

## [단생 배경]

# Reviewer: 김정학

기존 존재했던 Gan들은 새롭게 만들어진 데이터가 원 데이터와 비슷한지만 확인했다. 이를 unsupervised 방법이라고하는데 사실은 표현화면  $P(X_{1:T})$ 에 집중했다. 그런데 time-series data의 경우에는 autoregressive 특성까지 포함해서 데이터를 만들어야 한다. (autoregressive 특성이라면, 과거의 정보가 다음 정보에 영향을 미치는 특성이다) 이를 사실로 표현하면  $P(X_t | X_{1:t-1})$ 로 나타낼 수 있고 이문제에서는 이 부분까지 학습을 해야한다고 주장한다.

## [특장]

① 기존에 unsupervised learning or supervised learning을 포함한다.

unsupervised: 만들어진 데이터가 진짜인지 거짓인지 판별.

supervised : 원데이터를 기반으로 loss를 계산해서 stepwise conditional distribution의 특징을 학습하게 함.

→ train data로 더 많은 부분을 학습할 수 있음.

② embedding network를 도입.

이를 통해서 Gan의 학습차선을 줄일 수 있고 temporal-dynamic 정보를 더 잘 뽑아낼 수 있음.

embedding network는 generator network를 동시에 학습하면서 supervised loss를 줄일 수 있다.

③ 시계열 데이터에 고정(static) 데이터를 포함해서 학습할 수 있는 general 모델 (코드는 고려하지 않음.. 오망?)

## [모델 학습 (코드 포함)]

이어서 연습해듯이 모델은 두 가지를 학습하는 것을 목표로 한다.

$$\min_{\hat{p}} D(p(\mathbf{S}, \mathbf{X}_{1:T}) \| \hat{p}(\mathbf{S}, \mathbf{X}_{1:T}))$$

$\mathbf{S}$ : static data

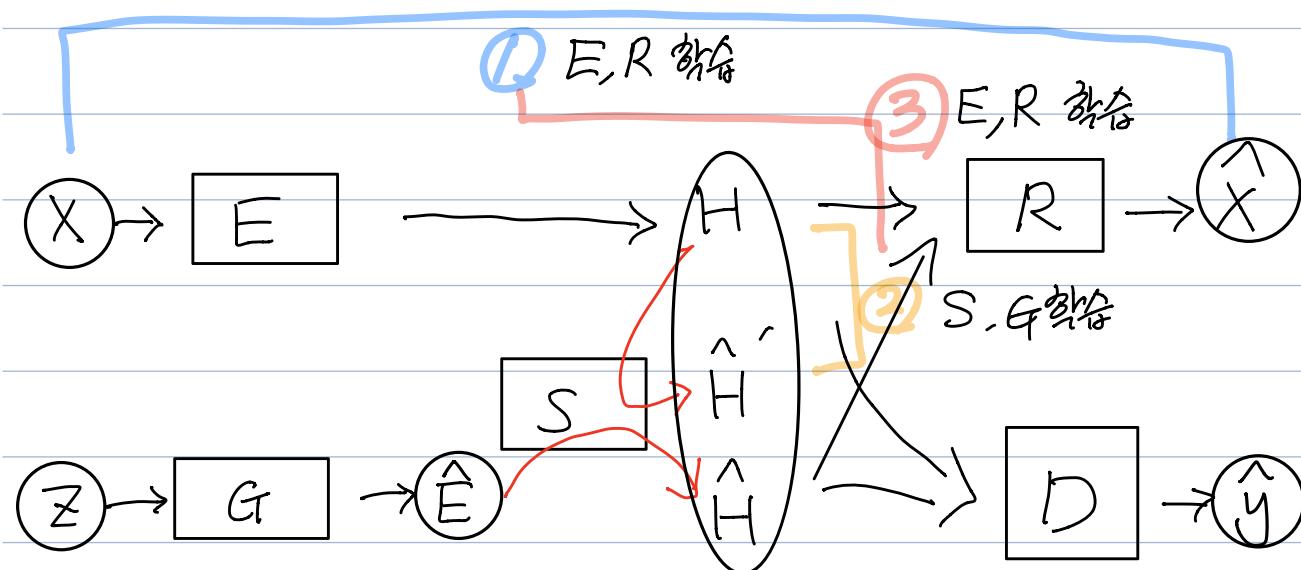
$T$ : temporal data

$$\min_{\hat{p}} D(p(\mathbf{X}_t | \mathbf{S}, \mathbf{X}_{1:t-1}) \| \hat{p}(\mathbf{X}_t | \mathbf{S}, \mathbf{X}_{1:t-1}))$$

① generator로 부터 나온 결과가 실제 데이터와 유사한지

② autoregressive 한 특성을 학습하기 위한지.

이 두 가지를 기억하고 모델을 살펴보자.



○ : tensor variable

□ : network

③  
S, G 학습  
D 학습.

<네트워크 및 forward 과정>

embedding  $\boxed{E}$  : 실제 데이터  $X$ 를 인코딩해 빼어 (latent vector)  $H$ 를 리턴한다.

$\rightarrow X$ 를 인코딩하는 역할.

recovery R : 히든벡터 ( $H, \hat{H}$ )을 인풋으로 받아서 담개레이어로  
복원

→ 인코딩된 벡터를 복원하는 역할.

generator G : seed 값  $Z$ 를 인풋으로 받아서 embedding network가  
학습한  $H$ 와 비슷하게  $\hat{H}$ 를 학습한다.

→ 새로운 히든벡터를 만드는 역할.

supervisor S : 히든벡터 ( $H, \hat{H}, \hat{H}'$ )를 인풋으로 받아서 또 다른 히든벡터  
( $\hat{A}, \hat{A}'$ )를 학습한다.

→ 히든벡터의 시계적 다음 시간의 히든벡터를 뽑아내는  
역할.

discriminator D : 히든벡터 ( $H, A, A'$ )를 바탕으로 진짜면 1, 가짜면 0  
학습한다.

→ 히든벡터가 진짜인지 가짜인지 판별하는 역할.

### <backward 및 학습>

세 가지 단계로 모델을 학습 (순차적으로)

① embedding, recovery network 학습.

$X$ 와  $H$ 를 복원한 값인  $\hat{X}$ 의 차이를 loss 값 ( $L_R$ )으로 해서  
 $E, R$ 을 학습.

② supervisor, generator를 학습 (supervised)

$H$ 와  $H$ 의 다음벡터인  $A'$ 의 차이를 loss 값 ( $L_S$ )으로 해서  
 $S, G$ 를 학습 (그런데  $S$ 밖에 학습시킬 수 없지 않나?  $G$ 를 어떻게  
학습하는 거지? 코드에서는  $G$ 도 학습함)

③ 모든 네트워크를 학습 (unsupervised)

- ①+②를 loss 값으로 해서  $E, R$ 을 학습.

- 학습기반 학습한 값  $\hat{y}$ 을 이용해서 ( $L_U$ ) S와 D를 학습.
- D의 성능이 너무 나을 경우에만 D 학습.

$$\mathcal{L}_R = \mathbb{E}_{\mathbf{s}, \mathbf{x}_{1:T} \sim p} [\|\mathbf{s} - \tilde{\mathbf{s}}\|_2 + \sum_t \|\mathbf{x}_t - \tilde{\mathbf{x}}_t\|_2]$$

$$\mathcal{L}_U = \mathbb{E}_{\mathbf{s}, \mathbf{x}_{1:T} \sim p} [\log y_S + \sum_t \log y_t] + \mathbb{E}_{\mathbf{s}, \mathbf{x}_{1:T} \sim \hat{p}} [\log(1 - \hat{y}_S) + \sum_t \log(1 - \hat{y}_t)]$$

$$\mathcal{L}_S = \mathbb{E}_{\mathbf{s}, \mathbf{x}_{1:T} \sim p} [\sum_t \|\mathbf{h}_t - g_{\mathcal{X}}(\mathbf{h}_S, \mathbf{h}_{t-1}, \mathbf{z}_t)\|_2]$$