# TimeGAN

Literature Review

Tom Dooney

# Contents

# Motivation

- For sequential data, = (, ..., ), wish to model p(|)
- GANs do not adequately attend to the temporal correlations in time-series data
  - Adversarial objective of GANs seeks to model p() directly, no autoregressive prior
  - Summing standard GAN loss over sequences of vectors may not be sufficient to ensure that the network efficiently captures stepwise dependencies
- Supervised models for sequence prediction allow finer control over network dynamics
  - inherently deterministic
- TimeGAN framework combines the the control afforded by supervised AR models with flexibility of unsupervised GANs (RNN critic and generator)

# Related work

- Deterministic methods
  - Autoregressive recurrent networks
  - Professor Forcing
  - Actor-Critic methods

- GANs in temporal settings
  - Using LSTM networks for generator and discriminator
  - Data is generated recurrently, input noise vector and data generated from previous time step
  - RCGAN – drop dependence on previous output and condition on additional input
  - Approaches rely only on the binary adversarial feedback for learning

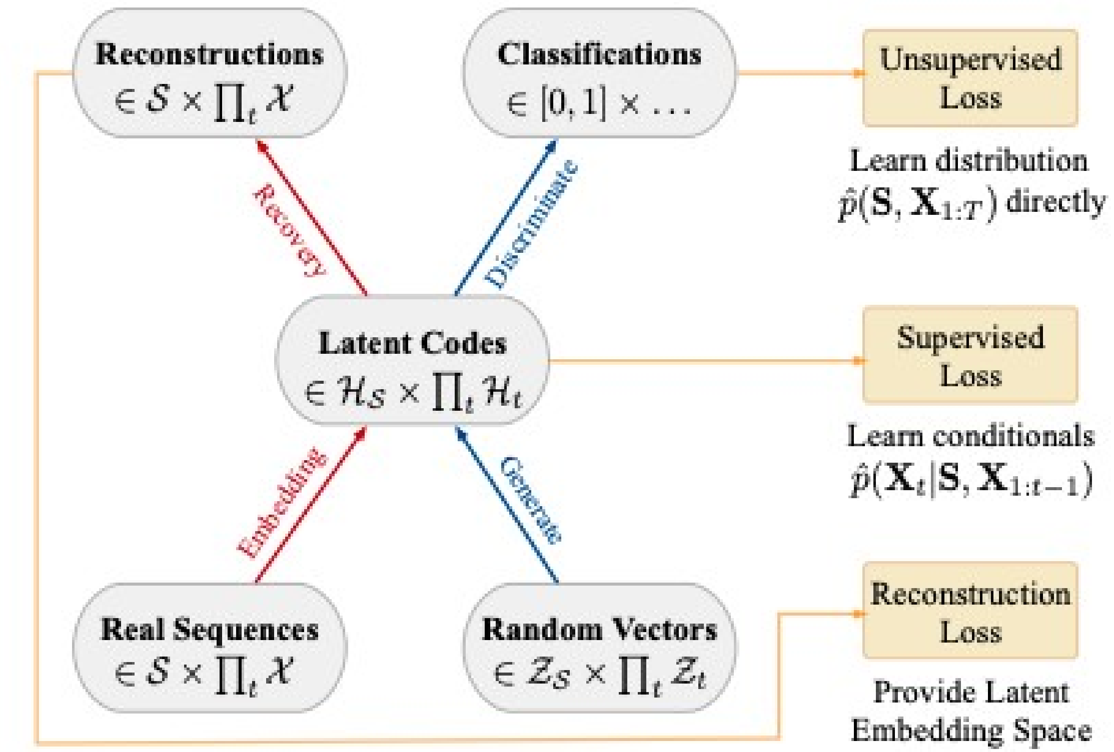
- Representation Learning
  - Learning compact encodings for the benefit of downstream tasks

# Methods

- TimeGANs: Supervised AR + Unsupervised GANs + Time Series Representation
- Unsupervised adversarial loss on real and synthetic sequences
- Addition of stepwise supervised loss using the original data as supervision
- Incorporates stochasticity at each time step
- Embedding network reduces the high-dimensionality of the adversarial learning space.
  - Provides a reversible mapping between features and latent representations
  - Generative model learns stepwise distributions and latent dynamics in lower-dimensional space
  - Temporal dynamics of even complex systems often driven by lower-dimensional factors of variation

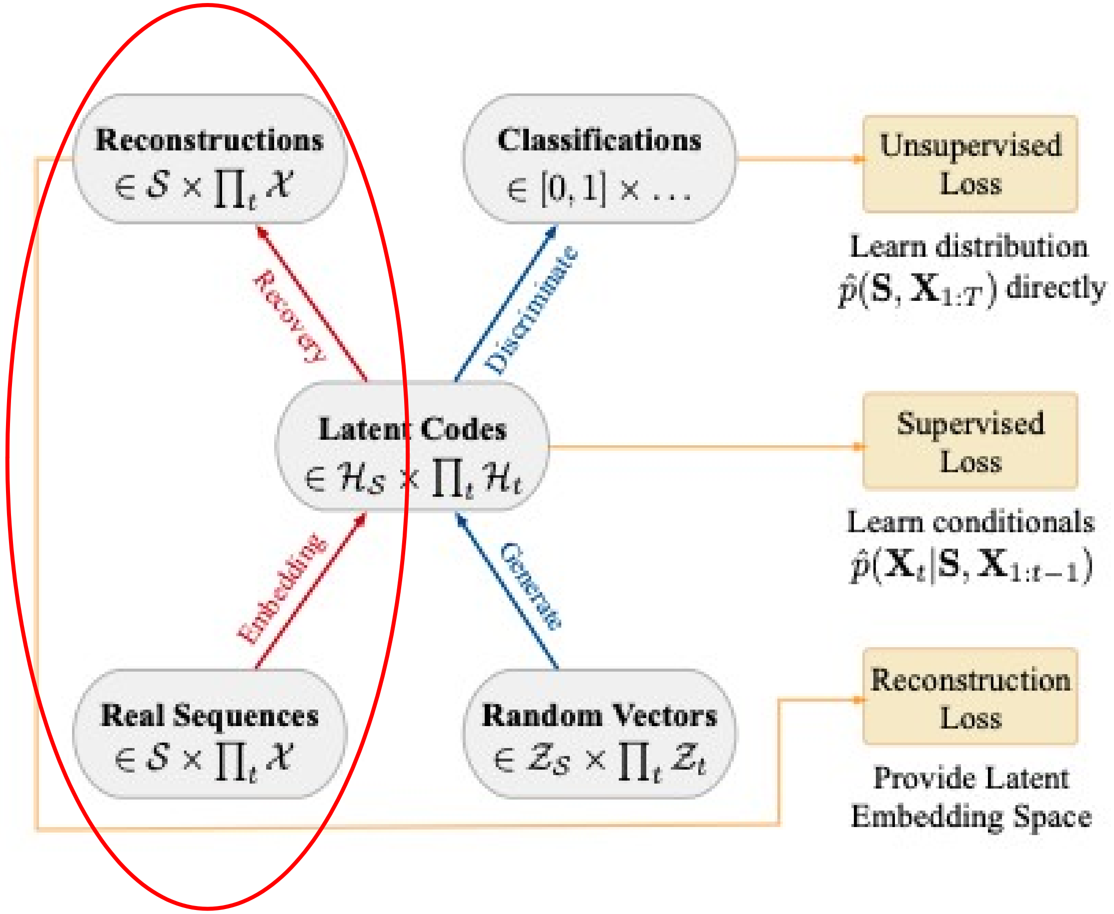
# TimeGAN Model Overview

- Four network components:



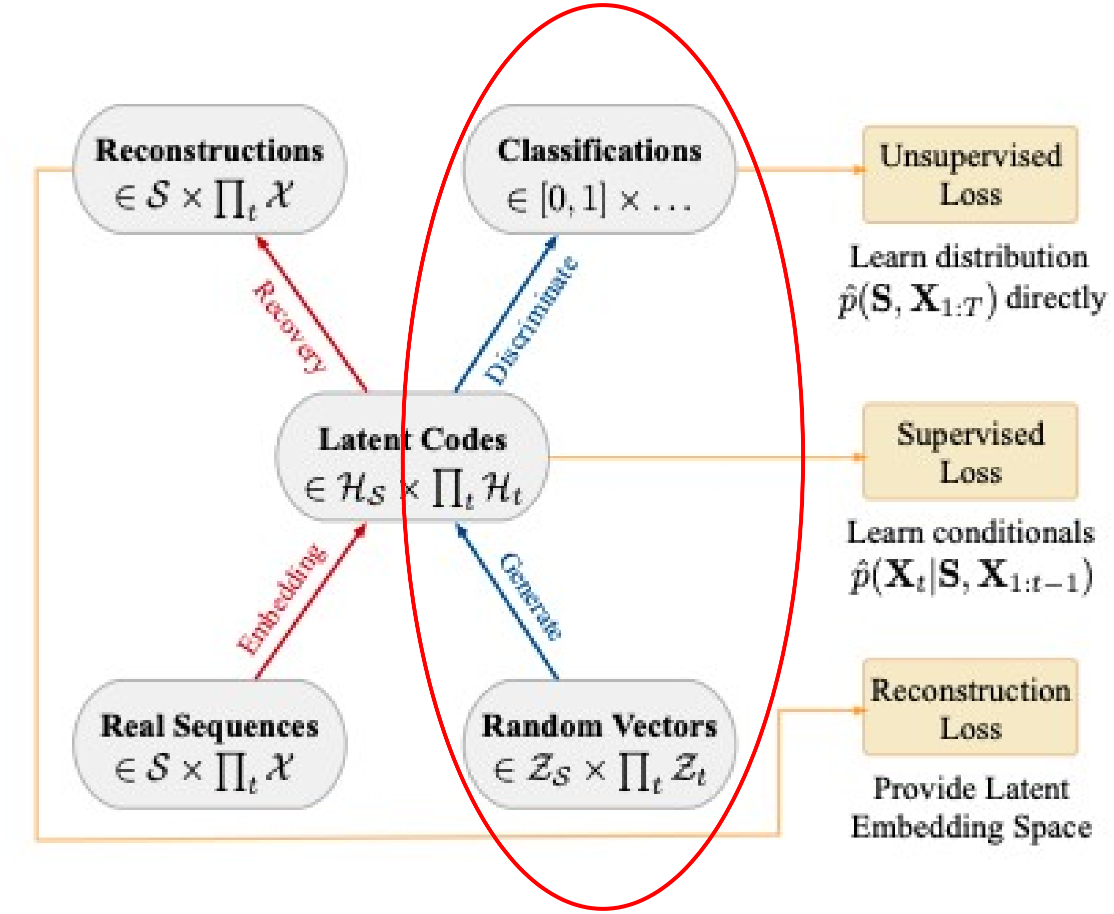
# TimeGAN Model Overview

- Four network components:
  - Autoencoding components: embedding function, recovery function



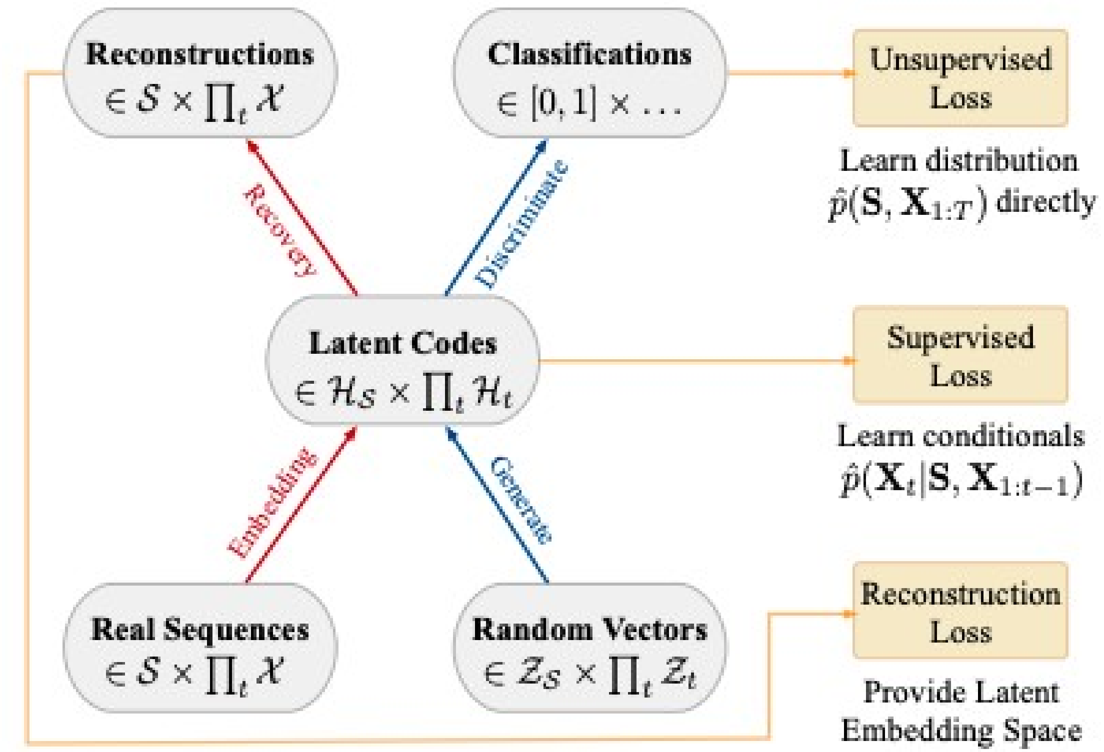
# TimeGAN Model Overview

- Four network components:
  - Autoencoding components: embedding function, recovery function
  - Adversarial components: sequence generator, and sequenc discriminator.



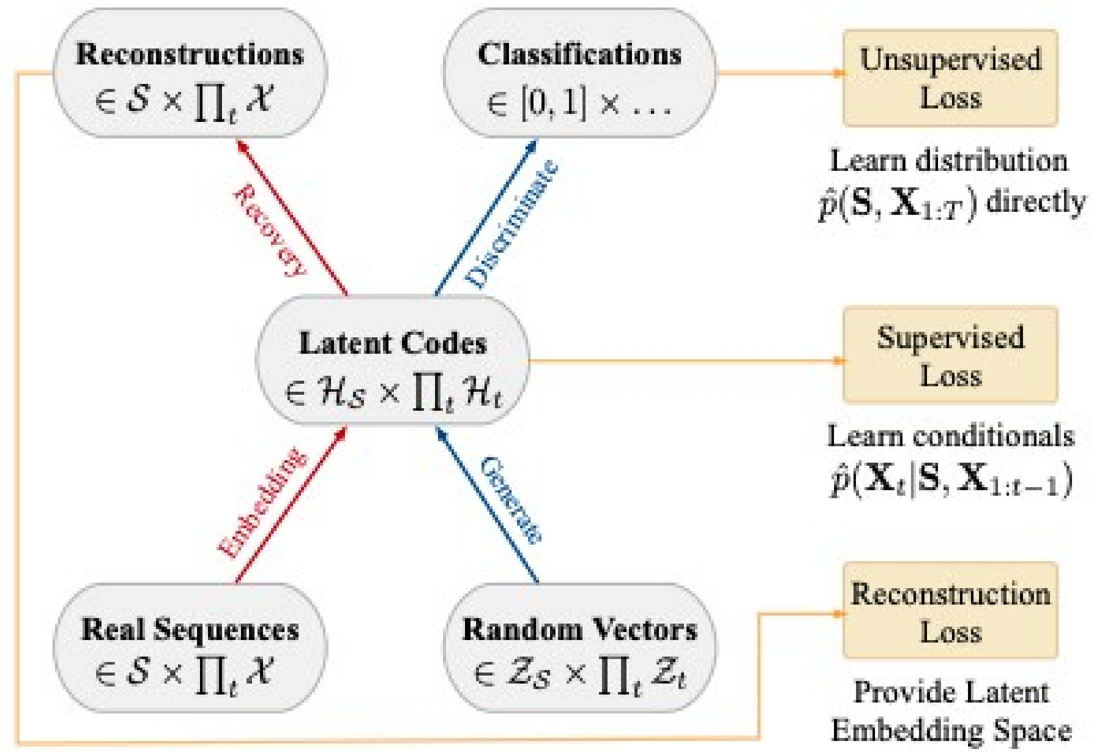
# TimeGAN Model Overview

- Four network components:
  - Autoencoding components: embedding function, recovery function
  - Adversarial components: sequence generator, and sequence discriminator.
- Autoencoding and adversarial components trained jointly
  - Simultaneously learn to encode features, generate representations, and iterate across time.



# TimeGAN Model Overview

- Four network components:
  - Autoencoding components: embedding function, recovery function
  - Adversarial components: sequence generator, and sequenc discriminator.
- Autoencoding and adversarial components trained jointly
  - Simultaneously learn to encode features, generate representations, and iterate across time.




- The embedding network provides the latent space, the adversarial network operates within this space, and the latent dynamics of both real and synthetic data are synchronized through a supervised loss.

# Autoencoding Components

- Embedding network takes separately:
  - Static features ***S*** (eg. Gender) in Static Embedding Network
  - Temporal features ***X*** in Recurrent embedding network

$$\mathbf{h}_{\mathcal{S}} = e_{\mathcal{S}}(\mathbf{s}) \qquad \mathbf{h}_t = e_{\mathcal{X}}(\mathbf{h}_{\mathcal{S}}, \mathbf{h}_{t-1}, \mathbf{x}_t)$$

- Recovery network takes separately static and temporal features, both in feedforward networks at each step

$$\tilde{\mathbf{s}} = r_{\mathcal{S}}(\mathbf{h}_s) \qquad \tilde{\mathbf{x}}_t = r_{\mathcal{X}}(\mathbf{h}_t)$$

- Embedding and recovery functions can be parameterized by any architecture of choice
  - Must be autoregressive and obey causal ordering
  - (i.e. output(s) at each step can only depend on preceding information)
  - It is just as possible to implement the former with temporal convolutions, or the latter via an attention-based decoder

# Adversarial Components

- RNN generator outputs to embedding space
  - generator network for static features
  - recurrent generator for temporal features

$$\hat{\mathbf{h}}_{\mathcal{S}} = g_{\mathcal{S}}(\mathbf{z}_{\mathcal{S}}) \qquad \hat{\mathbf{h}}_t = g_{\mathcal{X}}(\hat{\mathbf{h}}_{\mathcal{S}}, \hat{\mathbf{h}}_{t-1}, \mathbf{z}_t)$$

- Random vector sampled from Gaussian distribution
- follows a stochastic process (Wiener process)

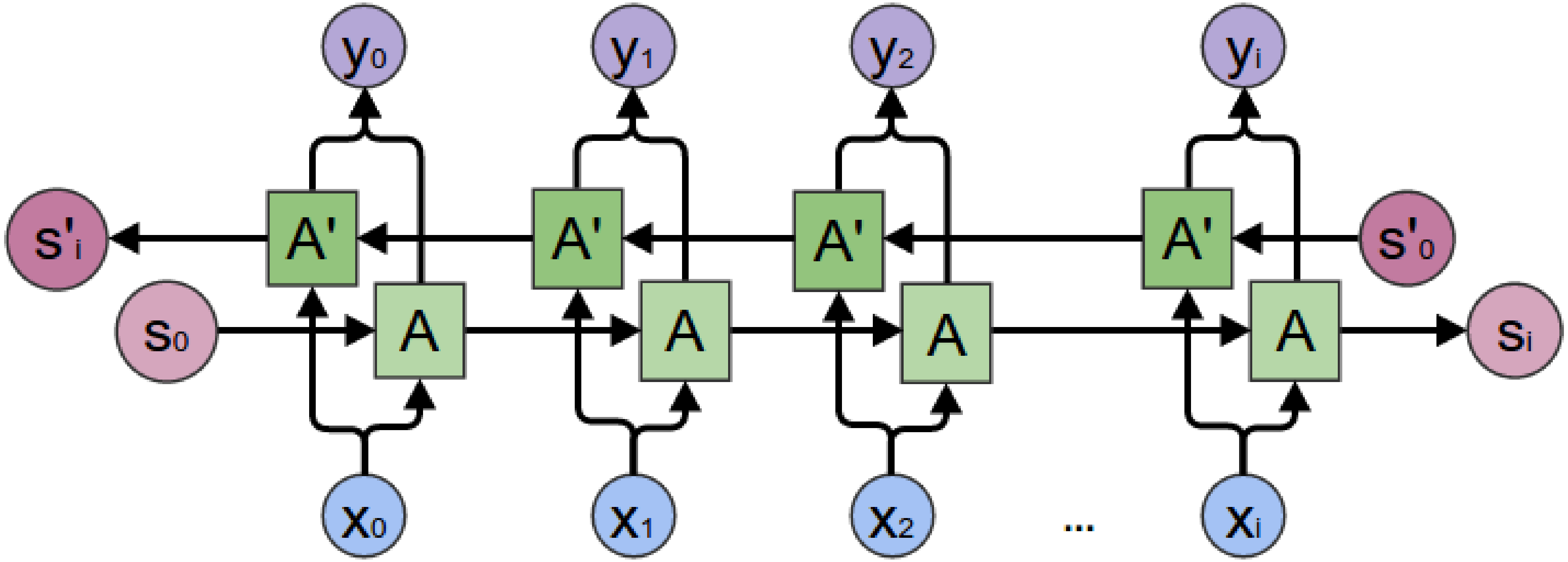
- Discriminator also operates from the embedding space
  - Receives the static and temporal codes, returning classifications $\tilde{y}_{\mathcal{S}}, \tilde{y}_{1:T} = d(\tilde{\mathbf{h}}_{\mathcal{S}}, \tilde{\mathbf{h}}_{1:T})$
  - *d* implemented via bidirectional recurrent network with a feedforward output layer

$$\tilde{y}_{\mathcal{S}} = d_{\mathcal{S}}(\tilde{\mathbf{h}}_{\mathcal{S}}) \qquad \tilde{y}_t = d_{\mathcal{X}}(\overleftarrow{\mathbf{u}}_t, \vec{\mathbf{u}}_t)$$

  - *With forward and backward hidden states*

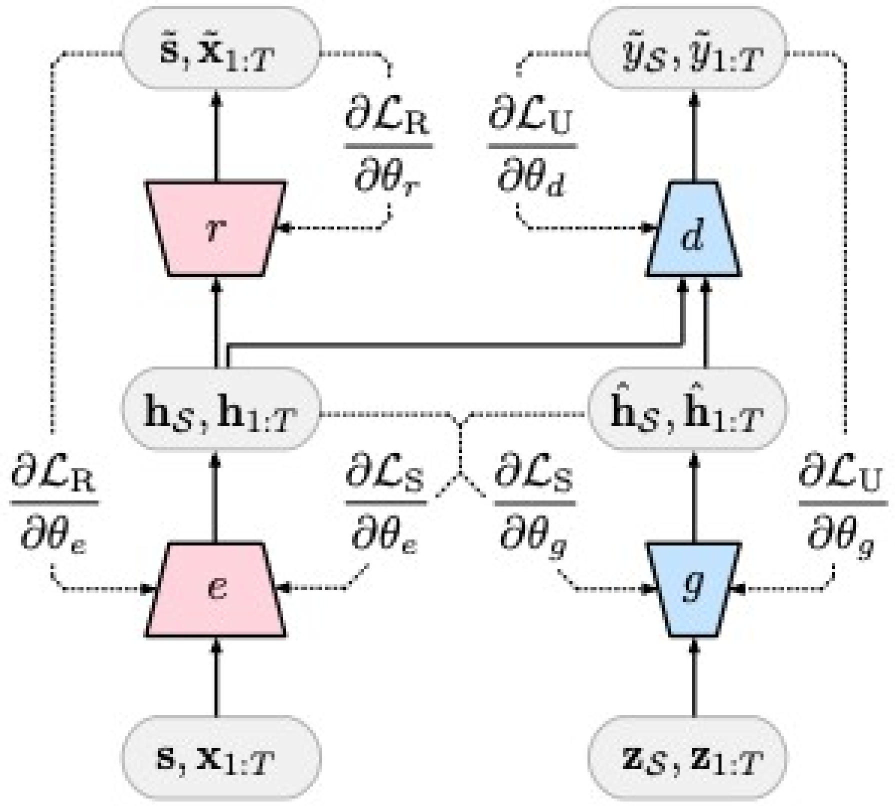
$$\vec{\mathbf{u}}_t = \vec{c}_{\mathcal{X}}(\tilde{\mathbf{h}}_{\mathcal{S}}, \tilde{\mathbf{h}}_t, \vec{\mathbf{u}}_{t-1}) \qquad \overleftarrow{\mathbf{u}}_t = \overleftarrow{c}_{\mathcal{X}}(\tilde{\mathbf{h}}_{\mathcal{S}}, \tilde{\mathbf{h}}_t, \overleftarrow{\mathbf{u}}_{t+1})$$

# Bi-Directional RNN



# Jointly Learning to Encode, Generate, and Iterate

- Require accurate reversible mappings between features and latent spaces
  - Reconstruction loss
- (Autoregressive) Generator receives two different types of inputs in alternating fashion
  1. Open-loop mode - previous synthetic embedding output to generate next vector
  2. Closed-loop mode – embeddings of actual data (embedding network) to generate next latent vector
- Open-loop
  - Gradients computed on Unsupervised loss (as one expects with GANs)
  - Maximizing (discriminator) or minimizing (generator) likelihood of providing correct classification for both training data and synthetic outputs
- Closed-loop
  - Gradients can be computed on loss that captures the discrepancy between distributions
  - Applying maximum likelihood yields the familiar Supervised loss
  - At each step during training, assess difference between actual next-step latent vector (from embedding function) and synthetic next-step latent vector (from generator conditioned on actual historical sequence of latents)



# Jointly Learning to Encode, Generate, and Iterate

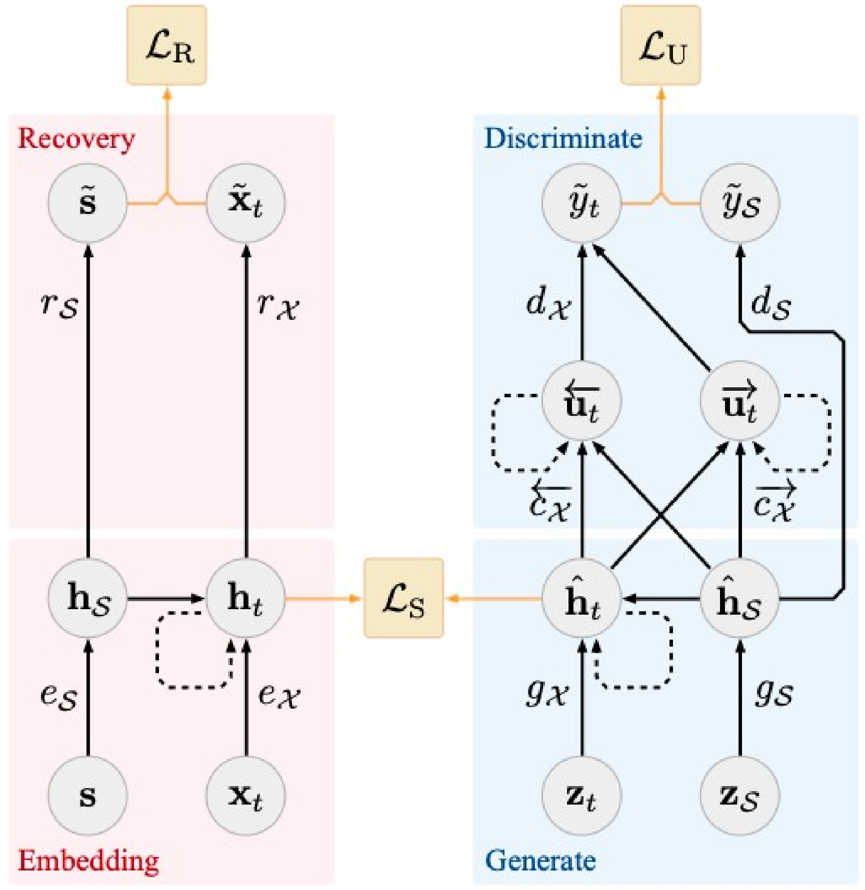
- ensures accurate reconstructions
  - Evaluated by input
- pushes generator to create realistic sequences
  - Evaluated by an imperfect adversary
- further ensures that it produces similar stepwise transitions
  - Evaluated by ground-truth targets
- Autoencoding components trained on both the reconstruction and supervised losses

$$\min_{\theta_e,\theta_r}(\lambda\mathcal{L}_\text{S} + \mathcal{L}_\text{R})$$

- Adversarial components are trained adversarially

$$\min_{\theta_g}(\eta\mathcal{L}_\text{S} + \max_{\theta_d}\mathcal{L}_\text{U})$$

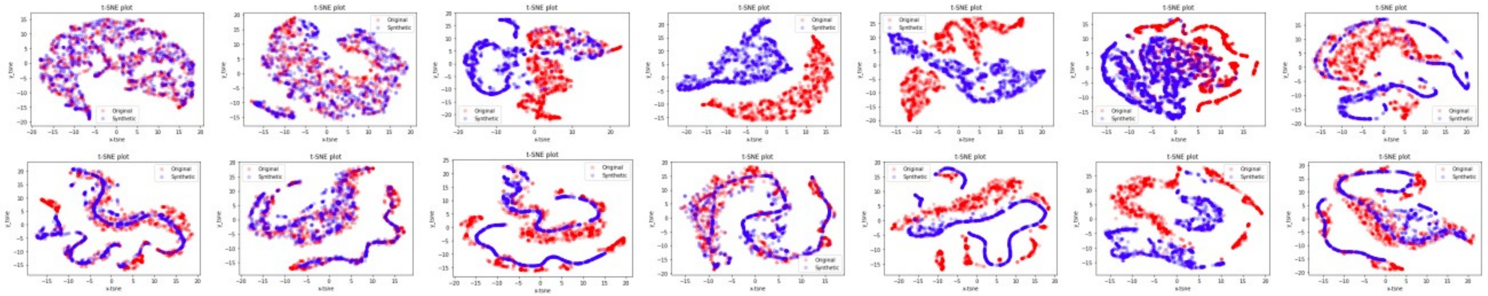
- TimeGAN is not sensitive to λ and η



# Experiments

- Three criteria to assess quality – Diversity, fidelity, usefulness
- Diversity – samples should be distributed to cover the real data
  - PCA and TSNE analyses
  - visualizes how closely the distribution of generated samples resembles that of the original in 2-dimensional space
- Fidelity – samples should be indistinguishable from the real data
  - Discriminative Score, quantitative measure of similarity
  - train a post-hoc time-series classification model (by optimizing a 2-layer LSTM) to distinguish between sequences from the original and generated datasets (2-class supervised task), reporting error on hold-out set.
- Usefulness – samples should be just as useful as the real data when used for the same predictive purposes (i.e. train-on-synthetic, test-on-real)
  - Predictive score, sampled data should inherit the predictive characteristics of the original
  - Train a post-hoc sequence-prediction model (by optimizing a 2-layer LSTM) to predict next-step temporal vectors, evaluate the trained model on the original dataset
  - Performance is measured in terms of the mean absolute error (MAE);

# Experiments - Diversity



(a) TimeGAN (b) RCGAN (c) CRNNGAN (d) T-Forcing (e) P-Forcing (f) WaveNet (g) WaveGAN

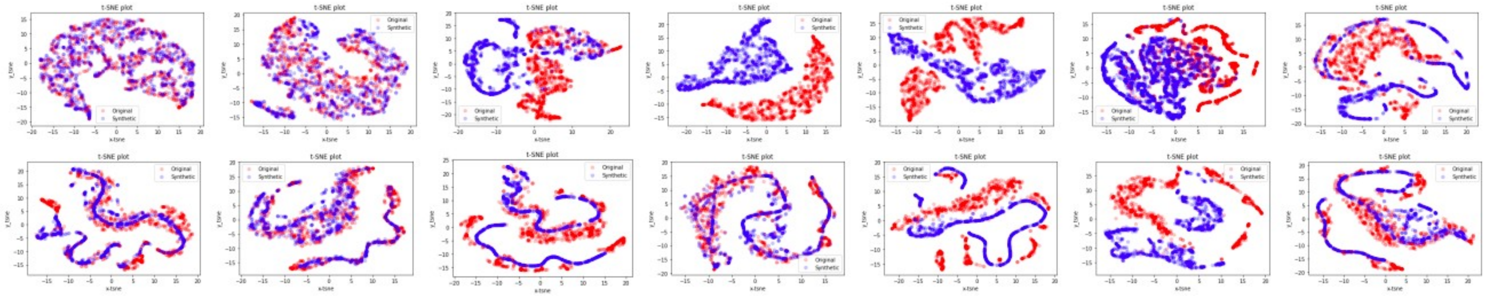
Table 1: Results on Autoregressive Multivariate Gaussian Data (Bold indicates best performance).

| | Temporal Correlations (fixing $\sigma = 0.8$) | | | Feature Correlations (fixing $\phi = 0.8$) | | |
|---|---|---|---|---|---|---|
| Settings | $\phi = 0.2$ | $\phi = 0.5$ | $\phi = 0.8$ | $\sigma = 0.2$ | $\sigma = 0.5$ | $\sigma = 0.8$ |
| Discriminative Score (Lower the better) | | | | | | |
| TimeGAN | **.175±.006** | **.174±.012** | **.105±.005** | **.181±.006** | **.152±.011** | **.105±.005** |
| RCGAN | .177±.012 | .190±.011 | .133±.019 | .186±.012 | .190±.012 | .133±.019 |
| C-RNN-GAN | .391±.006 | .227±.017 | .220±.016 | .198±.011 | .202±.010 | .220±.016 |
| T-Forcing | .500±.000 | .500±.000 | .499±.001 | .499±.001 | .499±.001 | .499±.001 |
| P-Forcing | .498±.002 | .472±.008 | .396±.018 | .460±.003 | .408±.016 | .396±.018 |
| WaveNet | .337±.005 | .235±.009 | .229±.013 | .217±.010 | .226±.011 | .229±.013 |
| WaveGAN | .336±.011 | .213±.013 | .230±.023 | .192±.012 | .205±.015 | .230±.023 |
| Predictive Score (Lower the better) | | | | | | |
| TimeGAN | **.640±.003** | **.412±.002** | **.251±.002** | **.282±.005** | **.261±0.002** | **.251±.002** |
| RCGAN | .652±.003 | .435±.002 | .263±.003 | .292±.003 | .279±.002 | .263±.003 |
| C-RNN-GAN | .696±.002 | .490±.005 | .299±.002 | .293±.005 | .280±.006 | .299±.002 |
| T-Forcing | .737±.022 | .732±.012 | .503±.037 | .515±.034 | .543±.023 | .503±.037 |
| P-Forcing | .665±.004 | .571±.005 | .289±.003 | .406±.005 | .317±.001 | .289±.003 |
| WaveNet | .718±.002 | .508±.003 | .321±.005 | .331±.004 | .297±.003 | .321±.005 |
| WaveGAN | .712±.003 | .489±.001 | .290±.002 | .325±.003 | .353±.001 | .290±.002 |

Table 2: Results on Multiple Time-Series Datasets (Bold indicates best performance).

| Metric | Method | Sines | Stocks | Energy | Events |
|---|---|---|---|---|---|
| Discriminative Score (Lower the Better) | TimeGAN | **.011±.008** | **.102±.021** | **.236±.012** | **.161±.018** |
| | RCGAN | .022±.008 | .196±.027 | .336±.017 | .380±.021 |
| | C-RNN-GAN | .229±.040 | .399±.028 | .499±.001 | .462±.011 |
| | T-Forcing | .495±.001 | .226±.035 | .483±.004 | .387±.012 |
| | P-Forcing | .430±.027 | .257±.026 | .412±.006 | .489±.001 |
| | WaveNet | .158±.011 | .232±.028 | .397±.010 | .385±.025 |
| | WaveGAN | .277±.013 | .217±.022 | .363±.012 | .357±.017 |
| Predictive Score (Lower the Better) | TimeGAN | **.093±.019** | **.038±.001** | **.273±.004** | **.303±.006** |
| | RCGAN | .097±.001 | .040±.001 | .292±.005 | .345±.010 |
| | C-RNN-GAN | .127±.004 | **.038±.000** | .483±.005 | .360±.010 |
| | T-Forcing | .150±.022 | **.038±.001** | .315±.005 | .310±.003 |
| | P-Forcing | .116±.004 | .043±.001 | .303±.006 | .320±.008 |
| | WaveNet | .117±.008 | .042±.001 | .311±.005 | .333±.004 |
| | WaveGAN | .134±.013 | .041±.001 | .307±.007 | .324±.006 |
| | Original | .094±.001 | .036±.001 | .250±.003 | .293±.000 |

# Experiments - Fidelity



(a) TimeGAN (b) RCGAN (c) CRNNGAN (d) T-Forcing (e) P-Forcing (f) WaveNet (g) WaveGAN

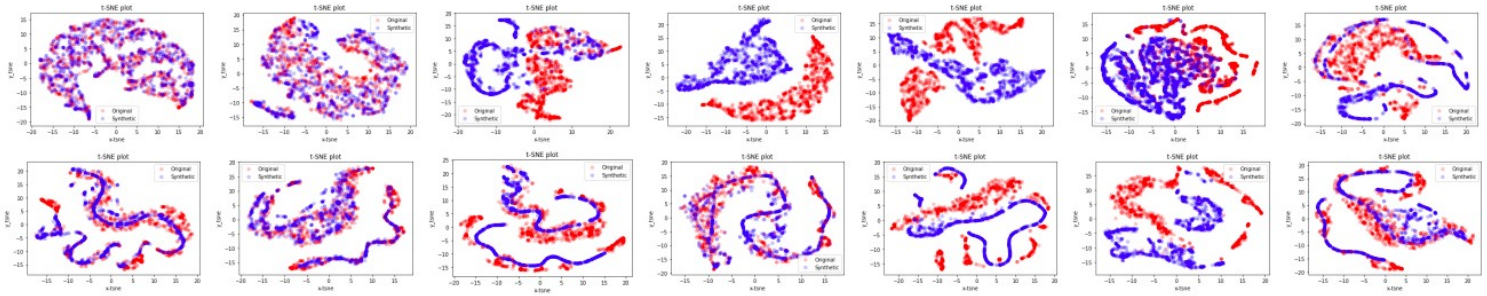
Table 1: Results on Autoregressive Multivariate Gaussian Data (Bold indicates best performance).

| | Temporal Correlations (fixing $\sigma = 0.8$) | | | Feature Correlations (fixing $\phi = 0.8$) | | |
|---|---|---|---|---|---|---|
| Settings | $\phi = 0.2$ | $\phi = 0.5$ | $\phi = 0.8$ | $\sigma = 0.2$ | $\sigma = 0.5$ | $\sigma = 0.8$ |
| Discriminative Score (Lower the better) | | | | | | |
| TimeGAN | **.175±.006** | **.174±.012** | **.105±.005** | **.181±.006** | **.152±.011** | **.105±.005** |
| RCGAN | .177±.012 | .190±.011 | .133±.019 | .186±.012 | .190±.012 | .133±.019 |
| C-RNN-GAN | .391±.006 | .227±.017 | .220±.016 | .198±.011 | .202±.010 | .220±.016 |
| T-Forcing | .500±.000 | .500±.000 | .499±.001 | .499±.001 | .499±.001 | .499±.001 |
| P-Forcing | .498±.002 | .472±.008 | .396±.018 | .460±.003 | .408±.016 | .396±.018 |
| WaveNet | .337±.005 | .235±.009 | .229±.013 | .217±.010 | .226±.011 | .229±.013 |
| WaveGAN | .336±.011 | .213±.013 | .230±.023 | .192±.012 | .205±.015 | .230±.023 |
| Predictive Score (Lower the better) | | | | | | |
| TimeGAN | **.640±.003** | **.412±.002** | **.251±.002** | **.282±.005** | **.261±0.002** | **.251±.002** |
| RCGAN | .652±.003 | .435±.002 | .263±.003 | .292±.003 | .279±.002 | .263±.003 |
| C-RNN-GAN | .696±.002 | .490±.005 | .299±.002 | .293±.005 | .280±.006 | .299±.002 |
| T-Forcing | .737±.022 | .732±.012 | .503±.037 | .515±.034 | .543±.023 | .503±.037 |
| P-Forcing | .665±.004 | .571±.005 | .289±.003 | .406±.005 | .317±.001 | .289±.003 |
| WaveNet | .718±.002 | .508±.003 | .321±.005 | .331±.004 | .297±.003 | .321±.005 |
| WaveGAN | .712±.003 | .489±.001 | .290±.002 | .325±.003 | .353±.001 | .290±.002 |

Table 2: Results on Multiple Time-Series Datasets (Bold indicates best performance).

| Metric | Method | Sines | Stocks | Energy | Events |
|---|---|---|---|---|---|
| Discriminative Score (Lower the Better) | TimeGAN | **.011±.008** | **.102±.021** | **.236±.012** | **.161±.018** |
| | RCGAN | .022±.008 | .196±.027 | .336±.017 | .380±.021 |
| | C-RNN-GAN | .229±.040 | .399±.028 | .499±.001 | .462±.011 |
| | T-Forcing | .495±.001 | .226±.035 | .483±.004 | .387±.012 |
| | P-Forcing | .430±.027 | .257±.026 | .412±.006 | .489±.001 |
| | WaveNet | .158±.011 | .232±.028 | .397±.010 | .385±.025 |
| | WaveGAN | .277±.013 | .217±.022 | .363±.012 | .357±.017 |
| Predictive Score (Lower the Better) | TimeGAN | **.093±.019** | **.038±.001** | **.273±.004** | **.303±.006** |
| | RCGAN | .097±.001 | .040±.001 | .292±.005 | .345±.010 |
| | C-RNN-GAN | .127±.004 | **.038±.000** | .483±.005 | .360±.010 |
| | T-Forcing | .150±.022 | **.038±.001** | .315±.005 | .310±.003 |
| | P-Forcing | .116±.004 | .043±.001 | .303±.006 | .320±.008 |
| | WaveNet | .117±.008 | .042±.001 | .311±.005 | .333±.004 |
| | WaveGAN | .134±.013 | .041±.001 | .307±.007 | .324±.006 |
| | Original | .094±.001 | .036±.001 | .250±.003 | .293±.000 |

# Experiments - Usefulness



(a) TimeGAN (b) RCGAN (c) CRNNGAN (d) T-Forcing (e) P-Forcing (f) WaveNet (g) WaveGAN

Table 1: Results on Autoregressive Multivariate Gaussian Data (Bold indicates best performance).

| Settings | Temporal Correlations (fixing $\sigma = 0.8$) $\phi = 0.2$ | $\phi = 0.5$ | $\phi = 0.8$ | Feature Correlations (fixing $\phi = 0.8$) $\sigma = 0.2$ | $\sigma = 0.5$ | $\sigma = 0.8$ |
|---|---|---|---|---|---|---|
| Discriminative Score (Lower the better) | | | | | | |
| TimeGAN | **.175±.006** | **.174±.012** | **.105±.005** | **.181±.006** | **.152±.011** | **.105±.005** |
| RCGAN | .177±.012 | .190±.011 | .133±.019 | .186±.012 | .190±.012 | .133±.019 |
| C-RNN-GAN | .391±.006 | .227±.017 | .220±.016 | .198±.011 | .202±.010 | .220±.016 |
| T-Forcing | .500±.000 | .500±.000 | .499±.001 | .499±.001 | .499±.001 | .499±.001 |
| P-Forcing | .498±.002 | .472±.008 | .396±.018 | .460±.003 | .408±.016 | .396±.018 |
| WaveNet | .337±.005 | .235±.009 | .229±.013 | .217±.010 | .226±.011 | .229±.013 |
| WaveGAN | .336±.011 | .213±.013 | .230±.023 | .192±.012 | .205±.015 | .230±.023 |
| Predictive Score (Lower the better) | | | | | | |
| TimeGAN | **.640±.003** | **.412±.002** | **.251±.002** | **.282±.005** | **.261±0.002** | **.251±.002** |
| RCGAN | .652±.003 | .435±.002 | .263±.003 | .292±.003 | .279±.002 | .263±.003 |
| C-RNN-GAN | .696±.002 | .490±.005 | .299±.002 | .293±.005 | .280±.006 | .299±.002 |
| T-Forcing | .737±.022 | .732±.012 | .503±.037 | .515±.034 | .543±.023 | .503±.037 |
| P-Forcing | .665±.004 | .571±.005 | .289±.003 | .406±.005 | .317±.001 | .289±.003 |
| WaveNet | .718±.002 | .508±.003 | .321±.005 | .331±.004 | .297±.003 | .321±.005 |
| WaveGAN | .712±.003 | .489±.001 | .290±.002 | .325±.003 | .353±.001 | .290±.002 |

Table 2: Results on Multiple Time-Series Datasets (Bold indicates best performance).

| Metric | Method | Sines | Stocks | Energy | Events |
|---|---|---|---|---|---|
| Discriminative Score (Lower the Better) | TimeGAN | **.011±.008** | **.102±.021** | **.236±.012** | **.161±.018** |
| | RCGAN | .022±.008 | .196±.027 | .336±.017 | .380±.021 |
| | C-RNN-GAN | .229±.040 | .399±.028 | .499±.001 | .462±.011 |
| | T-Forcing | .495±.001 | .226±.035 | .483±.004 | .387±.012 |
| | P-Forcing | .430±.027 | .257±.026 | .412±.006 | .489±.001 |
| | WaveNet | .158±.011 | .232±.028 | .397±.010 | .385±.025 |
| | WaveGAN | .277±.013 | .217±.022 | .363±.012 | .357±.017 |
| Predictive Score (Lower the Better) | TimeGAN | **.093±.019** | **.038±.001** | **.273±.004** | **.303±.006** |
| | RCGAN | .097±.001 | .040±.001 | .292±.005 | .345±.010 |
| | C-RNN-GAN | .127±.004 | **.038±.000** | .483±.005 | .360±.010 |
| | T-Forcing | .150±.022 | **.038±.001** | .315±.005 | .310±.003 |
| | P-Forcing | .116±.004 | .043±.001 | .303±.006 | .320±.008 |
| | WaveNet | .117±.008 | .042±.001 | .311±.005 | .333±.004 |
| | WaveGAN | .134±.013 | .041±.001 | .307±.007 | .324±.006 |
| | Original | .094±.001 | .036±.001 | .250±.003 | .293±.000 |

# Experiments – Sources of gain

- Analyze importance of each contribution, report the discriminative and predictive scores with the following modifications
  1. without the supervised loss,
  2. without the embedding networks
  3. without jointly training the embedding and adversarial networks on the supervised loss.
- (The first corresponds to λ = η = 0, and the third to λ = 0).

$$\min_{\theta_e,\theta_r}(\lambda\mathcal{L}_{\mathrm{S}} + \mathcal{L}_{\mathrm{R}}) \qquad \min_{\theta_g}(\eta\mathcal{L}_{\mathrm{S}} + \max_{\theta_d}\mathcal{L}_{\mathrm{U}})$$

Table 3: Source-of-Gain Analysis on Multiple Datasets (via Discriminative and Predictive scores).

| Metric | Method | Sines | Stocks | Energy | Events |
|---|---|---|---|---|---|
| Discriminative Score (Lower the Better) | TimeGAN | **.011±.008** | **.102±.021** | **.236±.012** | **.161±.018** |
| | w/o Supervised Loss | .193±.013 | .145±.023 | .298±.010 | .195±.013 |
| | w/o Embedding Net. | .197±.025 | .260±.021 | .286±.006 | .244±.011 |
| | w/o Joint Training | .048±.011 | .131±.019 | .268±.012 | .181±.011 |
| Predictive Score (Lower the Better) | TimeGAN | **.093±.019** | **.038±.001** | **.273±.004** | **.303±.006** |
| | w/o Supervised Loss | .116±.010 | .054±.001 | .277±.005 | .380±.023 |
| | w/o Embedding Net. | .124±.002 | .048±.001 | .286±.002 | .410±.013 |
| | w/o Joint Training | .107±.008 | .045±.001 | .276±.004 | .348±.021 |

# Conclusions and Applications

- Novelties are two-fold:
  - Supervised loss to better capture temporal dynamics
  - Embedding network that provides a lower-dimensional adversarial learning space.

- Both novelties proven to be successful in generating more realistic and useful synthetic time-domain data
  - Supervised loss for high temporal correlations – CBCs and (some) glitches
  - Success of embedding network in all settings show the benefits of learning from a lower dimensional latent space

- Experiments show relevant benchmarks for testing
  - Visualizations
  - Multiple time series datasets
  - A/B testing for sources of gain

- Review of GANs in time-domain:
  - https://arxiv.org/pdf/2107.11098.pdf
  - Latest approaches, popular datasets and evaluation matrices

# Questions