Incorporating Diffusion Models into Conditional Text Generation

Speaker: Shansan Gong

Shanghai AI Lab
hisansas@gmail.com

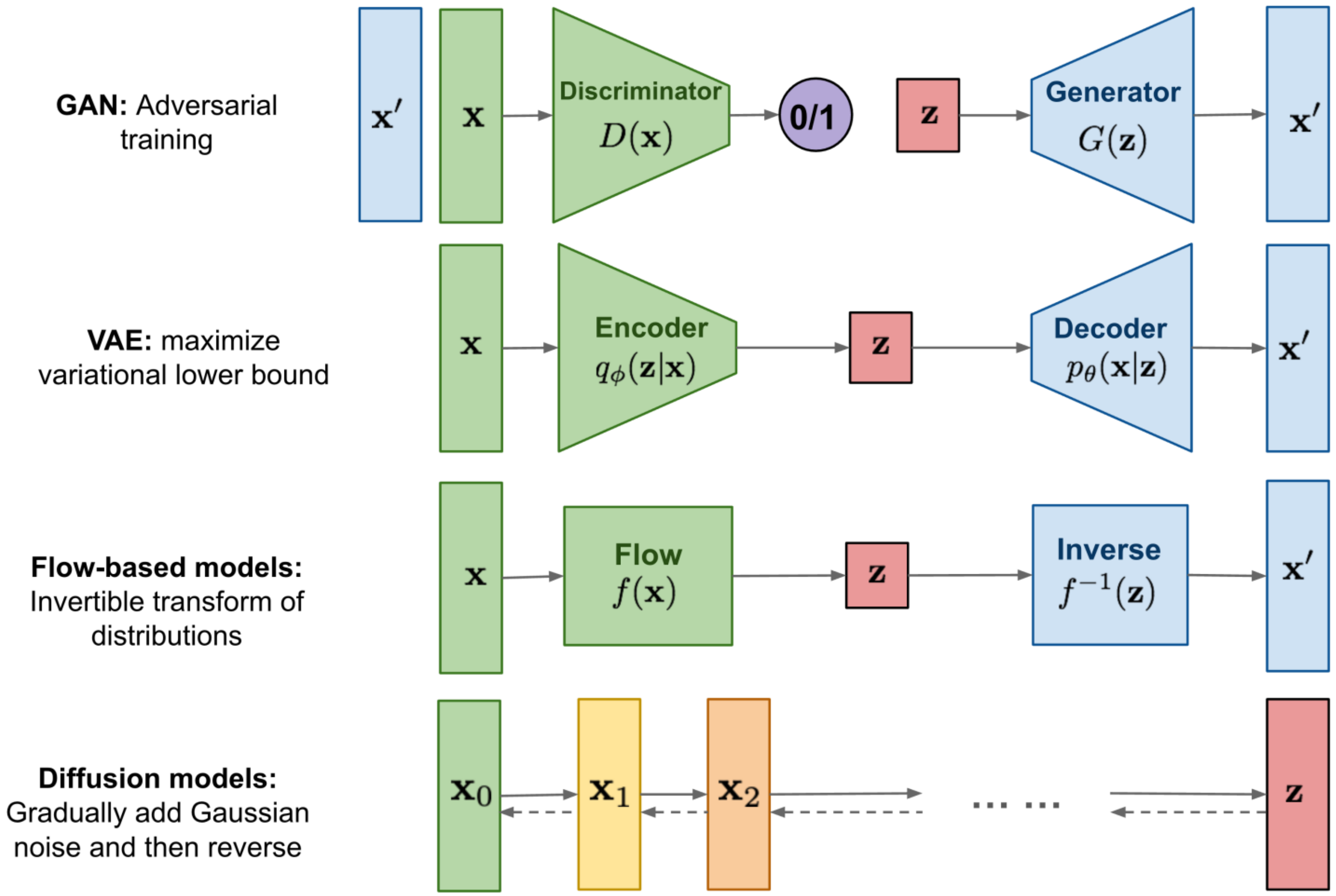
https://summmeer.github.io/

Contents

- Preliminary knowledge about diffusion models
- Related work about text generation using diffusion models
- DiffuSeq and connections to NAR/AR models
- Follow up works
- Conclusion and future work

1 Preliminary – Generative models

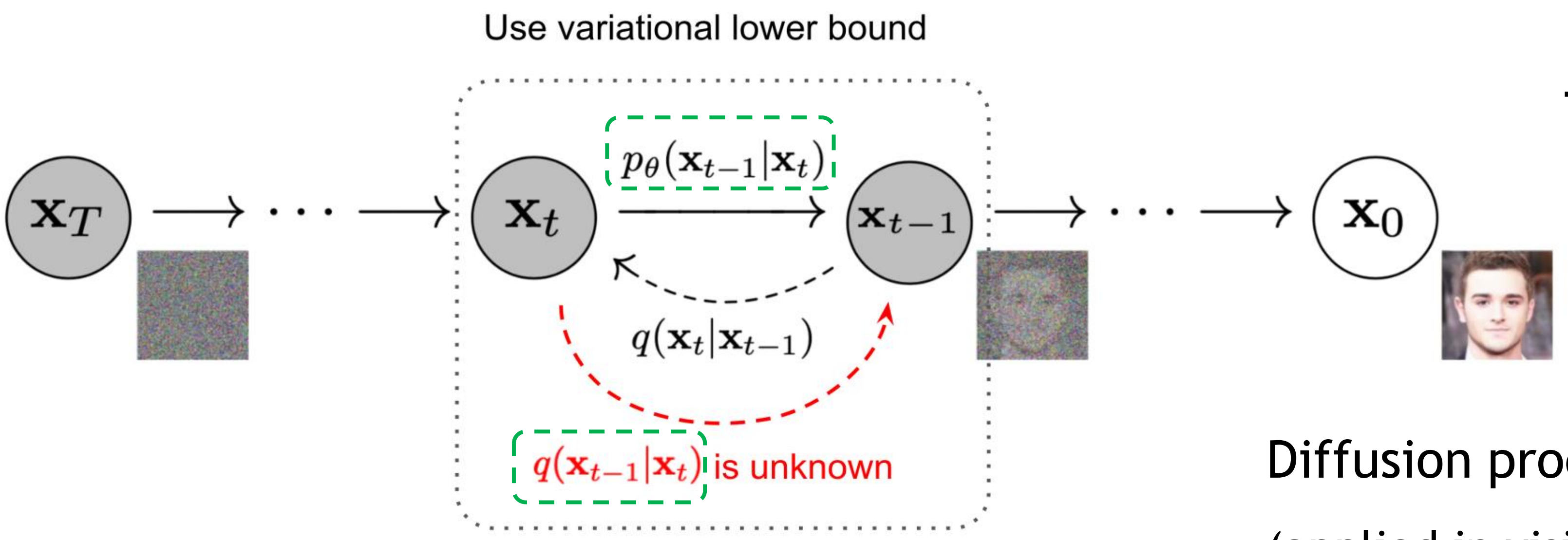
Different types of generative models



- GAN:
  - unstable training
- VAE, Flow-based:
  - rely on latent variable
- Diffusion model:
  - Many middle states, diversity, slower
- Consistency model:
  - generates in a single step

1 Preliminary – Diffusion process

Forward and backward process



← forward: add noise

→ backward: learn to denoise

Diffusion process in continuous space

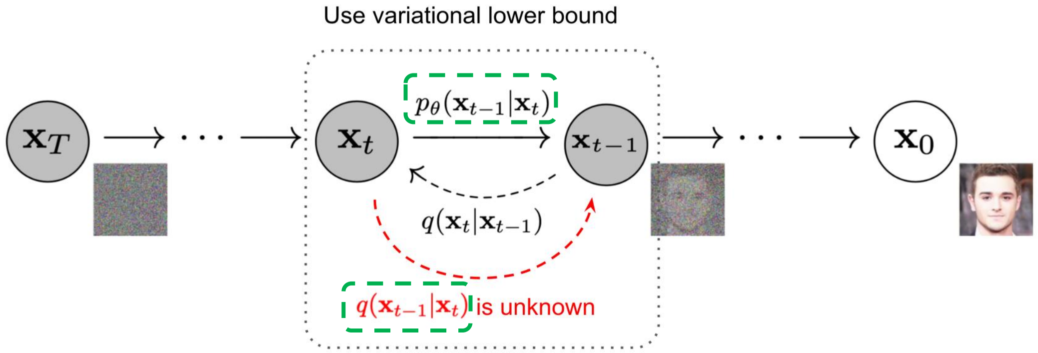
(applied in vision, audio, time series and etc.…)

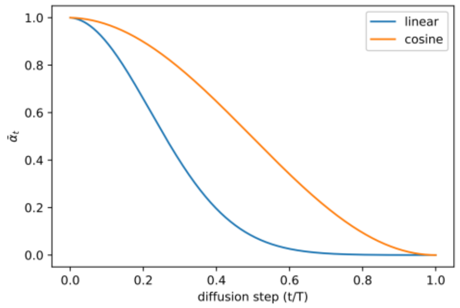
[1] https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

[2] https://github.com/heejkoo/Awesome-Diffusion-Models

[3] https://benanne.github.io/2022/05/26/guidance.html

Detailed derivation





$$\alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

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

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

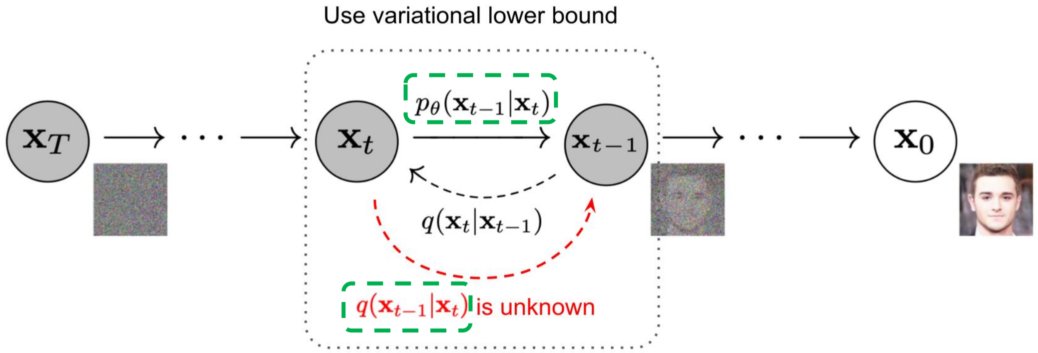
$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}$$

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

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

[1] Noise-conditioned score network (NCSN; Yang & Ermon, 2019)
[2] Denoising diffusion probabilistic models (DDPM; Ho et al. 2020)

Variational lower bound



$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

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

Train μ_θ to predict $\tilde{\mu}_t$ (mse)

$$-\log p_\theta(\mathbf{x}_0) \le -\log p_\theta(\mathbf{x}_0) + D_{\mathrm{KL}}(q(\mathbf{x}_{1:T}|\mathbf{x}_0) \| p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0))$$

$$\text{Let } L_{\mathrm{VLB}} = \mathbb{E}_{q(\mathbf{x}_{0:T})}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big] \ge -\mathbb{E}_{q(\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0)$$

$$= \mathbb{E}_q[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}]$$

1 Preliminary: Diffusion process

Diffusion process in summary:

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \mathbf{z}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

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

- Forward process:
  - $\mathbf{x}_0 \sim q(\mathbf{x}) \rightarrow \mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$
  - $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I})$
- Reverse process:
  - $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_\theta(\mathbf{x}_t, t))$
  - $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}$
- Training loss:
  - $L_t = D_{KL}(q||p_\theta)$
  - Parameterization of $L_t =$

$$\mathbb{E}_{\mathbf{x}_0}(||(\mathbf{x}_0 - f_\theta(\mathbf{x}_t, t))||^2)$$

2 Related work – Discrete space

Unlike vision or audio domain, text is discrete, can be regarded as categorical vectors

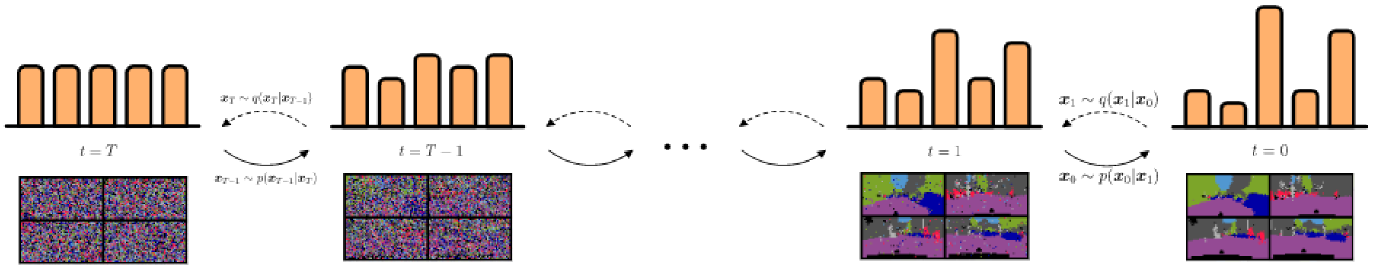
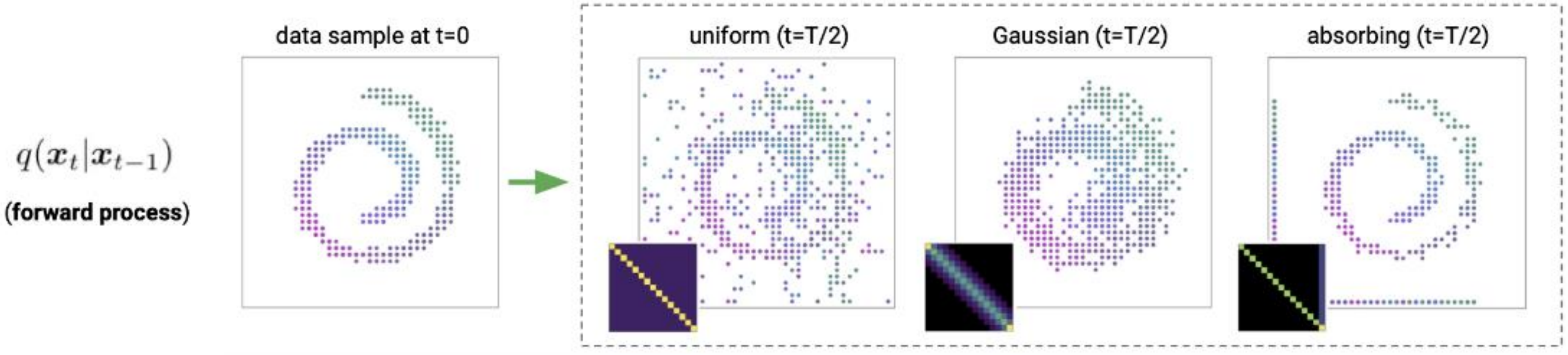


Figure 2: Overview of multinomial diffusion. A generative model $p(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ learns to gradually denoise a signal from left to right. An inference diffusion process $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ gradually adds noise form right to left.

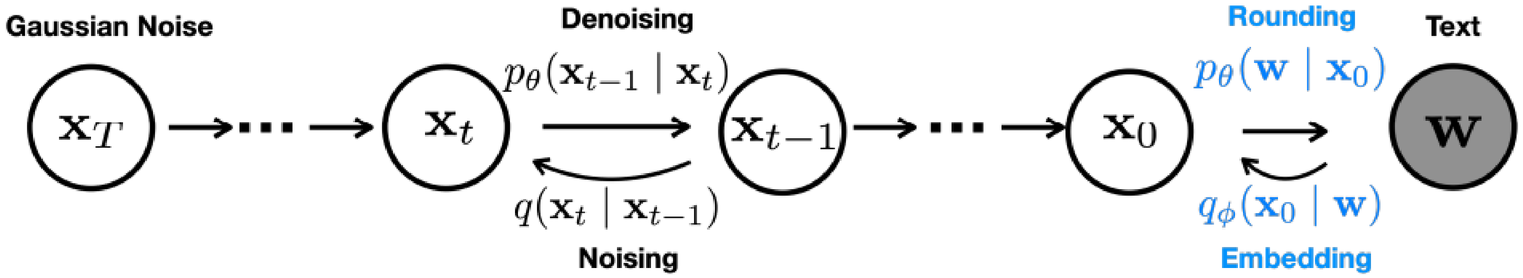


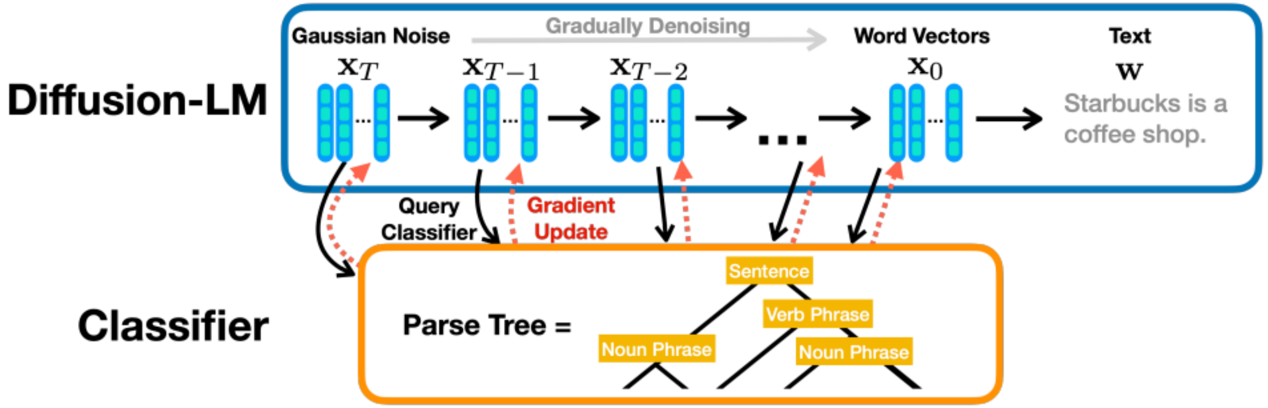
[1] Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions, NeurIPS, 2021

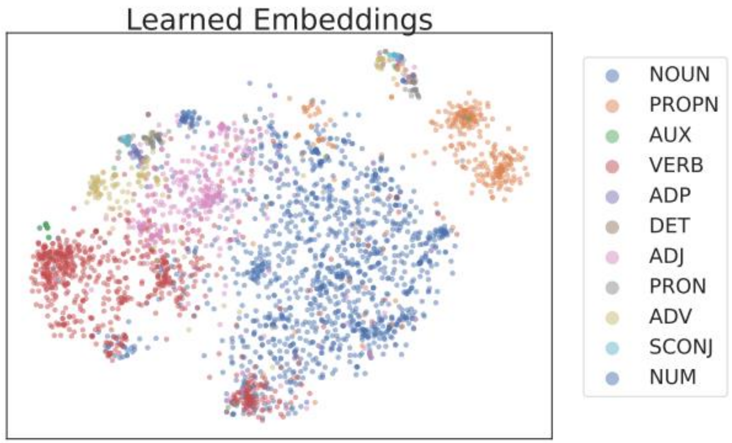
[2] Structured Denoising Diffusion Models in Discrete State-Spaces (D3PM), NeurIPS, 2021

2 Related work – Diffusion-LM

In word embedding space; classifier-guided; generation with constraints







[1] Diffusion-LM Improves Controllable Text Generation, NeurIPS, 2022

2 Related work – Text related

Text modeling

- Text-to-image: two-stage or jointly training, with one-side fixed

- Pure text modeling:

Token level ← Token prob. ← Token emb.

Token level

- D3PM (absorbing)
- SUNDAE/DiffusER (editing based)

Token prob.

- SSD-LM

Token emb.

- Diffusion-LM
- DiffuSeq
- Analog Bits

[1] Step-unrolled Denoising Autoencoders for Text Generation, ICLR, 2022

[2] SSD-LM: Semi-autoregressive Simplex-based Diffusion Language Model for Text Generation and Modular Control, 2022

[3] Analog Bits: Generating Discrete Data using Diffusion Models with Self-Conditioning, 2022

2 Related work – Classifier-free

Text-to-image: embedding space alignment;

Classifier-free training: take extra input argument for f_θ

Algorithm 1 Training. WaveGrad directly conditions on the continuous noise level $\sqrt{\bar{\alpha}}$. l is from a predefined noise schedule.

repeat
  $y_0 \sim q(y_0)$
  $s \sim \text{Uniform}(\{1, \dots, S\})$
  $\sqrt{\bar{\alpha}} \sim \text{Uniform}(l_{s-1}, l_s)$
  $\epsilon \sim \mathcal{N}(0, I)$
  Take gradient descent step on
  $\nabla_\theta \left\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}}\, y_0 + \sqrt{1-\bar{\alpha}}\, \epsilon, \boxed{x}, \sqrt{\bar{\alpha}}) \right\|_1$
until converged

[1] GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models, 2021

[2] WaveGrad: Estimating Gradients for Waveform Generation, ICLR, 2021

[3] Classifier-Free Diffusion Guidance, NeurIPS workshop, 2021

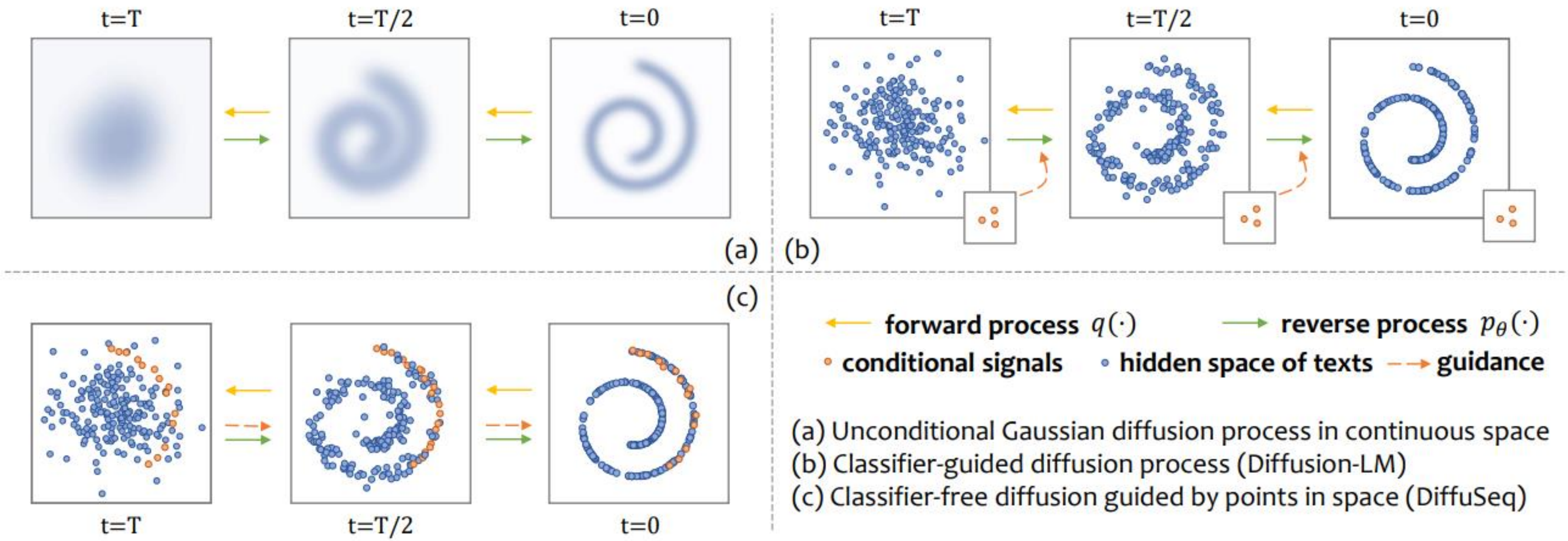
3 DiffuSeq – Motivation

From unconditional models to conditional models:

Seq2Seq tasks: $\mathbf{x} \rightarrow \mathbf{y}$

Diffusion-LM (classifier-guided) v.s. DiffuSeq (classifier-free)

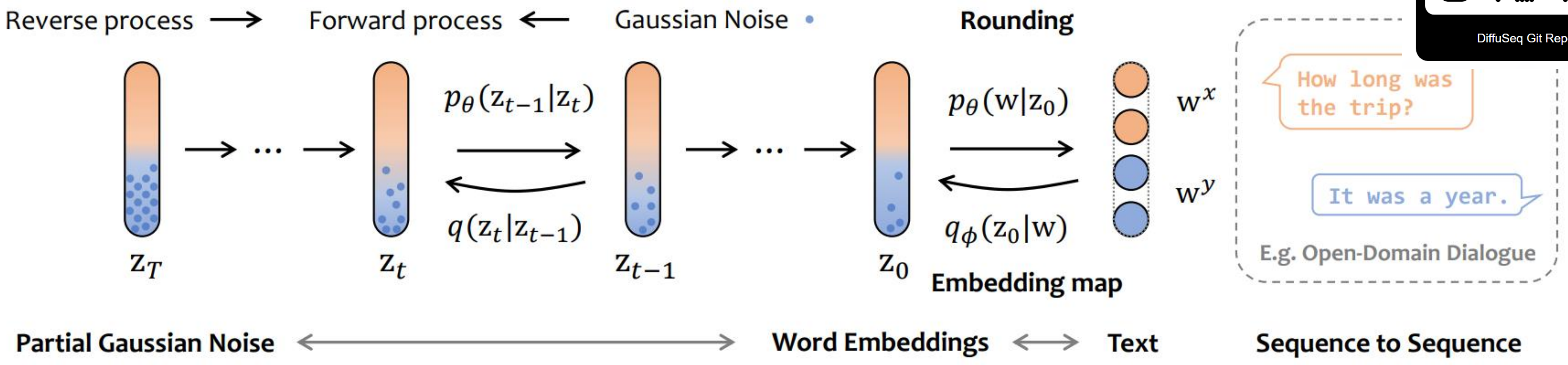
DiffuSeq Git Repo



[1] DiffuSeq: sequence to sequence text generation with diffusion models, ICLR, 2023

DiffuSeq Git Repo

Technical details:



- Forward Process with Partial Noising:
  - $q(\mathbf{z}_0|\mathbf{w}^{x\oplus y}) = \mathcal{N}(\mathrm{EMB}(\mathbf{w}^{x\oplus y}), \beta_0 \mathbf{I}); \mathbf{z}_t = \mathbf{x}_t \oplus \mathbf{y}_t$
- Reverse Process with Conditional Denoising:
  - $L_t = \mathbb{E}_{\mathbf{x}_0, \mathbf{y}_0}(||(\mathbf{y}_0 - f_{\tilde{\theta}}(\mathbf{z}_t, t))||^2)$

- Training:
  - importance sampling
- Inference:
  - Rounding to embeddings
  - Anchoring input signals

3 DiffuSeq - Experiments

DiffuSeq Git Repo

Four tasks: Dialogue, QG, Text Simplification, Paraphrase

Three groups of baselines: Plain encoder-decoder, PLMs, NAR

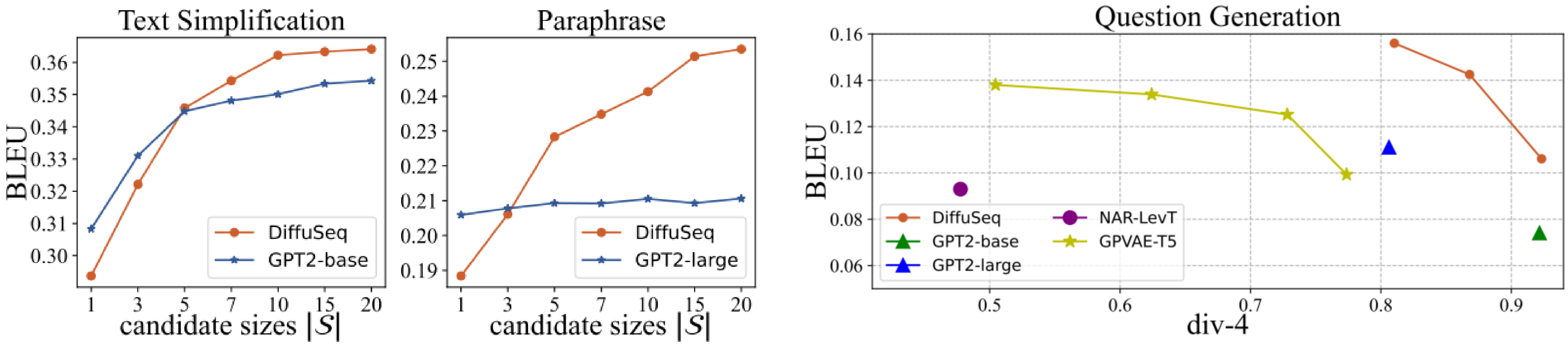
Tasks	Methods	BLEU↑	R-L↑	Score↑	dist-1↑	selfB↓ / div-4↑	Len
Paraphrase	GRU-attention ◇	0.1894	0.5129	0.7763	0.9423	0.9958/0.3287	8.30
	Transformer-base ◇	**0.2722**	0.5748	0.8381	0.9748	0.4483/0.7345	11.2
	GPT2-base FT •	0.1980	0.5212	0.8246	0.9798	0.5480/0.6245	9.67
	GPT2-large FT •	0.2059	0.5415	0.8363	**0.9819**	0.7325/0.5020	9.53
	GPVAE-T5 •	0.2409	**0.5886**	**0.8466**	0.9688	0.5604/0.6169	9.60
	NAR-LevT ‡	0.2268	0.5795	0.8344	0.9790	0.9995/0.3329	8.85
	DIFFUSEQ (Ours) ‡	0.2413	0.5880	0.8365	0.9807	**0.2732**/**0.8641**	11.2

Comparable quality, better diversity

3 DiffuSeq - Experiment analysis

DiffuSeq Git Repo

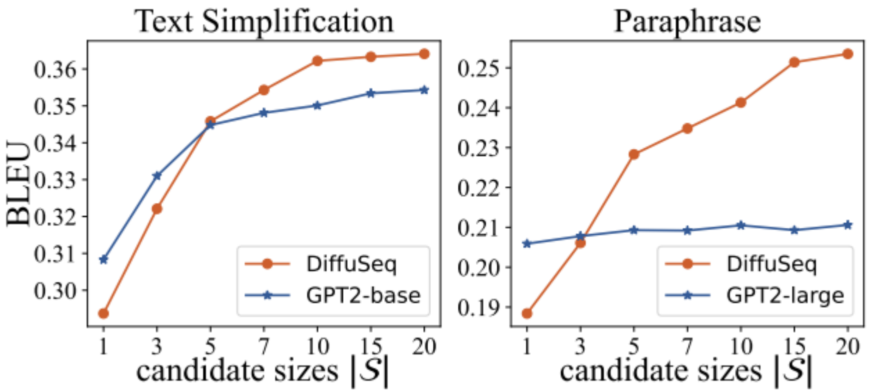
Diversity Ensures Quality

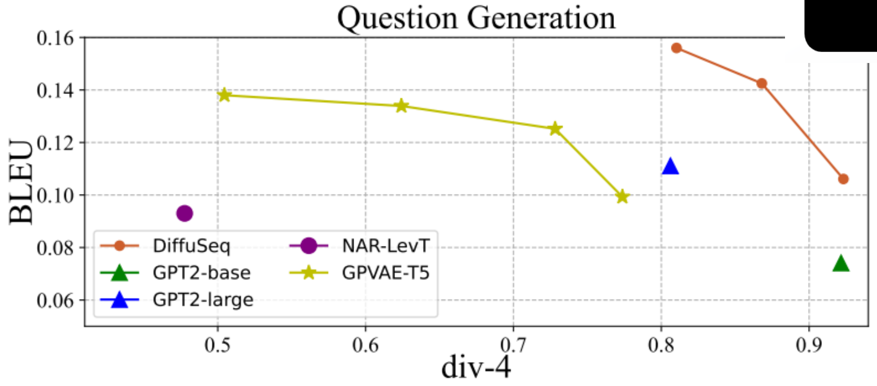


3 DiffuSeq - Experiment analysis

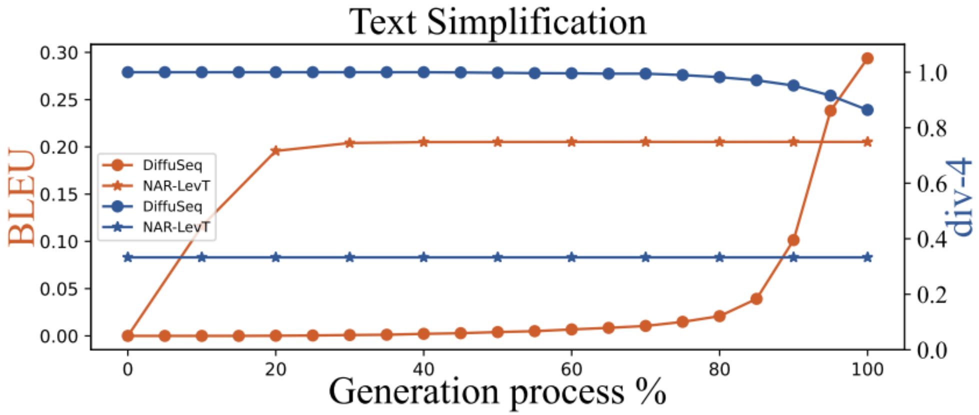
DiffuSeq Git Repo

Diversity Ensures Quality





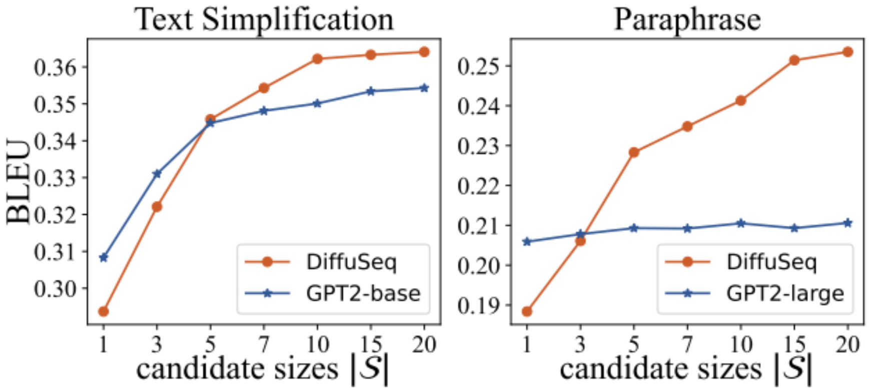
Step-wise Analysis against Iterative NAR

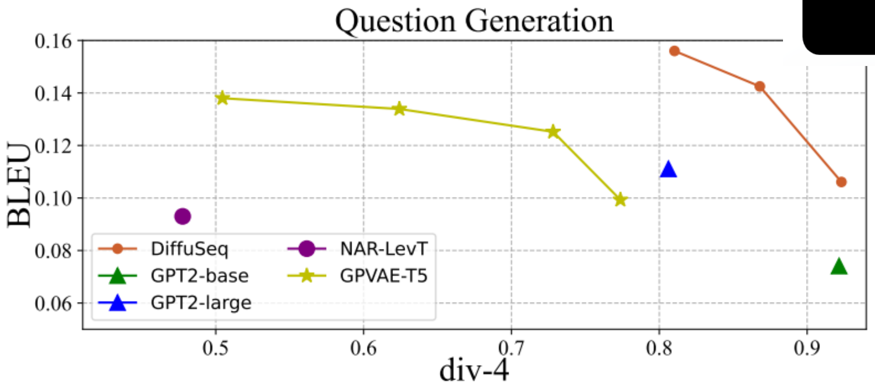


3 DiffuSeq - Experiment analysis

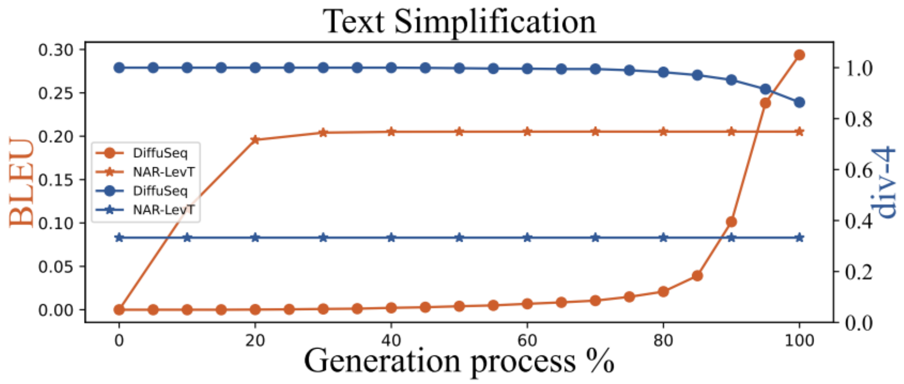
DiffuSeq Git Repo

Diversity Ensures Quality

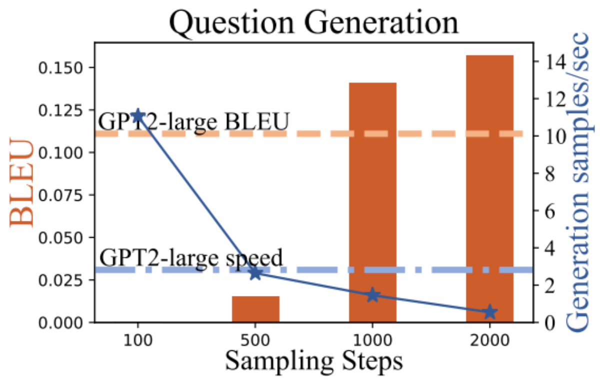




Step-wise Analysis against Iterative NAR



Inference Speed



3 DiffuSeq - Connections to NAR

DiffuSeq Git Repo

AR/iter-NAR/DiffuSeq: Generation process is along with different dimensions:

$$p_{\mathrm{AR}}(\mathbf{w}^y_{1:n}|\mathbf{w}^x) = \underbrace{p(w^y_1|\mathbf{w}^x)}_{\text{initial prediction}} \underbrace{\prod_{i=1,\ldots,n-1} p(w^y_{i+1}|\mathbf{w}^y_{1:i}, \mathbf{w}^x)}_{\text{progressive left-context prediction}},$$

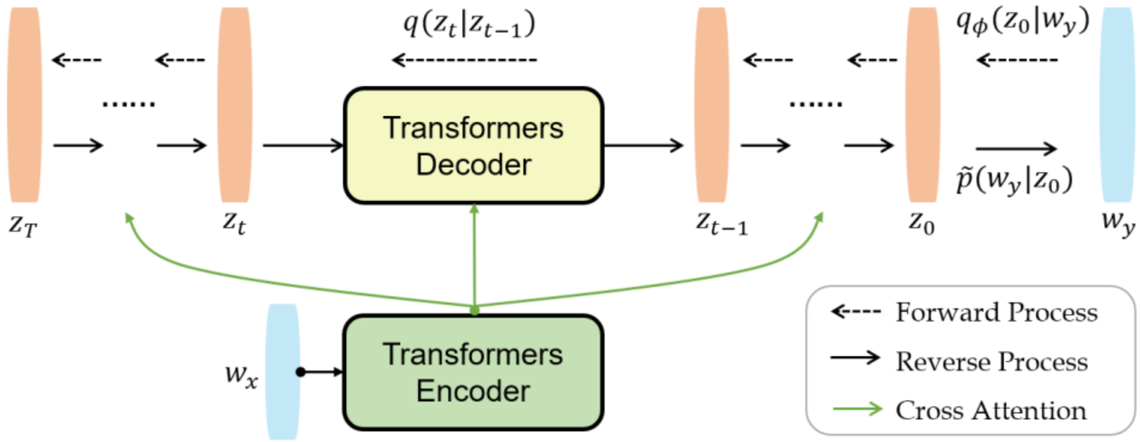
$$p_{\text{iter-NAR}}(\mathbf{w}^y_{1:n}|\mathbf{w}^x) = \sum_{\mathbf{w}^y_1,\ldots,\mathbf{w}^y_{K-1}} \underbrace{\prod_{i=1\ldots n} p(w^y_{1,i}|\mathbf{w}^x)}_{\text{initial prediction}} \underbrace{\prod_{k=1..K-1} \prod_{i=1\ldots n} p(w^y_{k+1,i}|\mathbf{w}^y_{k,1:n}, \mathbf{w}^x)}_{\text{progressive full-context prediction}}.$$

$$p_{\textsc{DiffuSeq}}(\mathbf{w}^y|\mathbf{w}^x) = \sum_{\substack{\mathbf{w}^y_T,\ldots,\mathbf{w}^y_1 \\ \mathbf{y}_T,\ldots,\mathbf{y}_0}} p(\mathbf{w}^y|\mathbf{y}_0, \mathbf{w}^x) \prod_{t=T,\ldots,1} p(\mathbf{w}^y_t|\mathbf{y}_t, \mathbf{w}^x) p(\mathbf{y}_{t-1}|\mathbf{w}^y_t)$$

4 Follow-up works - SeqDiffuSeq

Intuition:

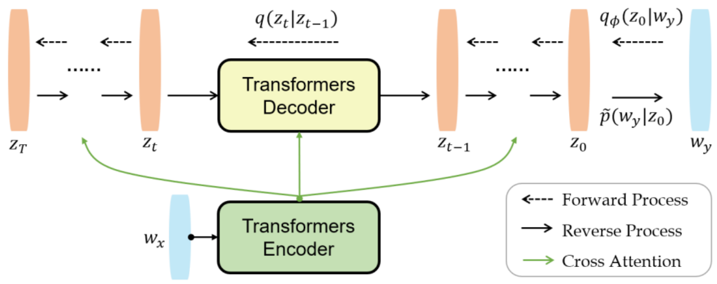
- explore diffusion models with encoder-decoder Transformers architecture for sequence-to-sequence generation.



SeqDiffuSeq: Text Diffusion with Encoder-Decoder Transformers

Hongyi Yuan[12*], Zheng Yuan[2], Chuanqi Tan[2], Fei Huang[2], Songfang Huang[2]
[1]Tsinghua University, [2]Alibaba Group
yuanhy20@mails.tsinghua.edu.cn
{yuanzheng.yuanzhen,chuanqi.tcq,f.huang,songfang.hsf}@alibaba-inc.com

4 Follow-up works - SeqDiffuSeq



Intuition:

- explore diffusion models with encoder-decoder Transformers architecture for sequence-to-sequence generation.

Main contributions:

- Self-conditioning: the denoising function takes previously estimated samples z_0^t as auxiliary inputs.
- Adaptive noise schedule: set different noise schedules for tokens at different positions according to losses

Concerns: diversity

SeqDiffuSeq: Text Diffusion with Encoder-Decoder Transformers

Hongyi Yuan[12]*, Zheng Yuan[2], Chuanqi Tan[2], Fei Huang[2], Songfang Huang[2]
[1]Tsinghua University, [2]Alibaba Group
yuanhy20@mails.tsinghua.edu.cn
{yuanzheng.yuanzhen,chuanqi.tcq,f.huang,songfang.hsf}@alibaba-inc.com

4 Follow-up works - SeqDiffuSeq

Results:

- Experiments on translation tasks
- Speed

	BLEU	BLEU-1/2/3/4
SeqDiffuSeq	30.31	62.73/36.94/24.07/16.09
-Adaptive Noise Schedule	28.94	61.39/35.44/22.82/15.12
-Self-Conditioning	25.74	58.74/31.97/19.67/12.44

Table 3: Ablation studies on IWSLT14 DE-EN validation set.

	Time	Acceleration
DiffuSeq	317 sec.	-
SeqDiffuSeq	89 sec.	$\times 3.56$

Table 4: Time needed for inference on QQP.

4 Follow-up works - GENIE

Intuition:

- pre-training has been proven effective and encoder-decoder model architecture is the most popular pre-train paradigm.

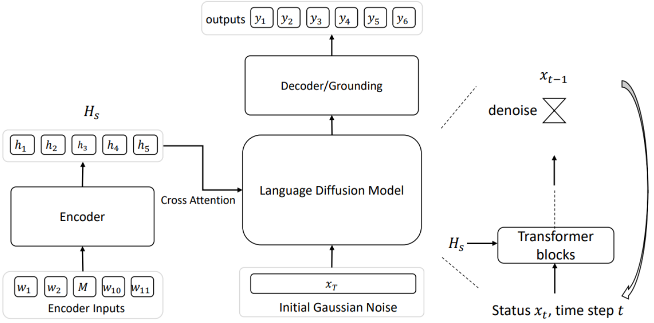
Main contributions:

- First pre-trained model using CPD (continuous paragraph denoise)

Text Generation with Diffusion Language Models: A Pre-training Approach with Continuous Paragraph Denoise

Zhenghao Lin [1 2] Yeyun Gong [3] Yelong Shen [4] Tong Wu [5 2] Zhihao Fan [6 2] Chen Lin [1] Nan Duan [3] Weizhu Chen [4]

[1]Xiamen University [2]This work was done during an internship in MSRA [3]Microsoft Research Asia [4]Microsoft [5]Tsinghua University [6]Fudan University. Correspondence to: Chen Lin <chenlin@xmu.edu.cn>.



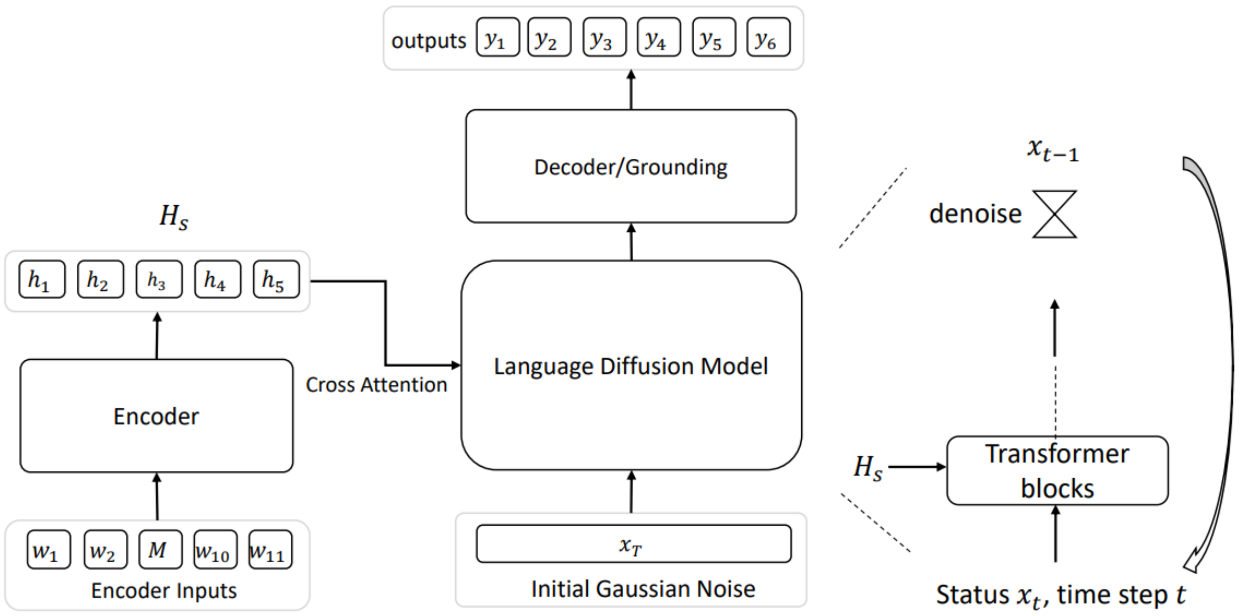
4 Follow-up works - GENIE

Intuition:

- pre-training has been proven effective and encoder-decoder model architecture is the most popular pre-train paradigm.

Main contributions:

- First pre-trained model using CPD (continuous paragraph denoise)



4 Follow-up works - GENIE

Results:

- Experiments on summarization tasks

Table 1. Results of Semi-NAR, NAR and AR on XSUM. Index **OVERALL** represents the average value of **ROUGE-1**, **ROUGE-2** and **ROUGE-L**. It should be noted that GENIE belongs to Semi-NAR.

Methods	Pattern	XSUM			
		ROUGE-1	ROUGE-2	ROUGE-L	OVERALL
NAT (Gu et al., 2017)	NAR	24.0	3.9	20.3	16.1
iNAT (Lee et al., 2018)	NAR	24.0	4.0	20.4	16.1
CMLM (Ghazvininejad et al., 2019)	NAR	23.8	3.6	20.2	15.9
LevT (Gu et al., 2019b)	NAR	24.8	4.2	20.9	16.6
BANG (Qi et al., 2021)	NAR	32.6	9.0	27.4	23.0
ELMER (Li et al., 2022a)	NAR	**38.3**	**14.2**	**29.9**	**27.5**
LSTM (Greff et al., 2017)	AR	25.1	6.9	19.9	17.3
Transformer (Vaswani et al., 2017b)	AR	30.7	10.8	24.5	22.0
MASS (Song et al., 2019)	AR	39.7	17.2	31.9	29.6
BART (Lewis et al., 2019)	AR	39.8	17.2	32.2	29.7
ProphetNet (Qi et al., 2020)	AR	39.9	17.1	32.1	29.7
BANG (Qi et al., 2021)	AR	**41.1**	**18.4**	**33.2**	**30.9**
InsT (Stern et al., 2019)	Semi-NAR	17.7	5.2	16.1	13.0
iNAT (Lee et al., 2018)	Semi-NAR	27.0	6.9	22.4	18.8
CMLM (Ghazvininejad et al., 2019)	Semi-NAR	29.1	7.7	23.0	20.0
LevT (Gu et al., 2019b)	Semi-NAR	25.3	7.4	21.5	18.1
BANG (Qi et al., 2021)	Semi-NAR	34.7	11.7	29.2	25.2
GENIE (w/o pre-train)	Semi-NAR	38.9	17.5	31.0	29.1
GENIE	Semi-NAR	**42.9**	**21.4**	**35.1**	**33.2**

4 Follow-up works – RDM

Intuition:

- Continuous diffusion models suffer from a slow runtime and discrete diffusion models are under-explored.

Main contributions:

- route-and-denoise process: at each iteration, each token within the sequence is either denoised or reset to noisy states according to an underlying stochastic routing mechanism

A Reparameterized Discrete Diffusion Model for Text Generation

Lin Zheng[1] **Jianbo Yuan**[2] **Lei Yu**[3] **Lingpeng Kong**[1]

[1]Department of Computer Science, The University of Hong Kong [2]ByteDance Inc. [3]DeepMind. Correspondence to: Lin Zheng <linzheng@connect.hku.hk>.

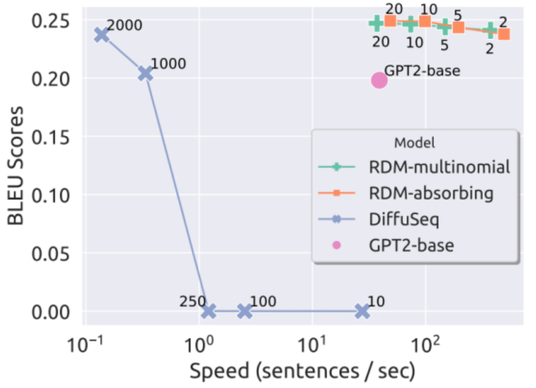
4 Follow-up works – RDM

Results:

- Experiments on machine translation tasks

Table 1: BLEU score comparisons on IWSLT14 DE-EN, WMT14 EN-DE, and WMT16 EN-RO benchmarks. * denotes results reported from previous work.

	Model	# Iterations	IWSLT14 DE-EN Vanilla	IWSLT14 DE-EN Reparam.	WMT16 EN-RO Vanilla	WMT16 EN-RO Reparam.	WMT14 EN-DE Vanilla	WMT14 EN-DE Reparam.
Continuous Diffusion	CDCD (Dieleman et al., 2022)	200	–		–		20.0*	
Discrete Diffusion	Multinomial Diffusion (Hoogeboom et al., 2021)	2	23.05	28.01	26.61	30.16	4.28	21.43
		4	24.24	30.57	27.81	31.70	4.31	24.05
		10	21.28	32.23	25.25	33.00	6.94	25.63
		16	20.59	32.58	24.36	33.11	6.07	25.64
		25	20.06	32.84	23.94	33.31	3.69	26.04
	Absorbing Diffusion (Austin et al., 2021)	2	25.24	27.60	27.24	30.72	16.46	21.00
		4	26.93	31.47	29.16	32.60	19.48	24.26
		10	28.32	33.91	30.41	33.38	21.62	26.96
		16	28.38	34.41	30.79	33.82	22.07	27.58
		25	28.93	34.49	30.56	33.99	22.52	**27.59**
Auto-regressive Models	Transformer-base (Vaswani et al., 2017)	n.a.	**34.51**		**34.16**		27.53	



5 Conclusion and future work

- Diffusion models as a new generation paradigm bring new possibilities to AR supremacy in text generation domain
  - Diverse
  - Editable
- Future work: inference speed and generation quality

Thank you for listening!

Speaker: Shansan Gong

Shanghai AI Lab
hisansas@gmail.com

https://summmeer.github.io/

https://github.com/Shark-NLP/DiffuSeq

DiffuSeq Git Repo