Paper study (5주차)

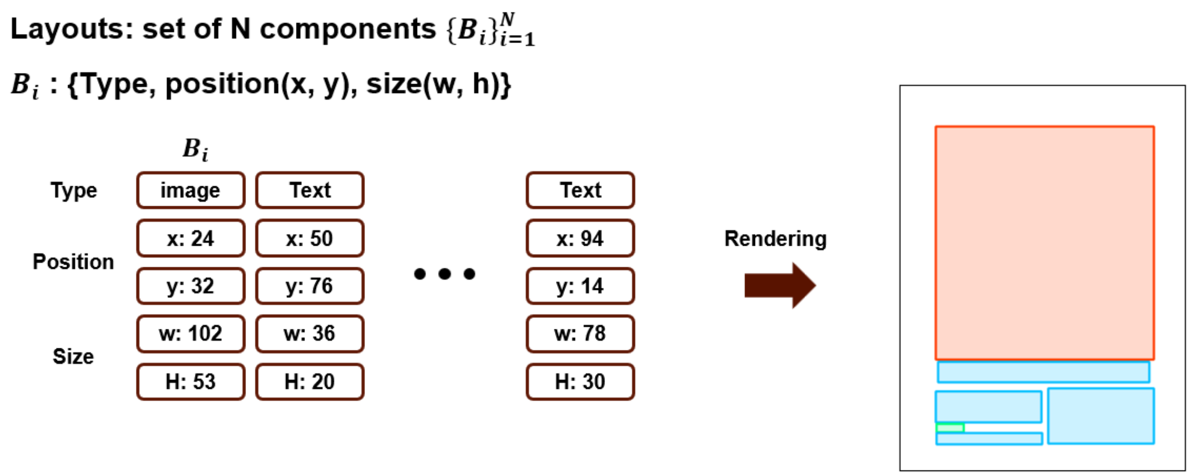
발표자 김도현

논문 제목 : **DLT: Conditioned layout generation with Joint Discrete-Continuous Diffusion Layout Transformer**

[논문 링크]

DLT: Conditioned layout generation with Joint Discrete-Continuous...
Generating visual layouts is an essential ingredient of graphic design. The ability to condition layout generation on a partial subset of component attributes is critical to real-world...
https://arxiv.org/abs/2303.03755

Goal: generate layouts conditioning on constraints **c**(True or Unknown for each attributes)



Method

Autoregressive Transformer based model ⇒ hard to consider Global context

GAN, VAE ⇒ not achieve significantly better performance

Diffusion ⇒ 각광 받고 있으며 여러 time step 동안 generate하기 때문에 global context 고려 가능

But layout data의 특성상 (discrete(type) + continuous(position, size)) diffusion 바로 적용 불가능

⇒ discrete diffusion D3PM continuous diffusion DDPM joint하게 적용

diffusion에 대한 고찰

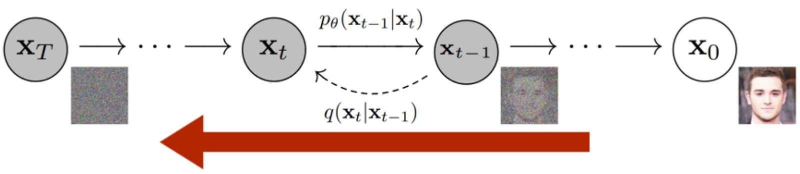
generative models: latent space z에서부터 training dataset과 유사한 X를 만들어 내는 것이 목표

denerative model 중 diffusion은 diffusion 과정에서 영감을 받았으며 천천히 data distribution을 부수고 이의 역 과정을 스텝별로 학습할 수 있지 않을까 해서 시작됨

또한 short time에서 각 데이터(termodynamics에서 분자)의 확률분포는 normal distribution을 따름

천천히 부수고 역 과정을 학습하기 위해 부수는 과정인 forward process를 q로 정의하고 역과정 p를 neural network를 이용해서 학습하는 것을 목표로 함

markov chain에 의해 q는 다음과 같이 정의 가능 이때 beta와 alpha 등은 fix하며 normal distribution에서의 sampling을 VAE와 동일하게 reparameterization trick을 이용하여 정의할 수 있



$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \qquad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

gradually adds Gaussian noise
→ pure Gaussian noise at timestep T

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

β_t: variance schedule
α_t: $1-\beta_t$
$\overline{\alpha_t} = \prod_{s=1}^{t} \alpha_s$

$$\mathbf{x}_t(\mathbf{x}_0, \boldsymbol{\epsilon}) = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Reparameterization trick

reverse process의 경우 우리의 target이 되며 loss function을 정의해야 함

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

$$\mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

다른 generative model과 동일하게 $logp_\theta(X_0)$ 를 maximize 하는게 목표이며 VAE와 비슷하게 upper bound를 정의 후 식을 예쁘게 전개하면 위와 같이 전개 가능

이때 L_{t-1} term이 최종 loss로 귀결되며 L_T와 L_0는 너무 작아 학습 과정에서 제외되며 추가적으로 VAE의 regularization term과 reconstruction term과 동일한 것을 확인할 수 있는데 diffusion에서는 1000 스텝으로 나눴기 때문에 reconstruction term을 생략할 수 있었고 또한 forward process를 정의해 애초에 latent space, forward process의 최종 X_T를 pure gaussian noise로 만들 수 있기 때문에 regularization term을 생각해도 됨

최종 loss term에서 forward process와 reverse process가 normal distribution에서 결정된다는 성질을 이용하고 variance를 fix하고 KL divergence를 계산하여 loss를 다음과 같이 전개 가능

$$L_{t-1} = \mathbb{E}_q\left[\underbrace{\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2}_{\text{KL divergence }(\mu)}\right] + C \quad \text{Other KL divergence term}$$

또한 denoising이라는 concep에 맞게 denoising model을 이용하는 형태로 수식을 재전개 할 수 있으며 X_0의 평균을 예측하는 것도 가능하지만 좋은 성능을 보이지 못했다고 주장

이후 loss term 앞의 상수 term을 제거하여 larger t에서 학습을 원활히 할 수 있도록 함 따라서 최종 loss term은 다음과 같음

$$L_{\mathrm{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

DDPM의 contribution은 loss term을 simplify해 high quality의 image를 generation한 것이라고 할 수 있음

이후 diffusion은 DDIM, classifier guidance, classifier free guidance, latent diffusion model 등 수많은 연구들이 진행되며 발전하게 되었고 DALL-E2나 stable diffusion, imagen 등 large model도 많이 탄생하게 됨

image뿐 아니라 audio, 3d image generation 등 여러 task에서 high performance를 보임

Why is diffusion model powerful?

1. 천천히 noising denoising하는 process를 통해 stable한 training process를 가질 수 있게 됨

⇒ overfitting에 대해 Robustness하게 되었고 flexible한 architecture를 지니게 되어 conditional generation 등으로 유연하게 확장할 수 있었고 scalability가 보장되어 large model들도 많이 등장할 수 있었으며 여러 task에서 이런 모델을 사용하여 확장될 수 있었음

2. interpretable latent space

⇒ 1000 step으로 나누면서 그 사이의 중간 representation을 모두 latent space로 볼 수 있고 이를 통해 안정적으로 X_0와 유사하도록 천천히 밀어줄 수 있게 됨 또한 step 사이에 randomness가 부여되어, diversity를 쉽게 확보할 수 있었음

3. 단단한 이론적 배경

⇒ score based model 등 이론을 통합하려는 시도가 성공적이었으며 따라서 안정적으로 모델 구조 등을 확장할 수 있게 됨

D3PM

discrete에서도 diffusion 해보자! but continuous 처럼 sampling을 통해 noise를 부여할 수 없음

⇒ transition matrix Q를 정의하여 noise 부여

$$[\boldsymbol{Q}_t]_{ij} = q(x_t = j | x_{t-1} = i) \qquad \mathbf{Q}_t^{\text{type}} = \begin{array}{c} \begin{array}{cccc} \textbf{Text} & \textbf{image} & \quad & \textbf{mask} \end{array} \\ \begin{bmatrix} 1-\gamma_t & 0 & \cdots & 0 \\ 0 & 1-\gamma_t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix} \end{array} \begin{array}{l} \\ \textbf{Text} \\ \textbf{image} \\ \\ \textbf{mask} \end{array}$$

$$q(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}) = \text{Cat}(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_{t-1}\boldsymbol{Q}_t)$$ → **categorical distribution**

$$q(\boldsymbol{x}_t | \boldsymbol{x}_0) = \text{Cat}\left(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_0 \overline{\boldsymbol{Q}}_t\right), \quad \text{with} \quad \overline{\boldsymbol{Q}}_t = \boldsymbol{Q}_1 \boldsymbol{Q}_2 \ldots \boldsymbol{Q}_t$$

→ **categorical distribution is converge at t =T (e.g. uniform distribution, all masked)**

Q를 통해 모든 time step을 거치게 되면 특정 distribution으로 수렴하게 되어 pure gaussian noise가 되는 것과 동일한 효과를 내게 됨

Focus on using a neural network to predict the logits of distribution $\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) \propto \sum_{\widetilde{\boldsymbol{x}}_0} q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_t|\widetilde{\boldsymbol{x}}_0)\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$$

Loss function:

$$L_{\mathrm{vb}} = \mathbb{E}_{q(\boldsymbol{x}_0)}\Bigg[\underbrace{D_{\mathrm{KL}}[\cancel{q(\boldsymbol{x}_T|\boldsymbol{x}_0)||p(\boldsymbol{x}_T)}]}_{L_T} + \sum_{t=2}^{T}\underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\big[D_{\mathrm{KL}}[q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)]\big]}_{L_{t-1}} \underbrace{-\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)]}_{L_0}\Bigg].$$

$$L_\lambda = L_{\mathrm{vb}} + \boxed{\lambda\, \mathbb{E}_{q(\boldsymbol{x}_0)}\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}[-\log \widetilde{p}_\theta(\boldsymbol{x}_0|\boldsymbol{x}_t)]}$$

auxiliary loss term: λ = 0.001 was best
→ Cross Entropy

loss의 경우 diffusion의 loss를 그대로 차용하지만 x_0를 예측하는 모델로 정의하여 조금 더 x_0를 예측하는데 focus할 수 있도록 함

또한 auxiliary loss term을 부여할 수 있게 되었음

Joint Discrete-Continuous Diffusion

Position, Size → **Continuous diffusion models: DDPM**

Type → **Discrete diffusion models: D3PM**

$$q^c(\bar{x}_t|\bar{x}_{t-1}) = \mathcal{N}(\bar{x}_t, \sqrt{1-\beta_t}\bar{x}_{t-1}, \beta_t \cdot I)$$

$$\mathbf{Q}_t^{\mathrm{type}} = \begin{bmatrix} 1-\gamma_t & 0 & \cdots & 0 \\ 0 & 1-\gamma_t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix}$$

keeping or masking (absorbing-state)

Predict X_0

Continuous loss: $\mathcal{L}_{box} = \mathbb{E}_{\bar{x}_0,\bar{y}_0\sim q(\bar{x}_0,\bar{y}_0|c),t\sim[0,1]}||F_\theta^c(\bar{x}_t, c, \bar{y}_t) - \bar{x}_0||^2$

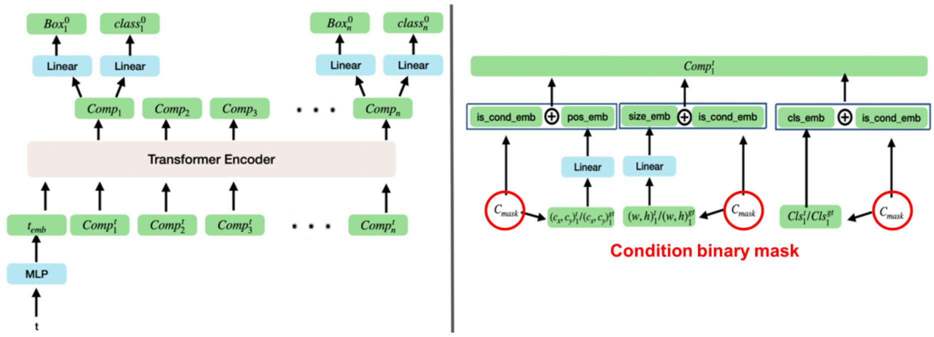
Discrete loss: $L_\lambda = L_{\mathrm{vb}} + \lambda\, \mathbb{E}_{q(\boldsymbol{x}_0)}\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}[-\log \widetilde{p}_\theta(\boldsymbol{x}_0|\boldsymbol{x}_t)]$ **D3PM**

$$\mathcal{L}_{cls} = \mathbb{E}_{\bar{y}_0,\bar{x}_0\sim q(\bar{y}_0,\bar{x}_0|c),t\sim[0,1]}CE(F_\theta^d(\bar{x}_t, c, \bar{y}_t), \bar{y}_0)$$ **Reweighted absorbing-state D3PM objective**

$$\mathcal{L}_{model} = \lambda_1 \cdot \mathcal{L}_{box} + \lambda_2 \cdot \mathcal{L}_{cls}$$

DLT의 loss는 따라서 위와 같이 정의되며 cls (class)의 loss term은 D3PM에서 absorbing-state의 Q를 사용하면 cross entropy만 사용하는 것과 유사하다는 것을 밝혀 이를 사용

Model Architecture



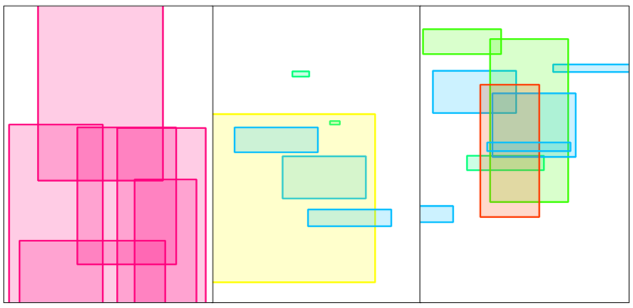
Input: {(type, C), (position, C), (size, C)}…
output: {type, position, size}…

※ C: condition (true or unknown)

model의 architecture는 위와 같으며 input으로 {(type, C), (position, C), (size, C)}… sequence를 넣어주게 되면 각 요소의 type, position, size를 예측

Experiments

Dataset	Publaynet											
	Conditioned on Category				Category + Size				Uncoditioned			
Model	pIOU	Overlap	Alignment	FID	pIOU	Overlap	Alignment	FID	pIOU	Overlap	Alignment	FID
LT [7]	2.7	7.6	0.41	26.8	7.1	11.7	0.14	22.0	0.62	2.4	0.11	19.3
BLT [16]	0.89	4.4	**0.10**	36.6	1.7	8.1	**0.09**	14.2	0.60	2.7	0.12	69.8
VTN [1]	2.1	6.8	0.29	22.1	5.3	15.3	**0.09**	17.9	0.68	**2.6**	**0.08**	14.5
DLT	**0.67**	**3.8**	0.11	**10.3**	**0.82**	**4.2**	**0.09**	**11.4**	**0.59**	**2.6**	0.11	**13.8**
Dataset	Rico											
	Conditioned on Category				Category + Size				Uncoditioned			
Model	pIOU	Overlap	Alignment	FID	pIOU	Overlap	Alignment	FID	pIOU	Overlap	Alignment	FID
LT [7]	25.6	75.2	0.58	14.7	23.8	**69.1**	0.41	8.4	23.2	65.1	0.40	15.2
BLT [16]	30.2	85.1	**0.12**	27.8	24.5	79.3	0.30	10.2	23.0	70.6	0.25	18.7
VTN [1]	25.4	74.2	0.43	14.3	24.1	69.6	0.44	7.1	29.4	72.1	0.26	29.4
DLT	**21.9**	**70.6**	0.18	**9.5**	**17.2**	70.2	**0.28**	**6.3**	**19.3**	**58.4**	**0.21**	**13.9**
Dataset	Magazine											
	Conditioned on Category				Category + Size				Uncoditioned			
Model	pIOU	Overlap	Alignment	FID	pIOU	Overlap	Alignment	FID	pIOU	Overlap	Alignment	FID
LT [7]	19.9	71.0	1.5	44.7	21.4	70.2	**1.2**	45.3	21.4	70.0	1.1	42.6
BLT [16]	36.4	133	1.4	49	20.5	56.8	**1.2**	27.3	30.1	134	1.1	52.7
VTN [1]	10.3	38.7	2.4	37.6	9.9	28.8	2.3	29.4	20.1	70.7	**0.9**	62.7
DLT	**5.9**	**16.1**	**1.3**	**26.2**	**6.8**	**19.4**	1.6	**21.7**	**4.8**	**12.1**	1.8	**40.9**



성능이 잘 나왔음을 주장

Contribution

- **apply Joint Discrete-Continuous Diffusion to layout generation**

Limitation

- **Utility↓: model does not generate layout by looking at each contents, the suitable contents must be manually inserted by a person**

⇒ 확장연구로 contents aware layout generation 연구 중

▼ 백성은

- autoregressive → global 반영 x
- layout data는 discrete / continuous 모두 고려해야 함
  - 따라서 joint discrete-continuous diffusion
  - continuous : DDPM / discrete : D3PM

$$L_{t-1} = \mathbb{E}_q\left[\underbrace{\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2}_{\text{KL divergence }(\mu)}\right] + C$$

Other KL divergence term