Time-series Generative Adversarial Networks (NeurIPS 2019)

synthetic sequential data를 만들어내는 알고리즘이다. GAN으로 만드는 다른 data들과 가장 큰 차이는 temporal dynamics를 잘 잡아내야한다는 것이다. 그렇다면 TimeGAN이 어떤식으로 문제들을 해결했는지 살펴보자.

일단 TimeGAN이 다른 GAN과 다른 점을 알아보자.

- unsupervised adversarial loss on both real and synthetic data는 동일
- supervised loss 추가
  - to capture the stepwise conditional distributions in the data
- embedding network 추가
  - to provide a reversible mapping between features and latent representations
  - thereby reducing the high-dimensionality of the adversarial learning space
- can handle the mixed-data
  - static과 time-series data를 같이 생성할 수 있다.

그렇다면 이제 TimeGAN이 어떻게 이루어져있는지 살펴보자.

- TimeGAN은 크게 4개의 network로 이루어져있다.
  - embedding function
  - recovery function
  - sequence generator
  - sequence discriminator

Embedding and Recovery Functions

Embedding, Recovery function은 feature와 latent space를 매핑하는 역할을 한다. 이를 통해 adversarial network가 lower-dimension으로 underlying temporal dynamics를 학습할 수 있게 된다. (lower-dim adversarial learning space)

- $\mathcal{H}_{\mathcal{S}}$, $\mathcal{H}_{\mathcal{X}}$: latent vector spaces of feature space $\mathcal{S}$, $\mathcal{X}$
- e : embedding function $\mathcal{S} \times \prod_t \mathcal{X} \to \mathcal{H}_{\mathcal{S}} \times \prod_t \mathcal{H}_{\mathcal{X}}$
  - takes static & temporal features to latent codes $\boldsymbol{h}_S, \boldsymbol{h}_{1:T} = e(\boldsymbol{s}, \boldsymbol{x}_{1:T})$
  - $\boldsymbol{h}_S = e_S(\boldsymbol{s})$, $\boldsymbol{h}_t = e_{\mathcal{X}}(\boldsymbol{h}_S, \boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$
  - e는 recurrent network로 만든다
- r : recovery function $\mathcal{H}_{\mathcal{S}} \times \prod_t \mathcal{H}_{\mathcal{X}} \to \mathcal{S} \times \prod_t \mathcal{X}$
  - takes latent codes to feature representations $\tilde{\boldsymbol{s}}, \tilde{\boldsymbol{x}}_{1:T} = r(\boldsymbol{h}_S, \boldsymbol{h}_{1:T})$

- $\tilde{\boldsymbol{s}} = r_{\mathcal{S}}(\boldsymbol{h}_s),\ \tilde{\boldsymbol{x}_t} = r_{\mathcal{X}}(\boldsymbol{h}_t)$
- r은 feedforward network로 만든다

Embedding, Recovery function들은 꼭 위의 network가 아니여도 attention, temporal convolution 등을 통해 만들 수도 있다.

Sequence Generator and Discriminator

일반적인 GAN처럼 feature space에서 바로 데이터를 만드는 것이 아니라 generator는 embedding space를 만든다.

- $\mathcal{Z}_{\mathcal{S}}$. $\mathcal{Z}_{\mathcal{X}}$: vector spaces over which known distributions are defined (generator의 input으로 r.v를 뽑아낸다)
- g : generator function $\mathcal{Z}_{\mathcal{S}} \times \prod_t \mathcal{Z}_{\mathcal{X}} \to \mathcal{H}_{\mathcal{S}} \times \prod_t \mathcal{H}_{\mathcal{X}}$
  - takes a tuple of static and temporal random vectors to synthetic latent codes $\hat{\boldsymbol{h}}_S, \hat{\boldsymbol{h}}_{1:T} = g(\boldsymbol{z}_S, \boldsymbol{z}_{1:T})$
  - $\hat{\boldsymbol{h}}_S = g_S(\boldsymbol{z}_S),\ , \hat{\boldsymbol{h}}_t = g_{\mathcal{X}}(\hat{\boldsymbol{h}}_S, \hat{\boldsymbol{h}}_{t-1,}\boldsymbol{z}_t)$
  - g는 recurrent network

discriminator는 embedding space에서 나온 값은 input으로 받는 것이다.

- d : discrimination function $\mathcal{H}_{\mathcal{S}} \times \prod_t \mathcal{H}_{\mathcal{X}} \to [0,1] \times \prod_t [0,1]$
  - receives the static and temporal codes, returning classification $\tilde{y}_S, \tilde{y}_{1:T} = d(\boldsymbol{h}_S, \boldsymbol{h}_{1:T})$
  - $\tilde{y}_{\mathcal{S}} = d_{\mathcal{S}}(\tilde{\boldsymbol{h}}_{\mathcal{S}}),\ \tilde{y}_t = d_{\mathcal{X}}(\overleftarrow{\boldsymbol{u}}_t, \overrightarrow{\boldsymbol{u}}_t)$
  - where $\overrightarrow{\boldsymbol{u}}_t = \overrightarrow{c}_{\mathcal{X}}(\tilde{\boldsymbol{h}}_{\mathcal{S}}, \tilde{\boldsymbol{h}}_t, \overrightarrow{\boldsymbol{u}}_{t-1}),\ \overleftarrow{\boldsymbol{u}}_t = \overleftarrow{c}_{\mathcal{X}}(\tilde{\boldsymbol{h}}_{\mathcal{S}}, \tilde{\boldsymbol{h}}_t, \overleftarrow{\boldsymbol{u}}_{t+1}))$
  - d는 bidirectional recurrent network with a feeforward output layer

Jointly Learning to Encode, Generate, and Iterate

- reconstruction loss

$$L_R = E_{\boldsymbol{s},\boldsymbol{x}_{1:T}\sim P}\left[\|\boldsymbol{s} - \tilde{\boldsymbol{s}}\|_2 + \sum_t \|\boldsymbol{x}_t - \tilde{\boldsymbol{x}_t}\|_2\right]$$

- unsupervised loss

$$L_U = E_{\boldsymbol{s},\boldsymbol{x}_{1:T}\sim P}\left[\log y_{\mathcal{S}} + \sum_t \log y_t\right] + E_{\boldsymbol{s},\boldsymbol{x}_{1:T}\sim \hat{P}}\left[\log(1-\hat{y}_{\mathcal{S}}) + \sum_t \log(1-\hat{y}_t)\right]$$

위의 discriminator를 통해 unsupervised loss는 부족하다고 판단되었다. 따라서 추가적으로 generator를 효율적으로 학습시키기 위해 superviesd loss를 추가로 사용하였다. generator는 실제 data의 sequences of embeddings $\boldsymbol{h}_{1:t-1}$의 값을 받는 것이다. 이를 통해 temporal dynamics를 더 잘 잡을 수 있었다고 한다.

- supervised loss

$$L_S = E_{\boldsymbol{s},\boldsymbol{x}_{1:T}\sim P}\left[\sum_t \|\boldsymbol{h}_t - g_{\mathcal{X}}(\boldsymbol{h}_{\mathcal{S}}, \boldsymbol{h}_{t-1}, \boldsymbol{z}_t)\|_2\right]$$

Optimization

$$\min_{\theta_e.\theta_r}(\lambda L_S + L_R)$$

$$\min_{\theta_g}(\eta L_S + \max_{\theta_d} L_U)$$

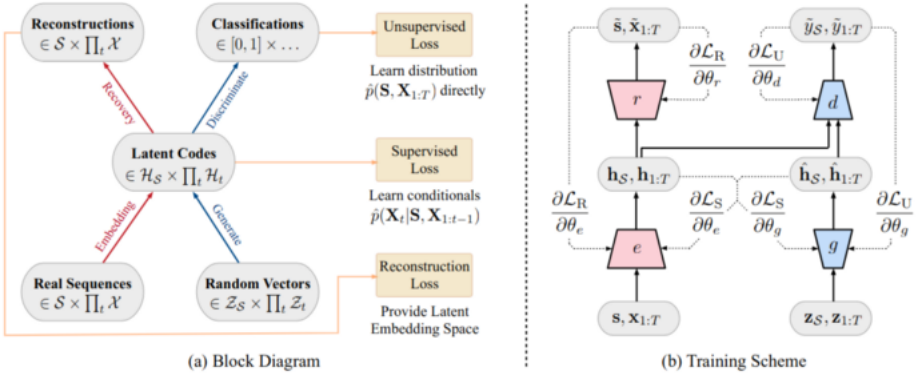

Figure 1: TimeGAN