0.626	0.501

Table 2: Alignment Result (results in the bracket are applied with the data transformation layer).

Context Alignment. As shown in Table 3, Market-GAN with C-TimesBlock outperforms baseline experts on L_d and L_l significantly. In addition, CTB outperforms RNN when used in Market-GAN. A L_d which is lower than that of the real test data indicates Market-GAN with CTB has learned the latent semantic of market dynamics based on the coarse semantic d label by the market dynamics model. With the autoencoder mining factors, Market-GAN successfully aligns the dynamics semantic to F. While we observe a steady improvement of both L_d and L_l on baseline methods when applied with the data transformation layer, the alignment result is still unacceptable. According to the result, we can confirm that the Market-GAN architecture, C-TimeBlock, and the data transformation layer contribute to a better context alignment.

Model	L_D
SigCWGAN	$9.28 \pm 6.11(7.54 \pm 3.72)$
TimeGAN	$5.67 \pm 2.53(6.52 \pm 5.25)$
CGMMN	12.43 ± 7.37(**4.12 ± 2.30**)
RCGAN	$6.01 \pm 2.85(7.07 \pm 6.68)$
Market-GAN(CTB)	8.05 ± 5.60
Market-GAN(RNN)	11.85 ± 9.89

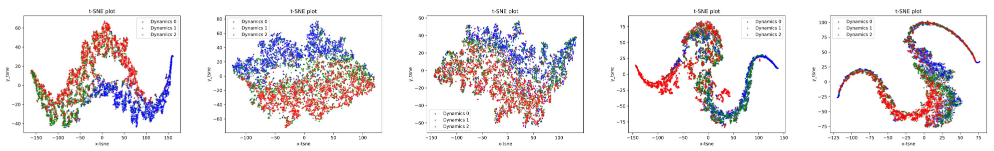
Table 3: Fidelity Result (results in the bracket are applied with the data transformation layer).

Fidelity. We train 87 discriminators with 3 dynamics and 29 stock tickers whose result distribution is shown in Table 3. Market-GAN with CTB got the 3rd lowest L_D among the 5 compared methods. For our contextual generation task, alignment to control semantics and replicate R can be adversary objectives. We observe that while the data transformation layer reduces L_d and L_l, applying it also increases the L_D of TimeGAN and RCGAN, whose L_D is lower than that of Market-GAN. This phenomenon replicates the FID, CLIP score trade-off which has been observed in text-to-image generation (Chang et al. 2023). While our generation goal is beyond letting $R = F$, minimizing L_D isn't the ultimate goal. Mark-GAN outperforms the benchmark methods by improving the context alignment while maintaining decent fidelity. Given these considerations, it is acceptable that Market-GAN doesn't achieve the minimum L_D.

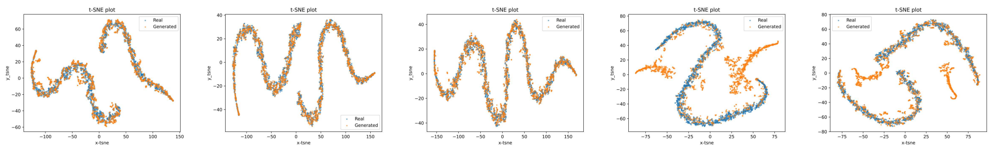
Training Set Data	TimesNet	TCN	LSTM	GRU
Only Real Data	1.76 ± 0.71	6.63 ± 0.76	20.9 ± 20.9	17.2 ± 17.9
SigCWGAN w/o TF	2.71 ± 8.07	15.0 ± 26.3	19.1 ± 34.8	17.6 ± 33.7
SigCWGAN w TF	5.64 ± 12.7	18.1 ± 22.8	24.0 ± 25.0	20.7 ± 23.7
TimeGAN w/o TF	1.73 ± 0.72	6.59 ± 0.70	19.5 ± 20.1	16.6 ± 17.5
TimeGAN w TF	1.36 ± 0.48	6.52 ± 0.72	19.1 ± 19.8	14.8 ± 16.3
CGMMN w/o TF	1.74 ± 4.98	8.92 ± 12.3	11.1 ± 19.4	8.74 ± 16.5
CGMMN w TF	1.38 ± 0.50	6.53 ± 0.73	19.1 ± 19.8	14.8 ± 16.3
RCGAN w/o TF	1.68 ± 0.67	6.64 ± 0.83	16.6 ± 18.1	13.9 ± 15.5
RCGAN w TF	1.40 ± 0.55	6.70 ± 1.91	19.2 ± 19.7	15.0 ± 16.2
Market-GAN(CTB)	1.26 ± 0.75	6.48 ± 0.73	12.9 ± 16.7	11.1 ± 14.2
Market-GAN(RNN)	**1.11±0.66**	**6.39±0.92**	**9.89±14.9**	**8.41±12.4**

Table 4: Prediction SMAPE loss on the test set using generated data of different models to augment the training set. TF is the abbreviation of the data transformation layer.

Data Usability. We train 87 predictors with 3 dynamics and 29 stock tickers, whose results are shown in Table 4. While



(a) Market-GAN (b) TimeGAN (c) RCGAN (d) CGMMN (e) SigCWGAN



(f) Market-GAN (g) TimeGAN (h) RCGAN (i) CGMMN (j) SigCWGAN

Figure 5: t-SNE visualizations. Top row: t-SNE plot where blue, green, and red marks data of dynamics 0,1,2. Bottom row: t-SNE plot of real data R as blue dots with F as orange dots. The Market-GAN uses the C-TimesBlock.

using generated data from benchmark methods results in a prediction loss that is comparable to or worse than using only real data, the generated data from Market-GAN consistently lowers the prediction loss on the test set. Market-GAN outperforms the benchmarks across all forecasting models, getting the lowest prediction loss on the test set, showcasing its effectiveness and reliability when applied to the downstream prediction task.

Model	L_f
Real Data	0
SigCWGAN	0.211(0)
TimeGAN	0.150(0)
CGMMN	0.501(0)
RCGAN	0.247(0)
Market-GAN(CTB)	**0**
Market-GAN(RNN)	**0**

Table 5: Market Facts Result

Market Facts. As shown in Table 5, Market-GAN guarantees 0 violation of the facts on data with its constrained generation pipeline while baseline methods don't capture the fact without the data transformation layer.

7.4 Qualitative Results

With research on limitations of the FID (Parmar, Zhang, and Zhu 2022) and CLIP score (Radford et al. 2021), recent works (Saharia et al. 2022) rely on human expert ratings instead of quantitative metric. However, due to the complexity of financial data, rating the fidelity and context alignment of Market-GAN with human experts is not plausible. As a counterpart, we visualize data with t-SNE plots.

Context Alignment. We plot the t-SNE plot of F where the data of different dynamics is marked respectively as illustrated in the top row of Fig 5 as a qualitative evaluation of the L_d. The t-SNE plot of R, as shown in Fig 2(b), shows a separated cluster of data from dynamics 0 and 2 while data from dynamics 1 is spread among the two clusters. While this pattern is replicated by the benchmarks to a different extent, we observe that F of Market-GAN has three distinct clusters, indicating that it discovers a more disentangled representation of d, resolving mode collapse and leading to a more diverse generated F, corresponding to the numerical result that Market-GAN has a lower L_d than real data R.

Comparison of R and F. We visualize the F with R with t-SNE plots as shown in the bottom row of Fig 5. This graphical representation provides further insights into the sources of L_D loss. i) While CGMMN and RCGAN have high L_D values (12.43 and 9.28) their t-SNE plot shows there are F clusters that are separated from the R clusters as outliers; ii) TimeGAN and RCGAN have low L_D values (5.67 and 6.01) suggesting successful reconstruction of R. Thus, their F overlaps with R in visualization; iii) Market-GAN, with a moderate L_D (8.05), does not contain significant outliers F clusters from R. Instead, the points in F that are not overlapping with R are still adjacent to the cluster. With the discussion of a low than-real L_D of Market-GAN, these non-overlapping F points could be markets with extreme dynamics of d that does not present in the historical data. With a strong ability to control generation with context, Market-GAN is able to simulate extreme markets (Orlowski 2012) which could be useful in downstream financial applications.

8 Conclusion

In this research, we present a financial simulator Market-GAN with the Contextual Market Dataset. With the innovative hybrid model devised for the contextual generation of financial data, Market-GAN surpasses existing methods in generating context alignment data which can improve downstream task performance while maintaining fidelity. Looking ahead, our model offers extensive potential for enhancements and broadened applications. One such possibility is the integration of more fundamental factors beyond stock tickers and accommodating financial data of varied structures and scales. Additionally, the introduction of downstream tasks to the pipeline can foster end-to-end learning. In essence, Market-GAN not only sets a new benchmark in the generation of financial data but also heralds exciting prospects for the evolution of this model, paving the way for the next generation of financial simulation.

Acknowledgments

This project is supported by the National Research Foundation, Singapore under its Industry Alignment Fund – Prepositioning (IAF-PP) Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

Axtell, R. L.; and Farmer, J. D. 2022. Agent-based modeling in economics and finance: Past, present, and future. *Journal of Economic Literature*.

Chan, N. H.; and Wong, H. Y. 2015. *Simulation Techniques in Financial Risk Management*. John Wiley & Sons.

Chang, H.; Zhang, H.; Barber, J.; Maschinot, A.; Lezama, J.; Jiang, L.; Yang, M.-H.; Murphy, K.; Freeman, W. T.; Rubinstein, M.; et al. 2023. Muse: Text-to-image generation via masked generative transformers. *arXiv preprint arXiv:2301.00704*.

Crowson, K.; Biderman, S.; Kornis, D.; Stander, D.; Hallahan, E.; Castricato, L.; and Raff, E. 2022. Vqgan-clip: Open domain image generation and editing with natural language guidance. In *European Conference on Computer Vision*, 88–105. Springer.

Dechow, P. M.; Hutton, A. P.; Meulbroek, L.; and Sloan, R. G. 2001. Short-sellers, fundamental analysis, and stock returns. *Journal of financial Economics*, 61(1): 77–106.

Duan, Y.; Wang, L.; Zhang, Q.; and Li, J. 2022. Factorvae: A probabilistic dynamic factor model based on variational autoencoder for predicting cross-sectional stock returns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 4, 4468–4476.

Esteban, C.; Hyland, S. L.; and Rätsch, G. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*.

Gite, S.; Khatavkar, H.; Kotecha, K.; Srivastava, S.; Maheshwari, P.; and Pandey, N. 2021. Explainable stock prices prediction from financial news articles using sentiment analysis. *PeerJ Computer Science*, 7: e340.

Gould, M. D.; Porter, M. A.; Williams, S.; McDonald, M.; Fenn, D. J.; and Howison, S. D. 2013. Limit Order Books. arXiv:1012.0349.

Hens, T.; and Schenk-Hoppé, K. R. 2009. *Handbook of Financial Markets: Dynamics and Evolution*. Elsevier.

Hessel, J.; Holtzman, A.; Forbes, M.; Bras, R. L.; and Choi, Y. 2021. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in Neural Information Processing Systems*, 30.

Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2020. Analyzing and improving the image quality of StyleGAN. arXiv:1912.04958.

Li, Y.; Swersky, K.; and Zemel, R. 2015. Generative moment matching networks. arXiv:1502.02761.

Lopez-Rojas, E. A.; and Axelsson, S. 2016. A review of computer simulation for fraud detection research in financial datasets. In *2016 Future Technologies Conference (FTC)*, 932–935. IEEE.

Miao, G. J. 2014. High frequency and dynamic pairs trading based on statistical arbitrage using a two-stage correlation and cointegration approach. *International Journal of Economics and Finance*, 6(3): 96–110.

Michael, F.; and Johnson, M. 2003. Financial market dynamics. *Physica A: Statistical Mechanics and its Applications*, 320: 525–534.

Mienye, I. D.; and Sun, Y. 2022. A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects. *IEEE Access*, 10: 99129–99149.

Mirza, M.; and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Mogren, O. 2016. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904*.

Ni, H.; Szpruch, L.; Wiese, M.; Liao, S.; and Xiao, B. 2020. Conditional sig-wasserstein gans for time series generation. *arXiv preprint arXiv:2006.05421*.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Orlowski, L. T. 2012. Financial crisis and extreme market risks: Evidence from Europe. *Review of Financial Economics*, 21(3): 120–130.

Parmar, G.; Zhang, R.; and Zhu, J.-Y. 2022. On Aliased Resizing and Surprising Subtleties in GAN Evaluation. arXiv:2104.11222.

Petrusheva, N.; and Jordanoski, I. 2016. Comparative analysis between the fundamental and technical analysis of stocks. *Journal of Process Management and New Technologies*, 4(2): 26–31.

Platanios, E. A.; Sachan, M.; Neubig, G.; and Mitchell, T. 2018. Contextual parameter generation for universal neural machine translation. *arXiv preprint arXiv:1808.08493*.

Preis, T.; Golke, S.; Paul, W.; and Schneider, J. J. 2007. Statistical analysis of financial returns for a multiagent order book model of asset trading. *Phys. Rev. E*, 76: 016108.

Purkayastha, S.; Manolova, T. S.; and Edelman, L. F. 2012. Diversification and performance in developed and emerging market contexts: A review of the literature. *International Journal of Management Reviews*, 14(1): 18–38.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 8748–8763.

Ramesh, A.; Pavlov, M.; Goh, G.; Gray, S.; Voss, C.; Radford, A.; Chen, M.; and Sutskever, I. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, 8821–8831.

Ren, Y.; Zhu, J.; Li, J.; and Luo, Y. 2016. Conditional generative moment-matching networks. *Advances in Neural Information Processing Systems*, 29.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10684–10695.

Rundo, F.; Trenta, F.; di Stallo, A. L.; and Battiato, S. 2019. Machine learning for quantitative finance applications: A survey. *Applied Sciences*, 9(24): 5574.

Saharia, C.; Chan, W.; Saxena, S.; Li, L.; Whang, J.; Denton, E. L.; Ghasemipour, K.; Gontijo Lopes, R.; Karagol Ayan, B.; Salimans, T.; et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35: 36479–36494.

Samanidou, E.; Zschischang, E.; Stauffer, D.; and Lux, T. 2007. Agent-based models of financial markets. *Reports on Progress in Physics*, 70(3): 409.

Shi, Z.; and Cartlidge, J. 2023. Neural Stochastic Agent-Based Limit Order Book Simulation: A Hybrid Methodology. *arXiv preprint arXiv:2303.00080*.

Staum, J. 2002. Simulation in financial engineering. In *Proceedings of the Winter Simulation Conference*, volume 2, 1481–1492.

Takahashi, S.; Chen, Y.; and Tanaka-Ishii, K. 2019. Modeling financial time-series with generative adversarial networks. *Physica A: Statistical Mechanics and its Applications*, 527: 121261.

Vyetrenko, S.; Byrd, D.; Petosa, N.; Mahfouz, M.; Dervovic, D.; Veloso, M.; and Balch, T. 2020. Get real: Realism metrics for robust limit order book market simulations. In *Proceedings of the First ACM International Conference on AI in Finance*, 1–8.

Wiatrak, M.; Albrecht, S. V.; and Nystrom, A. 2019. Stabilizing generative adversarial networks: A survey. *arXiv preprint arXiv:1910.00927*.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-variation modeling for general time series analysis. arXiv:2210.02186.

Yoon, J.; Jarrett, D.; and Van der Schaar, M. 2019. Time-series generative adversarial networks. *Advances in Neural Information Processing Systems*, 32.

Zhang, L.; and Agrawala, M. 2023. Adding conditional control to text-to-image diffusion models. arXiv:2302.05543.