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

ICCV 2023

Dohyun Kim
a12s12@korea.ac.kr
Multimodal Interactive Intelligence Laboratory (MIIL)

Goal

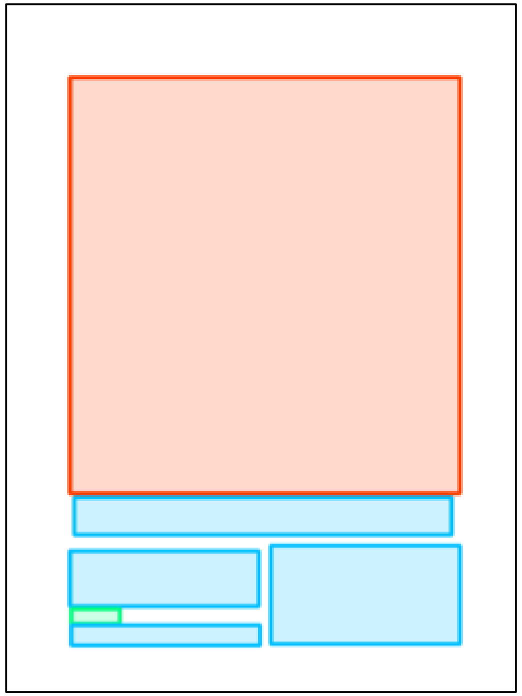
Generate layouts conditioning on constraints c — True or Unknown for each attributes (type, position, size)

Layouts: set of N components $\{B_i\}_{i=1}^{N}$

B_i : {Type, position(x, y), size(w, h)}

	B_i			
Type	image	Text		Text
Position	x: 24	x: 50	• • •	x: 94
	y: 32	y: 76		y: 14
Size	w: 102	w: 36		w: 78
	H: 53	H: 20		H: 30

Rendering



Method

Autoregressive Transformer based models

- one element only depend on the generated part of the layout

→ hard to consider Global context

Non-autoregressive models(GAN, VAE) to consider global context

- not achieve significantly better performance with single pass

Diffusion model can consider global context and achieve better performance

- takes the layout in the last step as global context

Method

Based on Diffusion model

Layout data : discrete(type) and continuous (position, size)

Joint Discrete-Continuous Diffusion

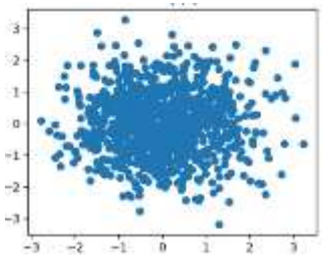
Position, Size → **Continuous diffusion models: DDPM**

Type → **Discrete diffusion models: D3PM**

Generative Models

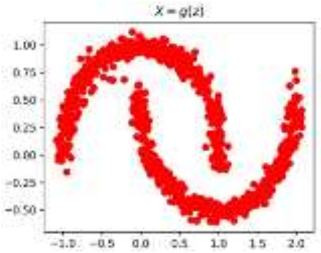
Generative model: Models that Generate similar data following the distribution of the training data by learning the training data

$p_z(z) \sim N$

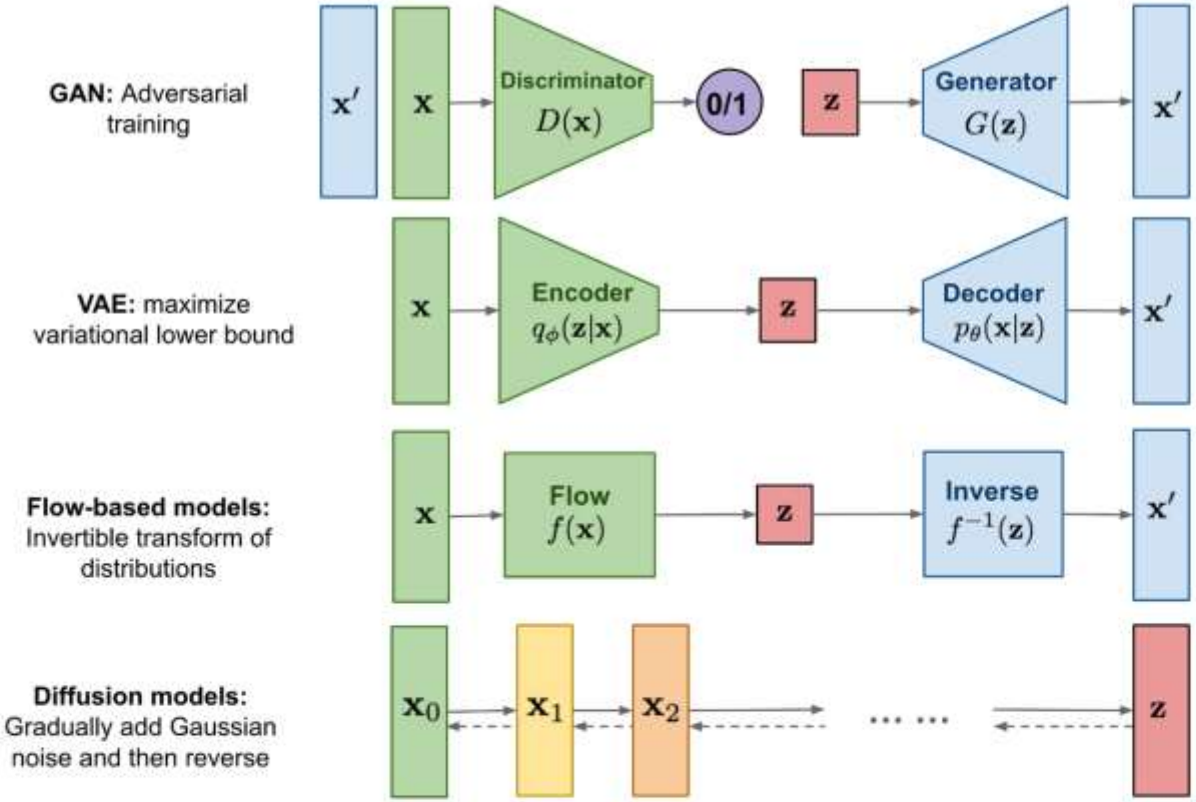


Generative Model

$p'_x(x)$



Latent space

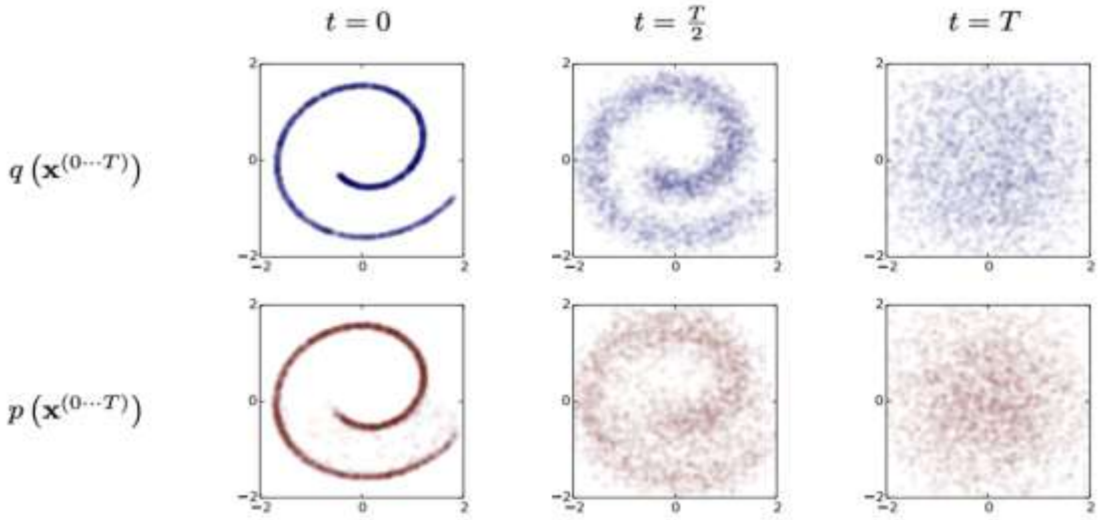


Multimodal Interactive Intelligence Laboratory

Diffusion Models

Suggested by “Deep Unsupervised Learning using Nonequilibrium Thermodynamics”(2015)

inspired by considerations from nonequilibrium thermodynamics
→ If the model learns the whole system which indicates moving to a uniform state,
could it also learn the process of reverting back to the original distribution?



In a very short time, The next position of the molecules is determined within the Gaussian Distribution

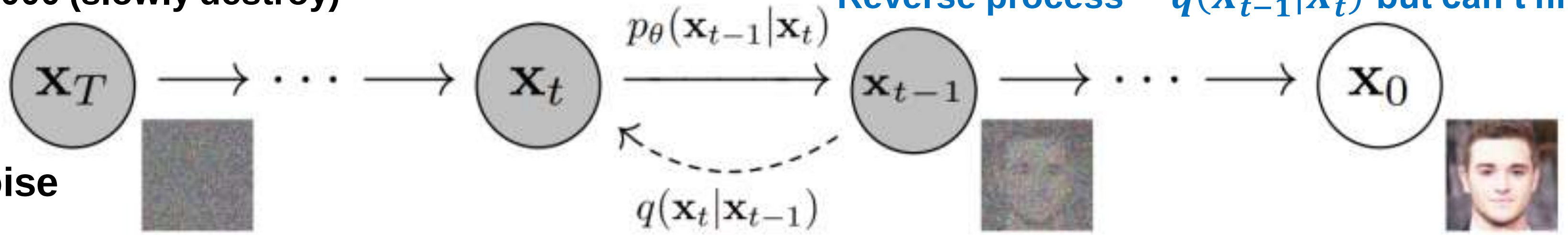
Diffusion Models

slowly destroy structure in a data distribution & learn a reverse diffusion process

→ Image sampling: sample X_T from pure gaussian distribution + reverse process

T = 1000 (slowly destroy)

Reverse process ~ $q(X_{t-1}|X_t)$ but can't find directly



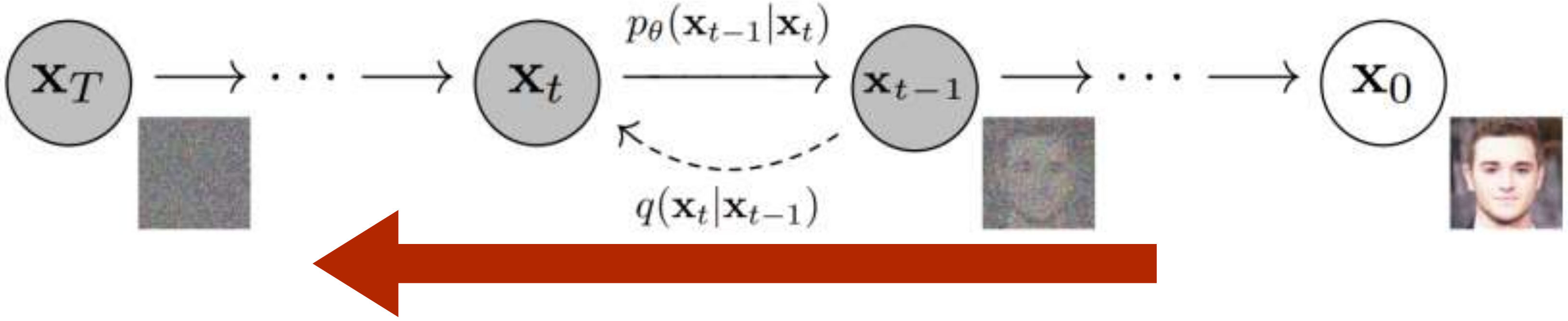
Pure noise

Forward(Diffusion) process

Figure 2: The directed graphical model considered in this work.

DDPM "Denoising Diffusion Probabilistic Models" (NeurIPS, 2020):
simplify loss term → high quality generated images

Forward(Diffusion) process



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

gradually adds Gaussian noise
→ pure Gaussian noise at timestep T

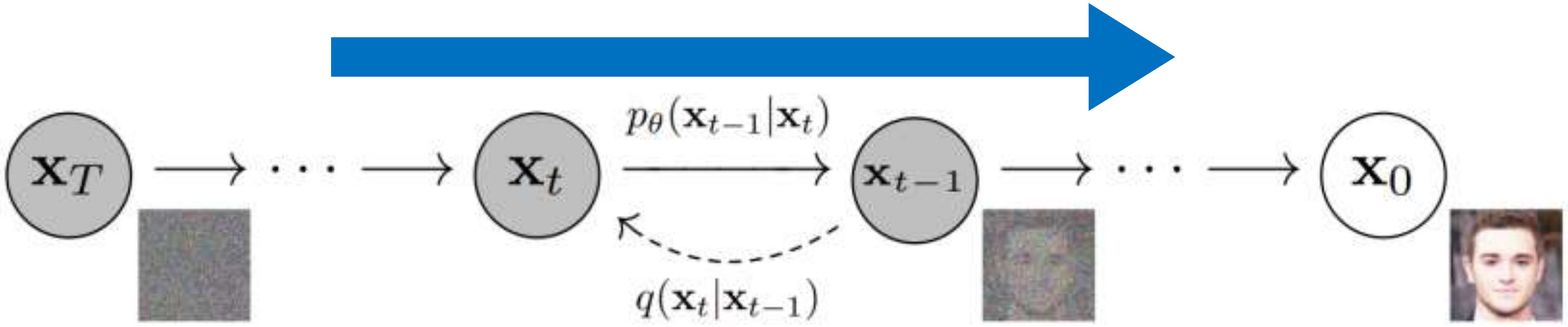
$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

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 process



Maximize $log p_\theta(X_0)$ but intractable ➡ Find Variational upper bound on negative log likelihood

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

Using bayes rule, Markov chain and some tricks

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

L_T: Regularization term
L_0: Reconstruction term
→ too small & ignored

In VAE ➡

$$\boxed{\mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}^{(i)}) = -D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z})\right]}$$

Reverse process

$$\sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \,\|\, p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}$$

➡ L_{t-1}: Final loss term

$$\boxed{\tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t}$$

Untrained (fixed as constant → simplified)

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \underline{\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0)}, \tilde{\beta}_t \mathbf{I}), \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

$$= \mathcal{N}(\mathbf{x}_{t-1}; \underline{\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)}, \sigma_t^2 \mathbf{I})$$

Forward process posterior mean

Model output

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

Other KL divergence term

$$\boxed{D_{KL}(N_1\|N_2) = \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{\sigma_1^2 + (\mu_1-\mu_2)^2}{2\sigma_2^2} - \frac{1}{2}}$$

KL divergence (μ)

Reverse process

Simplify loss term → predict noise

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

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t$$

denoising model

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \underline{\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t))}\right) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$$

$$\boxed{\mathbf{x}_t(\mathbf{x}_0, \boldsymbol{\epsilon}) = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}$$

Reformulating the loss function to predict residuals! (It is possible to predict X_0 but worse quality)

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

Model can focus on more difficult denoising tasks at lager t terms

DDPM

Overall algorithm

Algorithm 1 Training

repeat
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
  $t \sim \mathrm{Uniform}(\{1, \dots, T\})$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Take gradient descent step on
  $$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\underbrace{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}}_{X_t}, t) \right\|^2$$
until converged

Algorithm 2 Sampling

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \dots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
end for
return $\mathbf{x}_0$

Main Contribution: simplify the loss

- Deleting some loss terms with fixed $\beta_{1:T}$
- Residual Estimation
- Not to learn variance (fix) → make it easy for training

Result of DDPM

Figure 14: Unconditional CIFAR10 progressive generation

Figure 11: CelebA-HQ 256 × 256 generated samples

Evolution of diffusion models

“Deep Unsupervised Learning using Nonequilibrium Thermodynamics.”
“Denoising Diffusion Probabilistic Models.” (DDPM)

“Improved Denoising Diffusion Probabilistic Models.”(DDIM) → accelerate sampling
“Diffusion Models Beat GANs on Image Synthesis.” → high quality on conditional image generation using classifier guidance
“High-Resolution Image Synthesis with Latent Diffusion models”(LDM) → diffusion in latent space (VQ-VAE)
“Classifier-Free Diffusion Guidance” → improve conditional image generation
“Prompt-to-Prompt Image Editing with Cross-Attention Control → image editing

⋮

DALL-E 2
Imagen
SDXL
ControlNet
Sora

Why is diffusion powerful?

Gradually noising and denoising process

→ **Stable training (compared with GAN)**

- Robustness to Overfitting
- Flexible generators for various types of conditioning
- Scalability → large models → fine tuning or utilize for derived tasks

→ **Interpretable Latent Space (1000 steps)**

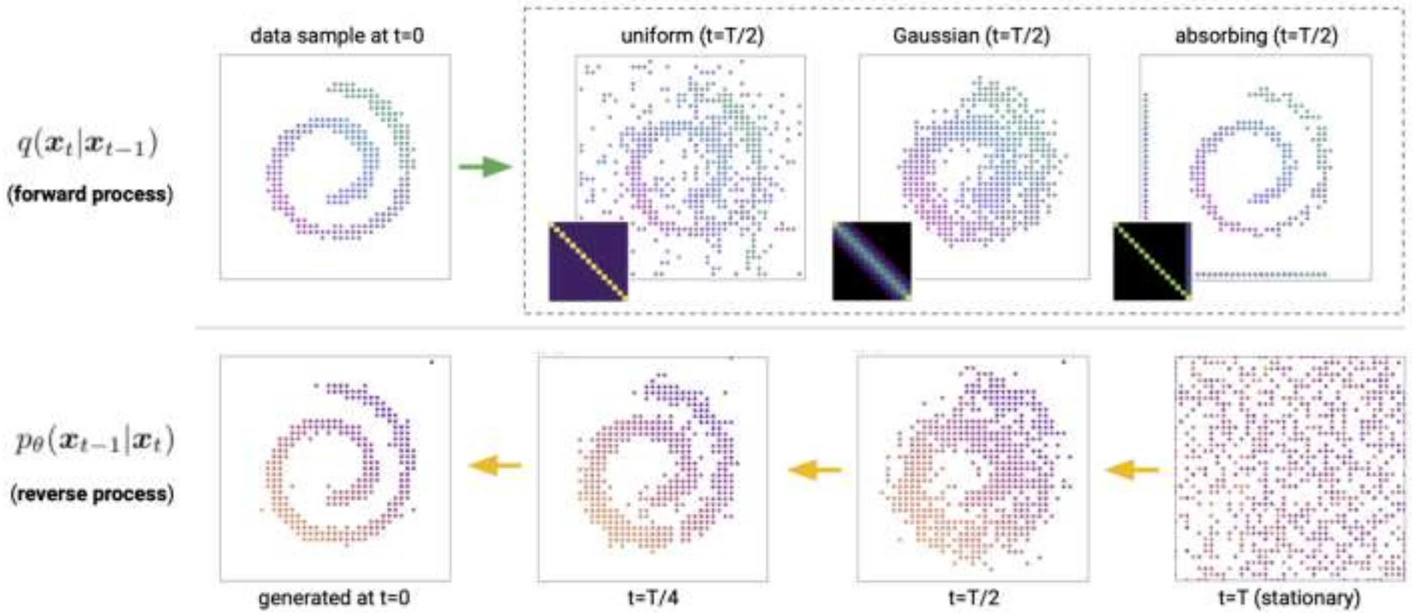
- generate diverse samples (randomness between steps)

D3PM

"Structured Denoising Diffusion Models in Discrete State-Spaces"(NeurIPS 2021)

Discrete Denoising Diffusion Probabilistic Models(D3PM):
Approach to modeling the diffusion process in discrete state space

- Using transition matrix Q_t



Forward process of D3PM

Can not directly using the forward process in a continuous space (sampling from gaussian distribution)

→ Forward process using transition matrix Q_t

$$[\boldsymbol{Q}_t]_{ij} = q(x_t = j | x_{t-1} = i)$$

$$\mathbf{Q}_t^{\text{type}} = \begin{array}{c} \begin{array}{cccc} \text{Text} & \text{image} & & \text{mask} \end{array} \\ \begin{bmatrix} 1-\gamma_t & 0 & \cdots & 0 \\ 0 & 1-\gamma_t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix} \end{array} \begin{array}{l} \text{Text} \\ \text{image} \\ \\ \text{mask} \end{array}$$

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

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

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

loss of D3PM

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:

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

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

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

DLT

Joint Discrete-Continuous Diffusion

Position, Size → Continuous diffusion models: DDPM

Type → Discrete diffusion models: D3PM

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

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

keeping or masking (absorbing-state)

Predict X_0

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

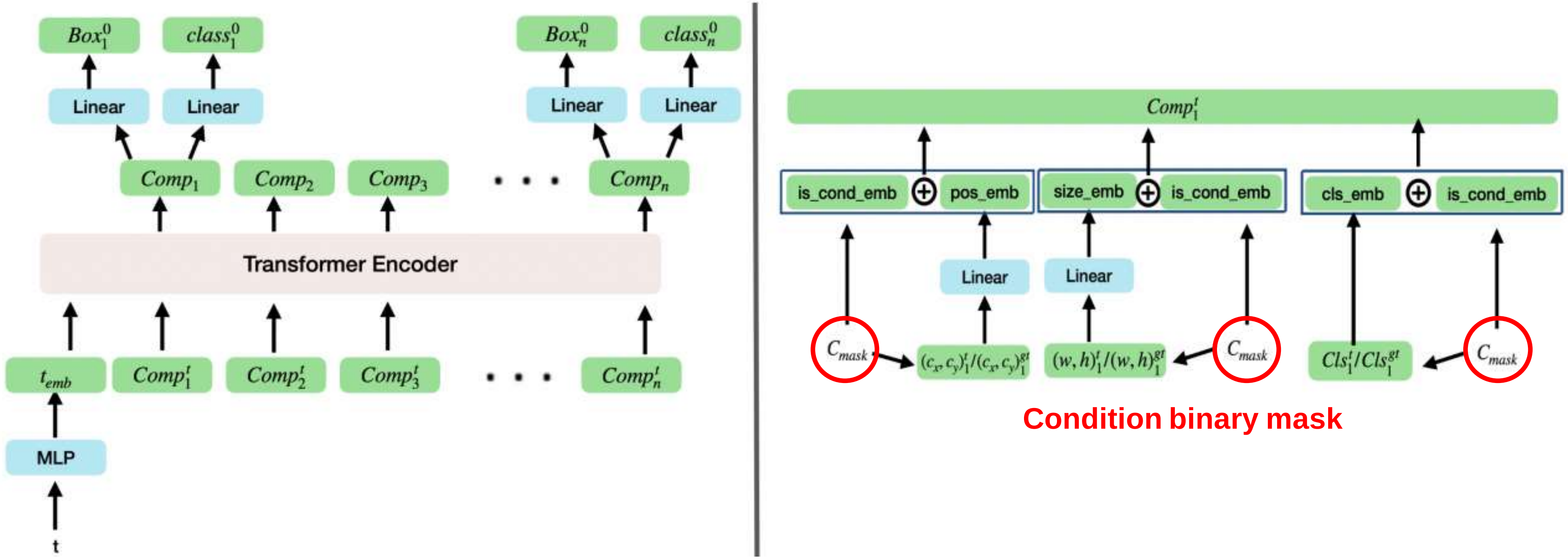
Discrete loss: $L_\lambda = L_{\text{vb}} + \lambda \, \mathbb{E}_{q(\boldsymbol{x}_0)} \mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} [-\log \widetilde{p}_\theta(\boldsymbol{x}_0|\boldsymbol{x}_t)]$ D3PM

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

Reweighted absorbing-state D3PM objective

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

Model Architecture

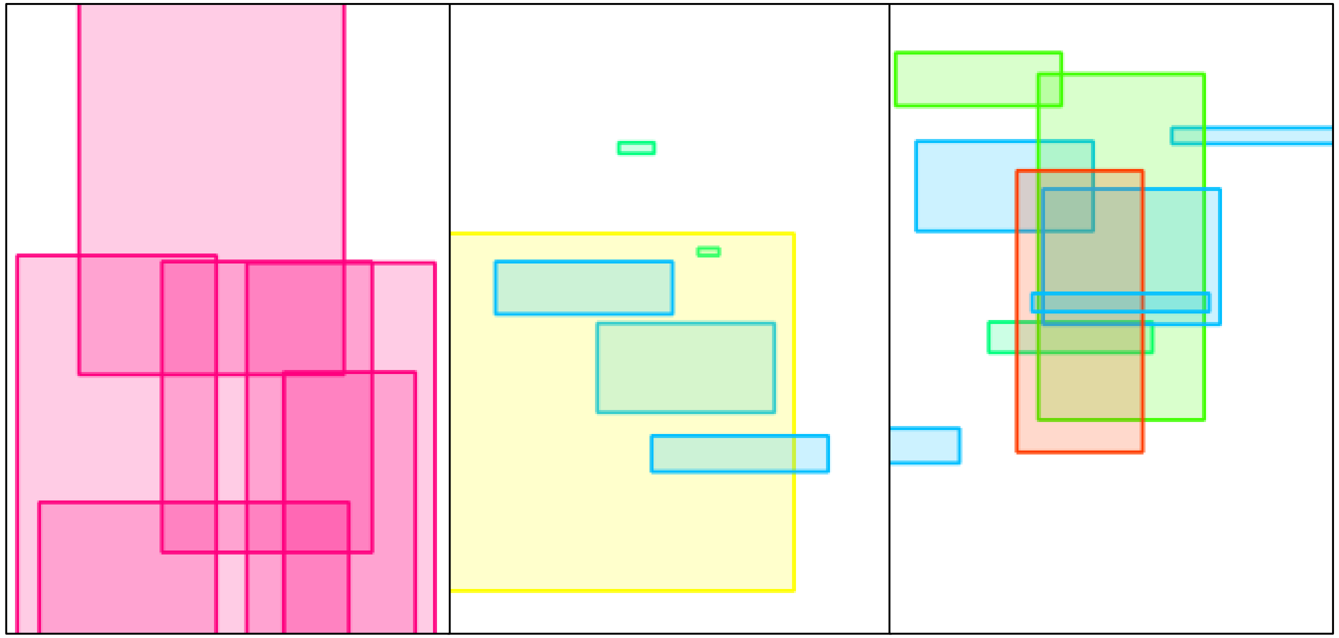


Input: {(type, C), (position, C), (size, C)}…
output: {type, position, size}…

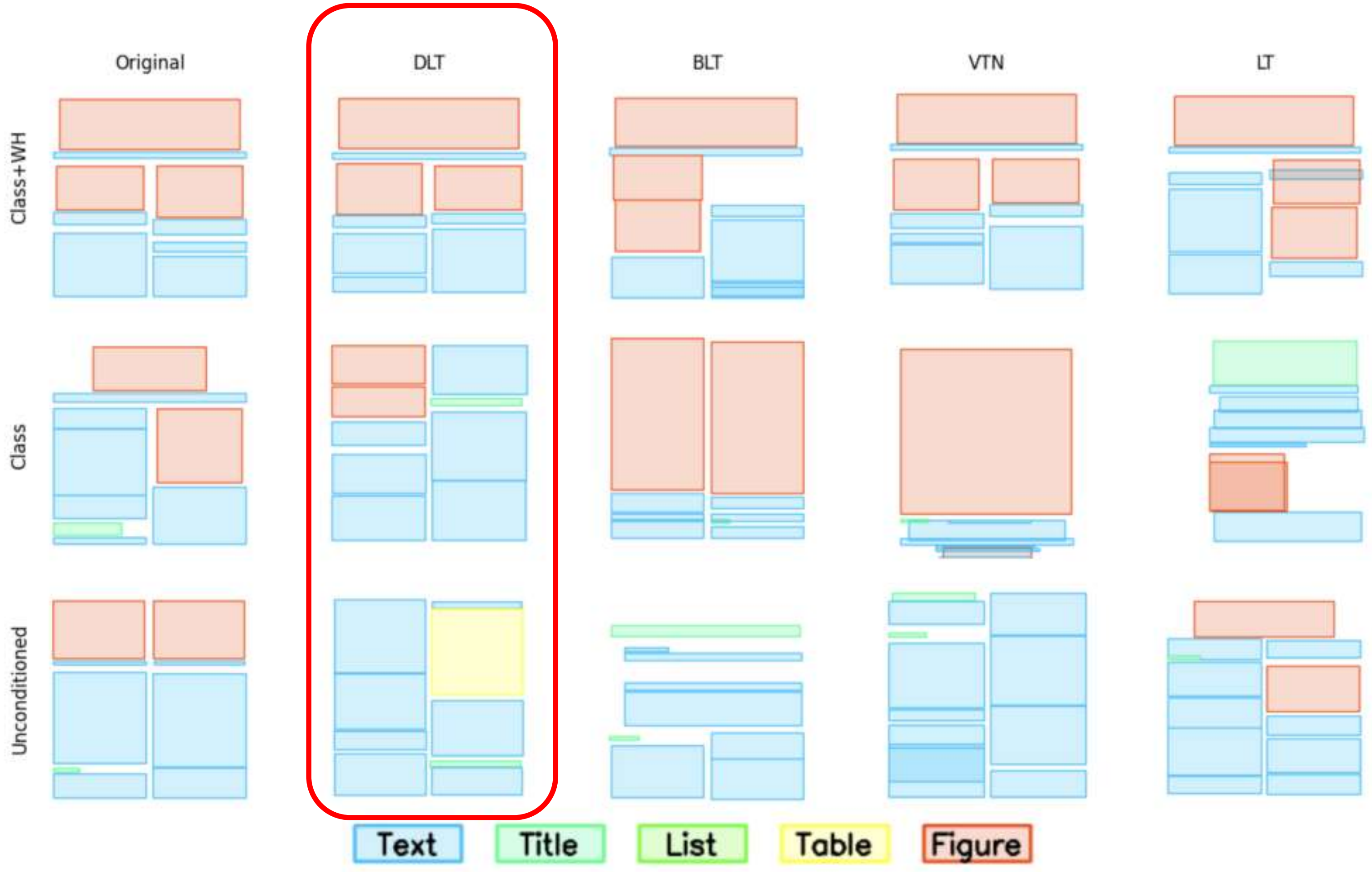
※ C: condition (true or unknown)

Experiments

Dataset	Publaynet											
	Conditioned on Category				Category + Size				Unconditioned			
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				Unconditioned			
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				Unconditioned			
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**



Experiments



Conclusion

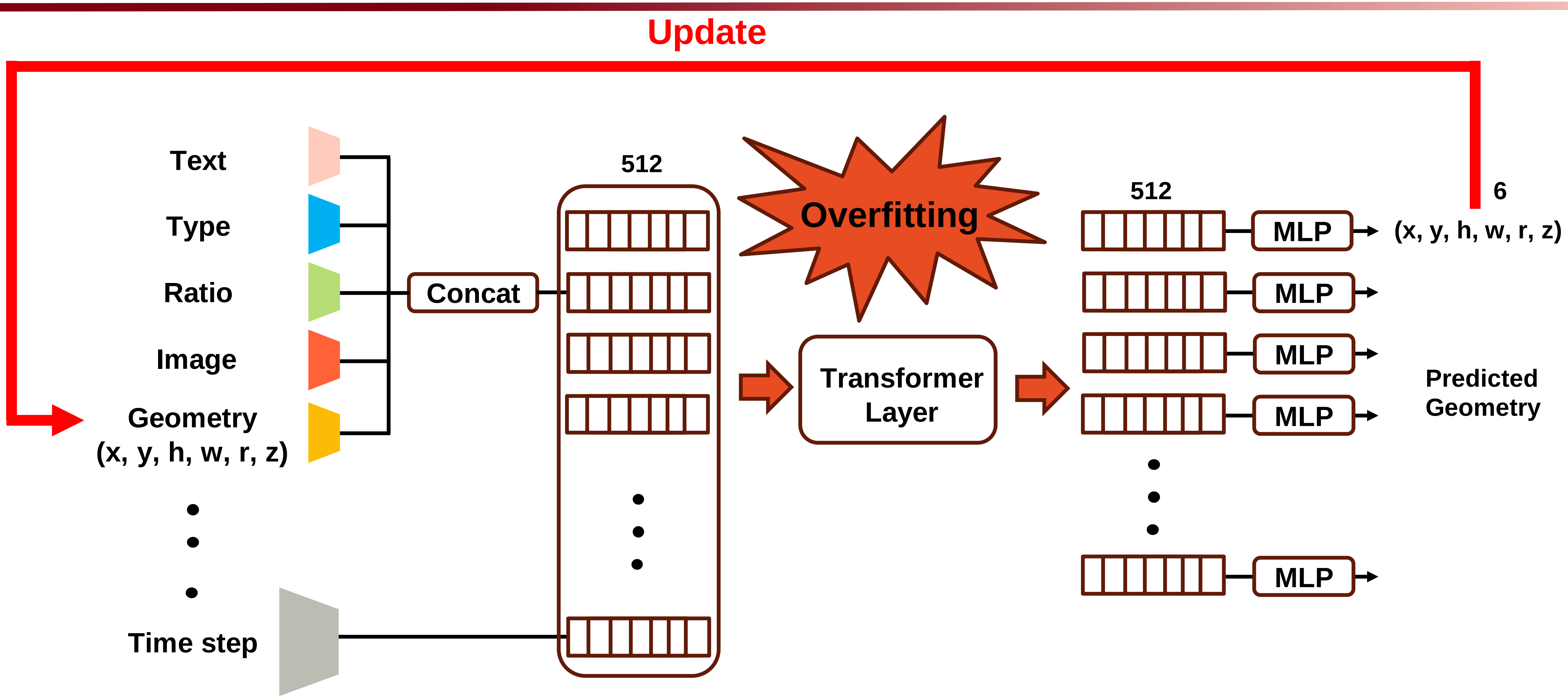
Contribution

- 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

Our model



Update
Text
Type
Ratio
Image
Geometry
(x, y, h, w, r, z)
Time step
Concat
512
Overfitting
Transformer Layer
512
MLP
MLP
MLP
MLP
MLP
6
(x, y, h, w, r, z)
Predicted Geometry