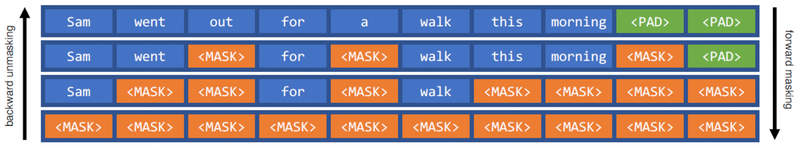
Beyond Masks: Efficient, Flexible Diffusion Language Models via Deletion-Insertion Processes

Fangyu Ding*, Ding Ding*, Sijin Chen*, Kaibo Wang, Peng Xu, Zijin Feng, Haoli Bai, Kai Han, Youliang Yan, Binhang Yuan, Jiacheng Sun
HKUST, Huawei, CUHK

Introduction

We propose a novel deletion-insertion diffusion language model (DID) beyond the state-of-the-art masked diffusion language model (MDLM) on **computation efficiency** and **generation flexibility**.



(a) MDLMs, sequences padded to length 10.

(b) DID.

Computational Efficiency

- More than 50% FLOPs spent on <MASK> and <PAD> in MDLM training and inference can be eliminated in DID.

Generation Flexibility

- DID brings about a positional self-correction mechanism.
- Variable-length text generation with pure diffusion.

Background

As a discrete diffusion language model, DID is formulated in the framework of continuous-time Markov chain (CTMC), and trained with the denoising score entropy (DSE) loss:

$$\mathcal{L}_\theta^{\text{DSE}}(\boldsymbol{x}_0) = \mathop{\mathbb{E}}_{t,\boldsymbol{x}_t} \sum_{\boldsymbol{y}\neq\boldsymbol{x}_t} Q_t(\boldsymbol{y},\boldsymbol{x}_t)\left[s_\theta(\boldsymbol{x}_t,t)_{\boldsymbol{y}} - \frac{p_{t|0}(\boldsymbol{y}|\boldsymbol{x}_0)}{p_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)} \log s_\theta(\boldsymbol{x}_t,t)_{\boldsymbol{y}} + K\left(\frac{p_{t|0}(\boldsymbol{y}|\boldsymbol{x}_0)}{p_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\right)\right]$$

where the concrete score $s_\theta(\boldsymbol{x}_t,t)_{\boldsymbol{y}}$ is the core item to learn.

Methodology

We formulate token deletion & insertion as discrete diffusion forward & backward processes for DID, to replace the masking & unmasking in MDLM. (as shown in Fig. a & b)

Forward Process: Deletion

$$p_{t+\Delta t|t}(v'|v) = \begin{cases} \sigma(t)\Delta t, & v' = \varnothing, \\ 1-\sigma(t)\Delta t, & v' = v. \end{cases}$$

Each token is deleted with a rate of $\sigma(t)$, while in MDLM, this is the masking rate.

$$p_{t|s}(\boldsymbol{x}_t|\boldsymbol{x}_s) = (1-e^{-(\bar\sigma(t)-\bar\sigma(s))})^{|\boldsymbol{x}_s|-|\boldsymbol{x}_t|} e^{-(\bar\sigma(t)-\bar\sigma(s))|\boldsymbol{x}_t|} N(\boldsymbol{x}_t,\boldsymbol{x}_s)$$

Notably, the sequence level probability contains a distinct subsequence count term $N(\boldsymbol{x}_t,\boldsymbol{x}_s)$, which is a classic dynamic programming problem with a quadratic time complexity.

$$Q_t(\boldsymbol{y},\boldsymbol{x}_t) = \lim_{\Delta t\to 0} \frac{p_{t+\Delta t|t}(\boldsymbol{x}_t|\boldsymbol{y})}{\Delta t} = \sigma(t) N(\boldsymbol{x}_t,\boldsymbol{y})$$

Backward Process: Insertion

The core item here is the insertion score we defined as:

$$\bar s(\boldsymbol{x}_t,t)[i,v] \overset{\text{def}}{=} \frac{\mathbb{E}_{\boldsymbol{x}_0}[(1-e^{-\bar\sigma(t)})^{|\boldsymbol{x}_0|} N(\text{Ins}(\boldsymbol{x}_t,i,v),\boldsymbol{x}_0)]}{\mathbb{E}_{\boldsymbol{x}_0}[(1-e^{-\bar\sigma(t)})^{|\boldsymbol{x}_0|} N(\boldsymbol{x}_t,\boldsymbol{x}_0)]}$$

with $s(\boldsymbol{x}_t,t)_{\boldsymbol{y}} = \dfrac{e^{-\sigma(t)}}{1-e^{-\bar\sigma(t)}} \dfrac{1}{N(\boldsymbol{x}_t,\boldsymbol{y})} \displaystyle\sum_{i\in I(\boldsymbol{x}_t,\boldsymbol{y})} \bar s(\boldsymbol{x}_t,t)[i,v(\boldsymbol{x}_t,\boldsymbol{y})],$

Which leads to theoretically sound **(I)** and **(II)**:

(I) Sampling Algorithm with Insertion Score

Probability to insert token v after position i at time t:

$$p^\theta_{t-\Delta t|t}((i,v)|\boldsymbol{x}_t) = \begin{cases} \frac{\sigma(t)e^{-\bar\sigma(t)}}{1-e^{-\bar\sigma(t)}} \bar s_\theta(\boldsymbol{x}_t,t)[i,v]\Delta t, & v\neq\varnothing \\ 1-\sum_{w\neq\varnothing} p^\theta_{t-\Delta t|t}((i,w)|\boldsymbol{x}_t), & v=\varnothing \end{cases}$$

(II) Training Objective for Insertion Score

By instantiating the DSE loss with the DID terms, we get a proper diffusion NELBO for DID:

$$\mathcal{L}_\theta^{\text{DISE}}(\boldsymbol{x}_0) = \mathop{\mathbb{E}}_{t,\boldsymbol{x}_t}\left\{ \frac{\sigma(t)e^{-\bar\sigma(t)}}{1-e^{-\bar\sigma(t)}} \sum_{i,v}\left[\bar s_\theta(\boldsymbol{x}_t,t)[i,v] - \frac{N(\text{Ins}(\boldsymbol{x}_t,i,v),\boldsymbol{x}_0)}{N(\boldsymbol{x}_t,\boldsymbol{x}_0)} \log \bar s_\theta(\boldsymbol{x}_t,t)[i,v] + C\right]\right\}$$

Crucially, the training target is the subsequence count ratio (N-ratio), which can be efficiently solved by a parallel dynamic programming with a minor overhead.

Experiments

Perplexities, Generation Diversity, Generation Speed

Table 1: Zero-shot language modeling perplexity. Results for diffusion models are perplexity upper bounds.

Size	Method	WikiText	Lambada	Pubmed	AG News	LM1B	Arxiv	PTB
Small	RADD	38.27	51.82	56.99	73.18	72.99	85.95	**108.79**
	DID-S	38.72	49.10	55.02	76.02	74.04	82.41	115.37
	DID-F	**36.91**	**48.00**	**52.89**	**71.48**	**72.04**	**78.38**	111.60
Medium	RADD	28.44	44.10	41.06	48.96	60.32	66.28	**81.05**
	DID-S	29.19	41.94	40.84	52.53	59.88	63.95	91.87
	DID-F	**28.35**	**41.00**	**38.71**	**48.84**	**58.05**	**61.77**	87.09

Table 2: Generative perplexity (PPL, evaluated by GPT2 Large), unigram entropy, inference time (in seconds), speedup, and average generation length for fixed-length models under different total denoising steps.

Method	Steps	16	32	64	128	256	512	1024
RADD	PPL	284.78	155.01	111.56	95.10	87.56	**84.00**	**84.05**
	Entropy	8.35	8.26	8.20	8.15	8.11	8.10	8.09
	Time (s)	0.220	0.317	0.499	0.879	1.644	2.882	4.512
DID	PPL	**158.93**	**110.06**	**97.32**	**91.25**	**86.98**	86.04	85.35
	Entropy	8.15	8.13	8.13	8.12	8.09	8.08	8.09
	Time (s)	**0.169**	**0.246**	**0.353**	**0.573**	**1.047**	**1.826**	**3.006**
	Speedup	1.30×	1.29×	1.41×	1.53×	1.57×	1.58×	1.50×
	Length	1023.29	1024.01	1024.18	1024.07	1023.91	1024.03	1024.10

Training Speed

Table 3: Average training time (in seconds) per 50 steps (i.e. batches) on OpenWebText.

	Small	Medium	Large
RADD	26.46	53.17	92.90
DID	**14.03**	**27.77**	**46.60**
Speedup	1.89×	1.91×	1.99×

Table 5: Average training time (in seconds) per 50 steps on Stories.

	Small	Medium	Large
RADD	19.93	37.87	67.75
DID	**7.71**	**12.30**	**19.83**
Speedup	2.58×	3.08×	3.42×

Table 4: Generative PPL, unigram entropy, inference time (in seconds), and average generation length for variable-length models under different denoising steps. *: as outliers significantly affect PPL, only samples with PPL < 300 are counted, †: speedup over RADD.

Method	Steps	64	128	256	512
ILM	PPL*	161.80	137.64	42.29	31.14
	Entropy	5.20	5.65	5.97	6.01
	Time (s)	**0.016**	**0.034**	**0.087**	**0.271**
	Length	63.34	120.77	206.44	234.44
RADD	PPL*	81.92	50.89	34.47	26.78
	Entropy	5.22	5.58	5.79	5.85
	Time (s)	0.246	0.441	0.827	1.461
	Length	110.66	200.73	349.54	353.47
DID	PPL	**22.78**	**21.07**	**21.90**	**23.88**
	Entropy	5.90	5.94	5.94	5.94
	Time (s)	0.090	0.132	0.218	0.388
	Speedup†	2.73×	3.34×	3.79×	3.76×
	Length	182.31	193.77	202.97	204.96

Generation Length Distribution

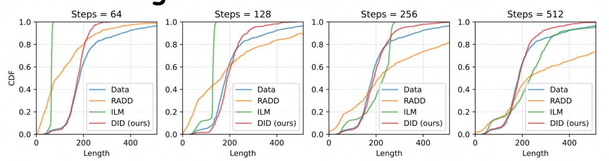


Figure 2: Cumulative distribution functions (CDFs) of generation length under different total denoising steps.

Downstream Task Evaluation

Table 20: Downstream task evaluation of 1.1B models. Zero-shot accuracy (%) is reported for each benchmark.

Method (FLOPs)	Arc-c	Arc-e	Hellaswag	Obqa	Piqa	Race	Siqa	Winogrande	Average
SMDM (1.6e21)	24.57	48.91	44.37	31.20	65.23	33.78	39.00	52.88	42.49
DID (8e20)	25.17	47.73	40.79	31.20	64.58	31.48	38.33	51.62	41.36
DID (1.2e21)	26.11	49.20	43.98	32.80	65.72	32.06	37.97	54.62	**42.81**
DID (1.6e21)	26.54	49.37	45.72	32.00	66.16	32.44	38.89	54.85	**43.25**

Table 21: GSM8K evaluation under different top-p sampling strategies. Zero-shot accuracy (%) is reported.

Top-p strategy	Steps	8	16	32	64	128	256
$p=1.0$ (direct sampling)	SMDM	27.82	33.97	35.78	36.85	36.54	36.69
	DID	**31.92**	**35.41**	**37.45**	**39.35**	**39.27**	**39.88**
$p=0.9$	SMDM	32.22	36.92	37.38	39.58	39.58	39.65
	DID	**32.37**	**38.51**	**42.38**	**41.55**	**42.23**	**43.75**
$p=0.6$	SMDM	36.01	40.03	40.56	42.00	43.37	42.61
	DID	**38.82**	**41.39**	**43.90**	**45.26**	**46.47**	**46.70**
$p=0.3$	SMDM	37.83	39.88	42.23	43.82	44.73	43.37
	DID	**38.89**	**42.08**	**43.21**	**44.81**	**45.49**	**45.79**