Diffusion Glancing Transformer for Parallel Sequence-to-Sequence Learning

Lihua Qian[1] **Mingxuan Wang**[1] **Yang Liu**[2] **Hao Zhou**[2,3]
[1] ByteDance [2]Institute for AI Industry Research (AIR), Tsinghua University
[3] Shanghai Artificial Intelligence Laboratory
{qianlihua,wangmingxuan.89}@bytedance.com
liuyang2011@tsinghua.edu.cn
zhouhao@air.tsinghua.edu.cn

Abstract

Previously, non-autoregressive models were widely recognized as being superior in generation efficiency but inferior in generation quality due to the challenges of modeling multiple target modalities. To enhance the multi-modality modeling ability, we propose the diffusion glancing transformer (DIFFGLAT), which employs a modality diffusion process and residual glancing sampling. The modality diffusion process is a discrete process that interpolates the multi-modal distribution along the decoding steps, and the residual glancing sampling approach guides the model to continuously learn the remaining modalities across the layers. Experimental results on various machine translation and text generation benchmarks demonstrate that DIFFGLAT achieves better generation accuracy while maintaining fast decoding speed compared with both autoregressive and non-autoregressive models.

1 Introduction

The Transformer (Vaswani et al., 2017) has been the most widely used architecture in sequence generation, outperforming its counterparts in (almost) all downstream tasks, such as machine translation and question answering (Brown et al., 2020; OpenAI, 2023). The typical transformer architecture adopts the autoregressive decoding strategy (AR), generating tokens in a predefined order, i.e., left to right. Recently, non-autoregressive generation models (NAR) attract increasing attention for their fast generation speed, which are *considered* to sacrifice generation quality by generating the outputs simultaneously instead of sequentially.

However, we argue that NAR has advantages compared to AR *beyond generation efficiency* for the following reasons: 1) the parallel generation enables NAR to remove the *inductive bias* of the *predefined generation order*, thereby liberating the potential of applications, such as molecules or proteins that have no well-defined orders; 2) furthermore, NAR can utilize *bi-directional context* for sequence modeling and generation, whereas AR could only exploit context from one direction.

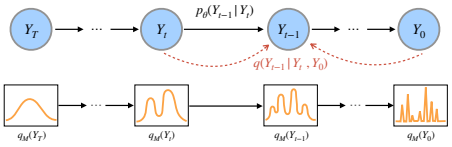

Figure 1: Our modality diffusion process. $q(\boldsymbol{y}_t)$ represents the marginal distribution of $\boldsymbol{y}_t$, which visualizes the modality distribution. The number of modalities increases as the timestep t decreases.

Although being promising, current state-of-the-art NAR models still fall behind the AR counterpart in terms of generation quality. The main drawback of NAR lies in the difficulty of modeling multi-modal distributions (Gu et al., 2018), i.e., one input has multiple valid outputs. For example, a source sentence could be translated into multiple different target sentences. As the NAR model decodes all the tokens in parallel, it may mix tokens from different translation targets. In contrast, the AR model can output consistency tokens more easily as every prediction is conditioned on previous ones. Recent progress of NAR has significantly improved the multi-modal learning ability of parallel generation, including knowledge distillation (Kim and Rush, 2016), iterative decoding (Lee et al., 2018), latent variable modeling (Ma et al., 2019) and revised learning objectives (Libovický and Helcl, 2018; Ghazvininejad et al., 2020a; Du et al., 2021; Huang et al., 2022b). However, these NAR models still hardly outperform the AR baseline consistently.

In this paper, we propose DIFFGLAT, which shows that NAR can outperform AR in both *efficiency* and *accuracy*, without requiring knowledge distillation from AR. Generally DIFFGLAT is designed within the denoising diffusion implicit model framework (Song et al., 2021a).

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 4846–4862

First, to reduce the difficulty of learning multi-modalities for NAR, DIFFGLAT defines a discrete *modality diffusion process* that *smoothly* decomposes the modality learning in the data across many diffusion transition steps. With modality decomposition, each diffusion transition only includes a scheduled number of modalities, which makes the modality learning much easier for NAR. And our preliminary experiments confirm the effectiveness of modality decomposition (See Section 2.2). Note that each diffusion transition is learned by several adjacent NAR layers, and thus DIFFGLAT can stack a sufficient number of layers to model complex multi-modal distributions.

Besides, we proposes a *residual glancing sampling* technique, which adaptively adjusts the learning difficulty of each diffusion transition in a layer-wise and residual manner.

Experiments demonstrate that the proposed DIFFGLAT significantly improves the quality of NAR generation on standard benchmarks. Using only 1 decoding iteration, DIFFGLAT can outperform all NAR baselines, including iterative ones. With 3 iterative decoding steps, DIFFGLAT beats the strong AR baseline with a moderate margin (+0.47 BLEU). Comparisons between DIFFGLAT and AR Transformer on 10 standard machine translation benchmarks, with both Transformer base and big settings, consistently verify the effectiveness of DIFFGLAT. Additionally, we also find that DIFFGLAT can slightly outperform the AR baseline on image captioning and paraphrasing tasks. These results show the extensive potential of DIFFGLAT in other generation tasks.

2 Preliminary

Given the input sequence $\boldsymbol{x} = x^1, x^2, ..., x^m$ and the target sequence $\boldsymbol{y} = y^1, y^2, ..., y^n$, the NAR model factorizes the joint probability $P(\boldsymbol{y}|\boldsymbol{x})$ with the conditional independence assumption:

$$p(\boldsymbol{y}|\boldsymbol{x};\theta) = \prod_{i=1}^{n} p(y^i|\boldsymbol{x};\theta), \tag{1}$$

where θ is the parameter of the model, and each token y^i in $\boldsymbol{y}$ is independent from other target tokens. The conditionally independent factorization demands the model to capture the modalities for the combination of all the tokens in one step, which can be difficult when the number of modalities is large. In contrast, the AR model factorizes the joint probability in an autoregressive way:

$$p(\boldsymbol{y}|\boldsymbol{x};\theta) = \prod_{i=1}^{n} p(y^i|y^{<i},\boldsymbol{x};\theta), \tag{2}$$

By conditioning on preceding tokens $y^{<i}$, the autoregressive factorization divides the modality learning of the sequence into multiple steps where each step learns to predict one token. Similar to the autoregressive factorization, the denoising diffusion models also smooth the source-target transformation by interpolating intermediate distributions between the source and the target.

Inspired by the learning decomposition in AR and diffusion models, we employ training procedures as in diffusion models to decompose the complex modality learning of NAR into several easier diffusion transition steps. Since the process of adding Gaussian noise is not designed for tackling the multi-modality problem, we also explore new diffusion processes to address the problem. A preliminary study demonstrates that a learning process with a gradually growing number of modalities is beneficial for learning modalities in NAR.

2.1 Denoising Diffusion Models

With a series of latent variables $\boldsymbol{y}_1\boldsymbol{y}_2...\boldsymbol{y}_T$, the diffusion process can be characterized by the posterior $q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0)$. This process guides the generative process $p(\boldsymbol{y}_{0:T}) := p_\theta(\boldsymbol{y}_T)\prod_{t=1}^{T} p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t)$ in diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) to fit the data $\boldsymbol{y}_0$ step by step. Depending on the Markov property of the process defined by $q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0)$, the diffusion processes can be divided into Markovian and non-Markovian ones.

Markovian Diffusion Process The Markovian diffusion processes are employed in many previous work for diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020). In these work, the forward process is a Markov chain where the posterior for $\boldsymbol{y}_t$ can be determined by conditioning on $\boldsymbol{y}_{t-1}$:

$$q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0) := \prod_{t=1}^{T} q(\boldsymbol{y}_t|\boldsymbol{y}_{t-1}) \tag{3}$$

For example, Ho et al. (2020) propose a denoising diffusion probabilistic model (DDPM), which adds Gaussian noise with increasing variances $\beta_{1:T} \in (0,1]^T$, and the forward transition probability is defined by: $q(\boldsymbol{y}_t|\boldsymbol{y}_{t-1}) := \mathcal{N}(\boldsymbol{y}_t; \sqrt{1-\beta_t}\boldsymbol{y}_{t-1}, \beta_t\mathbf{I})$. The posterior $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$ of DDPM can be computed in closed form, and the detailed derivation can be found in (Ho et al., 2020).

Source		Modality		Target
52 1 937 1234	I	$y^i = x^i + 5000$, i is odd;	$y^i = x^i + 10000$, i is even	5052 10001 5937 11234
	II	$y^i = x^i + 10000$, i is odd;	$y^i = x^i + 5000$, i is even	10052 5001 10937 6234
	III	$y^i = x^i + 15000$, i is odd;	$y^i = x^i + 20000$, i is even	15052 20001 15937 21234
	IV	$y^i = x^i + 20000$, i is odd;	$y^i = x^i + 15000$, i is even	20052 15001 20937 16234

Table 1: Illustration of the synthetic data.

Model	DATA	AR	NAR	NAR w/ PML Middle	NAR w/ PML Last
Token Acc.	100	100.0	56.6	98.8	97.7
Seq Acc.	100	100.0	0.0	93.4	88.6

Table 2: Results of synthetic experiments.

Non-Markovian Diffusion Process Besides the Markovian process, the posterior $q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0)$ can also be modeled as a non-Markovian process (Song et al., 2021a):

$$q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0) := q(\boldsymbol{y}_T|\boldsymbol{y}_0)\prod_{t=2}^{T} q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0), \quad (4)$$

where each decomposition term depends on $\boldsymbol{y}_0$. According to (Song et al., 2021a), non-Markovian processes are less stochastic than the Markovian process defined in Ho et al. (2020), leading to more deterministic and efficient generative models. Here, we propose to use the non-Markovian process to define the modality increasing process directly.

2.2 Proof of Concept

In addition, we also include a synthetic experiment to demonstrate that learning with growing number of modalities across layers can effectively benefit the performance of NAR. Specifically, we create a synthetic dataset with the 4 modalities shown in Table 1, where each source has only one target but may come from one of four different modalities. We simulate the modality diffusion process by using the progressive modality learning (PML) inside NAR layers. In PML, we train the middle layers of NAR with only 2 modalities by transforming the target of modality I to modality II and the target of modality of III to modality IV. And we train the last decoder layer by the targets with 4 modalities (See Appendix A for more details).

The results in Table 2 show that the vanilla NAR model fails to output correct sequences, and both the middle and last layer of the NAR trained with PML achieve significant accuracy gains. From the modality distribution plot in Figure. 2, we can find that the NAR trained with PML learns only 2 modalities in the middle layer, and captures all the modalities in the last layer.

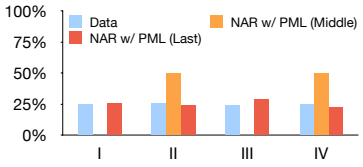

Figure 2: The output modality distributions.

3 The Proposed Diffusion-GLAT

In this section, we will introduce DIFFGLAT, which enables parallel sequence-to-sequence learning in a denoising diffusion implicit model (DDIM, Song et al., 2021a) framework. The primary goals of designing DIFFGLAT are: a) decomposing the modality learning to reduce the training difficulty of NAR and b) achieving high generation quality with few denoising transitions to keep fast decoding.

In denoising diffusion models, the parameters θ are trained to fit the data distribution $q(\boldsymbol{y}_0)$ by maximizing a variational lower bound (Ho et al., 2020):

$$\mathcal{L}_{\text{VLB}} = \mathbb{E}_{q(\boldsymbol{y}_0|\boldsymbol{x})}\Big[-\log p_\theta(\boldsymbol{y}_0||\boldsymbol{x})\Big] \leq \quad (5)$$
$$\mathbb{E}_{q(\boldsymbol{y}_{0:T}|\boldsymbol{x})}\Big[D_{\text{KL}}\big(q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0, \boldsymbol{x})||p_\theta(\boldsymbol{y}_{0:T}|\boldsymbol{x})\big)\Big]$$

To achieve the two goals described in the beginning, we define a discrete diffusion process q^{MDP} that adds the modalities in the data gradually as the diffusion step t decreases. Additionally, the training process is equipped with the residual glancing sampling techniques p_θ^{RGS}, which further boosts the modality learning ability. Therefore, we can rewrite the training objective in Eq. 5 as:

$$\begin{aligned}\mathcal{L}_{\text{VLB}} =& \mathbb{E}_q D_{\text{KL}}(q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0, \boldsymbol{x})|p_\theta(\boldsymbol{y}_{0:T}|\boldsymbol{x})) \quad (6)\\ =& \mathbb{E}_q D_{\text{KL}}(q^{\text{MDP}}(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0, \boldsymbol{x})||p_\theta^{\text{RGS}}(\boldsymbol{y}_{0:T}|\boldsymbol{x}))\end{aligned}$$

The modality diffusion process q^{MDP} gradually interpolates modalities among intermediate layers, which adaptively schedules sequence modalities across the NAR Transformer layers. And the residual glancing sampling (RGS) samples the target tokens that are not correctly captured across neural layers to enhance the modality learning process.

3.1 Modality Diffusion Process: q^{MDP}

In order to reduce the difficulty of learning modalities for NAR, we introduce a discrete diffusion process that distributes modalities to multiple transitions, which is illustrated in Figure 1. As shown in Section 2.2, a reasonable denoising process for learning modalities is adding the modalities in $\boldsymbol{y}_0$ gradually. But the modalities in the real world data

cannot be explicitly manipulated, thus we need to divide the modalities in $\boldsymbol{y}_0$ implicitly. Previous work reveals that the trained NAR models could capture part of the modalities in the data (Zhou et al., 2020). Therefore, we leverage the modality distribution captured by the model itself to construct the modality diffusion process.

Specifically, q^{MDP} should guarantee that the difficulty of the assigned modality learning task is appropriate for each transition. Thus, we define each transition of the modality diffusion process as the interpolation of distribution for target $\mathcal{P}(\boldsymbol{y}_0)$ and the prediction distribution of the next step $\mathcal{P}_{t-1}$. Mathematically, the posterior $q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0, \boldsymbol{x})$ can be decomposed as:

$$q^{\text{MDP}}(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0, \boldsymbol{x}) := q(\boldsymbol{y}_T|\boldsymbol{y}_0, \boldsymbol{x}) \prod_{t=2}^{T} q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0, \boldsymbol{x}) \tag{7}$$

For simplicity, we omit $\boldsymbol{x}$ for q in subsequent elaboration. With the model p_θ and the target $\boldsymbol{y}_0$, we interpolate $\boldsymbol{y}_0$ and $p_\theta(\boldsymbol{y}_{T-1}|\boldsymbol{y}_T, \boldsymbol{x})$ to construct intermediate distributions whose numbers of modalities are between the modalities in $\boldsymbol{y}_0$ and those captured by p_θ:

$$\begin{aligned} q(\boldsymbol{y}_t|\boldsymbol{y}_0) &:= \Big(\gamma_t \mathbf{1} + (1-\gamma_t)\boldsymbol{y}_0\Big) \odot \mathcal{P}_{t-1}/Z_t \\ &:= \Big(\gamma_t \mathcal{P}_{t-1} + (1-\gamma_t)\mathcal{P}_{t-1} \odot \boldsymbol{y}_0\Big)/Z_t, \end{aligned} \tag{8}$$

where $\mathcal{P}_{t-1} = p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x})$, $\boldsymbol{y}_0$ is the one-hot vector sequence of the target in data, $\gamma_t \in (0, 1]$ is the hyper-parameter for controlling the interpolation, $\odot$ represents element-wise multiplication and $Z_t \in (0, 1]^n$ is the factor for normalization.[1] We can consider the definition of Eq. 8 as data corruption on the modality probability landscape of $\mathcal{P}$ by extracting the $\mathcal{P}$ term out of the parenthesis. To ensure the form in Eq. 8 holds for $t \geq 1$ (See Appendix B), we define $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$ as:

$$\begin{aligned} q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0) := \gamma_{t-1}\mathcal{P}_{t-2} + \Big((1-\gamma_{t-1}) \\ \mathcal{P}_{t-2} - \omega_t \mathcal{P}_{t-1}\Big) \odot \boldsymbol{y}_0 + \omega_t Z_t \boldsymbol{y}_0 \odot \boldsymbol{y}_t \end{aligned} \tag{9}$$

[1]Because $\sum_{\boldsymbol{y}_t} q(\boldsymbol{y}_t|\boldsymbol{y}_0) = 1$ is not guaranteed after interpolation, we normalize the distribution with $Z_t = \sum_{w=1}^{|V|} \dot{q}(\boldsymbol{y}_t^w|\boldsymbol{y}_0)$, where $\dot{q}(\boldsymbol{y}_t|\boldsymbol{y}_0) = \gamma_t \mathcal{P}_{t-1} + (1-\gamma_t)\mathcal{P}_{t-1} \odot \boldsymbol{y}_0$, $|V|$ is the size of token categories and $\boldsymbol{y}_t^w$ is a sequence with the token w on all the positions.

Here, ω_t is a hyper-parameter. Similarly, we also re-normalize $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$. Intuitively, the sum of the first term with $\mathcal{P}_{t-2}$ and the second term with $\boldsymbol{y}_0$ performs the modality interpolation as in Eq. 8, and adding the third term with $\boldsymbol{y}_0 \odot \boldsymbol{y}_t$ attempts to preserve the modalities captured by $\boldsymbol{y}_t$ in the denoising transitions. Thus, $\boldsymbol{y}_{t-1}$ serves as a intermediate target, with fewer modalities than $\boldsymbol{y}_0$ but more modalities than $\boldsymbol{y}_t$, for smoothing the modality learning.

3.2 Non-autoregressive Generative Process

We define the generative process similar to the iterative refinement process in NAR. Specifically, we use a generative process that only conditions on the input computed by the model itself for every step, removing the gap between training and inference. Besides, we also augment the training with residual glancing sampling (RGS) to further smooth the modality learning, which improves over the glancing training (Qian et al., 2021) by layer-wise sampling and selecting tokens that are not captured in the input.

Self-Conditioned Generative Process To keep consistency between the training and inference, we parameterized $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t)$ completely by the model itself. We set the decoder input sequence as the initial state $\boldsymbol{y}_T$, and the embedding of decoder inputs $emb(\boldsymbol{y}_T)$ to be the initial representation $\boldsymbol{H}_T$. For the generative transitions $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t)$, we compute the next representation $\boldsymbol{H}_{t-1}$ with $\boldsymbol{H}_t$, and maps the hidden states representation to the softmax distribution for $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x})$:

$$\begin{aligned} &p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x}) = \text{softmax}(\boldsymbol{H}_{t-1}\boldsymbol{V}^T), \\ &\text{where} \quad \boldsymbol{H}_{t-1} = f_\theta^{(t)}(\boldsymbol{H}_t, \text{Enc}(\boldsymbol{x})), \end{aligned} \tag{10}$$

Here, Enc is the encoder that maps the input $\boldsymbol{x}$ to hidden states, $\boldsymbol{V}$ is the vocabulary embedding matrix, and $f_\theta^{(t)}$ is parameterized by neural layers. Since $\boldsymbol{H}_t$ contains the information for $\boldsymbol{y}_t$ and is consistent in both training and inference, we use $\boldsymbol{H}_t$ as the input for $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x})$. The self-conditioned generative process shares similar motivation with self-conditioning (Chen et al., 2022) and step-unroll (Savinov et al., 2022), but operates in a purer way as we directly transits the hidden states as in standard decoders.

Residual Glancing Sampling Previous work demonstrate that adaptive sampling target tokens according to the model performance can improve

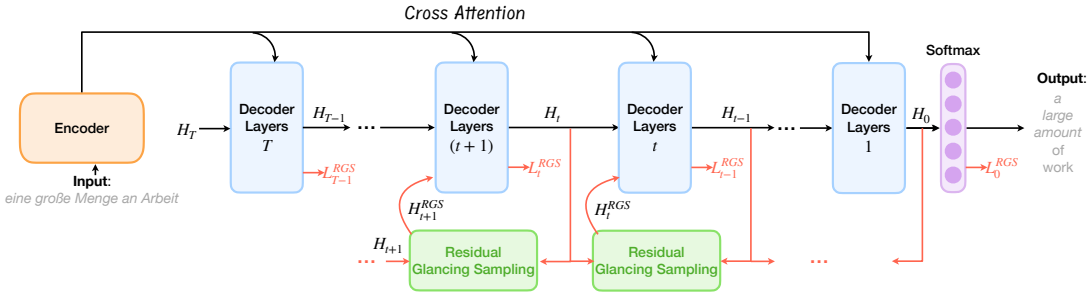

Figure 3: Learning procedure of the layer-wise residual glancing sampling.

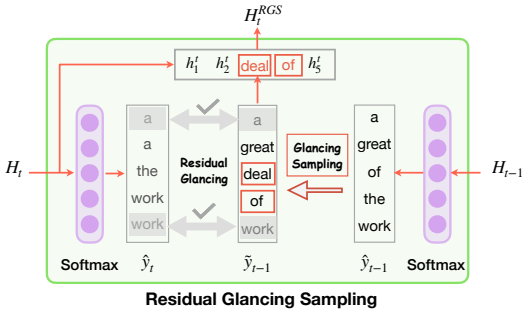

Figure 4: The details of residual glancing sampling.

the generation quality of NAR (Qian et al., 2021). For models training with diffusion processes, we strengthen the glancing training in a fine-grained way by layer-wise glancing and glancing remaining tokens. The overall residual glancing training procedure is shown in Figure 3.

For layer-wise glancing, we replace part of intermediate hidden states $\boldsymbol{H}_t$ with the embedding of tokens sampled from the intermediate target $\tilde{\boldsymbol{y}}_{t-1} = \arg\max q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$, and predict the remaining target tokens $\overline{\tilde{\boldsymbol{y}}^{\text{RGS}}_{t-1}}$ using the glancing target tokens $\tilde{\boldsymbol{y}}^{\text{RGS}}_{t-1}$:

$$\begin{aligned}\mathcal{L}^{\text{RGS}}_{t-1} &= -\log p^{\text{RGS}}_{\theta}(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x}) \\ &= -\log p_{\theta}(\overline{\tilde{\boldsymbol{y}}^{\text{RGS}}_{t-1}}|\tilde{\boldsymbol{y}}^{\text{RGS}}_{t-1}, \boldsymbol{y}_t, \boldsymbol{x})\end{aligned} \tag{11}$$

The sampling number of glancing tokens $\tilde{\boldsymbol{y}}^{RGS}_{t-1}$ is determined by the distance between the model output $\hat{\boldsymbol{y}}_{t-1} = \arg\max p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x})$ and the intermediate target $\tilde{\boldsymbol{y}}_{t-1}$: $S_{t-1} = \alpha \cdot d(\tilde{\boldsymbol{y}}_{t-1}, \hat{\boldsymbol{y}}_{t-1})$, where α is a hyper-parameter for adjusting the sampling number. Details for glancing could be found in Qian et al. (2021).

Besides layer-wise glancing, we propose to modify the scope for selecting glancing tokens as in Figure 4. Since the input captures part of the modalities, the model can learn only the remaining part that are not correctly captured in the input. Thus, we utilize $\hat{\boldsymbol{y}}_t$ to schedule the learning, and modify the glancing sampling to only sampling S_{t-1} tokens that are different between $\hat{\boldsymbol{y}}_t$ and $\tilde{\boldsymbol{y}}_{t-1}$:

$$\mathbb{RGS}_{t-1}(\tilde{\boldsymbol{y}}_{t-1}) = Random(\tilde{\boldsymbol{y}}_{t-1}/\hat{\boldsymbol{y}}_t) \tag{12}$$

In residual glancing training, we randomly sample $\tilde{\boldsymbol{y}}^{RGS}_{t-1}$ from $\tilde{\boldsymbol{y}}_{t-1}$ with the operation $\mathbb{RGS}_{t-1}$.

3.3 Implementation Details

In this section, we will introduce the implementaion details for DIFFGLAT. To enhance the ability of modality capturing, we employ the DA-Transformer (Huang et al., 2022b). The DA-Transformer (DAT) can strengthen each transition of DIFFGLAT by distributing the modalities to different positions in the expanded output sequences. We also elaborate the difference between the decoding iterations and the diffusion transitions in the parameterization part.

DA-Transformer In training, the DAT model expands the output lengths to the predefined max length, and maps target tokens to multiple positions in the expanded sequence. Since the output length of DAT is not equal to the length of the target $\boldsymbol{y}$, we use the best alignment of $\tilde{\boldsymbol{y}}_t$ as the glancing target and for computing q^{MDP}. The best alignment is obtained by maximizing the output probability:

$$\boldsymbol{y}^{align}_t = \arg\max_{\boldsymbol{a}\in\Gamma(\tilde{\boldsymbol{y}}_t)} p_t(\boldsymbol{a}|\boldsymbol{x}), \tag{13}$$

where Γ expands $\boldsymbol{y}$ to the expanded output length by inserting blanks. For the intermediate target $\tilde{\boldsymbol{y}}$ to compute loss L^{RGS}_{t-1}, we use the decoding result with the original target length $|\boldsymbol{y}|$ rather than the aligned target. In inference, after parallel neural network computation, DAT uses links between positions to extract output tokens in the expanded sequence, where we use the Joint-Viterbi decoding proposed by Shao et al. (2022).

Parameterization As each transition of the generative process directly forward the hidden states

Model		Iter	WMT14 En-De	WMT14 De-En	WMT17 En-Zh	WMT17 Zh-En	Average Gap	Speedup
AR	Transformer *base* (Ours)	M	27.18*	31.48*	34.65*	23.39*	0	1.0x
Iterative Models	Diff-LM (Li et al., 2022)	20	17.41	19.69	-	-	-10.78$^{\diamondsuit}$	0.6x
	CDCD (Dieleman et al., 2022)	100	20.0	26.0	-	-	-6.33$^{\diamondsuit}$	--
	Difformer (Gao et al., 2022)	20	23.80	-	-	-	-3.38$^{\diamondsuit}$	--
	DINOISER (Ye et al., 2023)	20	24.26	29.05	-	-	-2.68$^{\diamondsuit}$	--
	SUNDAE (Savinov et al., 2022)	10	25.99	30.24	-	-	-1.36$^{\diamondsuit}$	2.2x
	DAT+Iterative refinement†	3	25.67*	30.85*	-	-	-1.07$^{\diamondsuit}$	7.2x
Non-iterative Models	CTC (Libovický and Helcl, 2018)	1	17.73*	21.48*	25.77*	12.33*	-9.85	14.3x
	GLAT+CTC (Qian et al., 2021)	1	24.85*	28.37*	30.20*	17.57*	-3.92	14.3x
	DAT † (Huang et al., 2022b)	1	26.47*	30.22*	33.27*	23.21*	-0.88	13.0x
	DAT+DSLP †	1	26.08*	30.34*	-	-	-1.12$^{\diamondsuit}$	12.6x
Ours	DIFFGLAT †	1	26.72	31.35	34.12	23.74	-0.19	13.0x
	DIFFGLAT †	3	27.91	31.55	35.09	24.02	+0.47	7.2x

Table 3: Results on WMT14 En↔De and WMT17 Zh↔En. The average gap is computed against our Transformer implementation. * represents the results are obtained from our re-implementation, and $\diamondsuit$ indicates that the average gap is only computed with available results. For models with †, we use the Joint-Viterbi decoding proposed by Shao et al. (2022) for inference. The average gap is computed against the results of our implemented Transformer *base*.

to the next transition, we can build fully non-autoregressive models by stacking the neural layers or iterative models by sharing the parameters of θ_t. And we can use part of the layers in the decoder to perform one denoising transition so that one forward inference of the decoder can perform multiple denoising transitions.

Training Since $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$ is a distribution consists of multiple modalities, directly minimizing the KL-divergence in Eq. 6 introduces multi-modal targets for every source input. Thus, we optimize the model using the sequence with the highest probability: $\tilde{\boldsymbol{y}}_{t-1} = \arg\max q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$. The KL-divergence can then be rewritten as:

$$\mathcal{L}_{t-1}^{\text{RGS}} = -\log p_\theta^{\text{RGS}}(\tilde{\boldsymbol{y}}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x}) \qquad (14)$$

To reduce the denoising steps needed to achieve high quality, we use a small number for the total diffusion steps T and attempt to fit more modalities with each generative transition $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x})$. For training efficiency, we sample a diffusion timestep t to compute the loss for each training step.

4 Experiments

Different from most of the previous work for NAR, we directly train our models on raw data without using data distilled from AR. To verify the effectiveness of our method, we compare DIFFGLAT with several strong NAR and AR baselines on several sequence generation tasks. Furthermore, analysis and ablation studies study are also conducted to demonstrate the effects of each component.

4.1 Experimental Settings

Benchmarks We conduct experiments on machine translation, paraphrase generation and image captioning. For machine translation, we use 10 machine translation benchmarks: WMT14 En↔De, WMT17 En↔Zh, WMT14 En↔Fr, WMT16 En↔Ro and WMT13 En↔Es. The preprocessing follows the procedure in Zhou et al. (2020) and Kasai et al. (2020). For paraphrase generation and image captioning, we use the Quora Question Pairs dataset[2] (QQP), and MS-COCO dataset (Lin et al., 2014) respectively. The data are tokenized and segmented into subwords using Byte-Pair Encoding (Sennrich et al., 2016).

Evaluation Metrics We report the sacreBLEU[3] (Post, 2018) scores for machine translation, and the tokenized bleu results are shown in Appendix C. For speedups compared with the Transformer *base*, we follow previous work (Gu et al., 2018) by evaluating on WMT14 En-De with batch size 1. Further comparison for decoding speedup are provided in Appendix D.

Hyper-parameters We build our models based on the Transformer (Vaswani et al., 2017) architecture and use the Transformer *base* setting as default. For machine translation, we train all the models, including the autoregressive Transformer, with batches of 64k tokens and 300k training steps using the Adam optimizer (Kingma and Ba, 2015).

[2] https://www.kaggle.com/c/quora-question-pairs

[3] Except 'tok.zh' for En-Zh, the signature is "BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.5.1"

Models	WMT14		WMT17		WMT14		WMT16		WMT13	
	En-De	De-En	En-Zh	Zh-En	En-Fr	Fr-En	En-Ro	Ro-En	En-Es	Es-En
Transformer *base*	27.18	31.48	34.65	23.39	38.26	35.70	33.91	33.70	33.53	34.19
DIFFGLAT *base*	**27.91**	31.55	**35.09**	**24.02**	**38.73**	**36.61**	33.92	33.64	**33.95**	**34.74**
Transformer *big*	28.01	32.11	36.31	23.60	40.16	37.90	33.46	32.68	34.64	35.01
DIFFGLAT *big*	**28.62**	**32.32**	36.21	**24.40**	40.12	37.87	**34.34**	**33.65**	34.71	**35.40**

Table 4: BLEU scores on the 10 Machine Translation Benchmarks

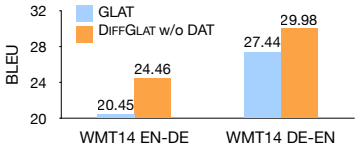

Figure 5: Results with GLAT

	QQP		MS-COCO 2014			
	BLEU4	ROUGE-L	BLEU4	METEOR	ROUGE-L	CIDEr
Transformer	28.13	58.19	34.0	28.1	56.0	112.3
DAT	28.82	59.76	33.5	27.0	56.5	106.5
DIFFGLAT	29.86	60.23	34.9	28.0	56.7	112.5

Table 5: Performance on paraphrasing and image captioning.

The training of DIFFGLAT with iterative decoding takes about 58 hours on 16 NVIDIA A100-80G GPUs. We average the best 5 checkpoints for BLEU scores on the validation set to get the final model. All the models are built with 6 Transformer decoder layers, and we use 3 decoder layers for each transition $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x})$. Thus, each decoding iterations performs 2 denoising diffusion steps. For paraphrasing and image captioning, we use smaller architecture and shorter training steps. The detailed hyper-parameters can be found in Appendix E.

4.2 Main Results

From the results in Table 3, we can find that DIFFGLAT achieves considerable improvement over strong baselines. With a process that gradually adds modalities in the denoising pass, DIFFGLAT reduces the number of modalities to learn for each transition, enhancing the ability for capturing modalities in data. Depending on whether the model parameters are reused iteratively, /method can be trained for non-iterative or iterative decoding. For the non-iterative setting, DIFFGLAT only modifies the training procedure, thus can keep the inference the same as the process of the base model. Compared with several strong baselines, we highlight the advantages of DIFFGLAT:

- Our method can achieve better generation quality than strong NAR or even AR. With only one iteration of parallel decoding, our model achives higher BLEU scores than that of previous non-autoregressive models. And when applying iterative decoding, our model can even outperform the Transformer with a margin of 0.47 BLEU on average. For comparison, directly applying iterative refinement to DAT does not improve the performance.
- DIFFGLAT overcomes the slow sampling of diffusion models and achieves high decoding efficiency. DIFFGLAT completely keeps the fast speed of parallel decoding for the non-iterative setting, and can still achieve a 7.2× speedup with 3 decoding iterations.
- We can also use DIFFGLAT without DAT or combine DIFFGLAT with CTC, where our approach achieves more improvements over the baselines. The results in Figure 5 show that our method improves 1.5∼4 BLEU over GLAT. The CTC results are shown in Appendix C.

Results on 10 Machine Translation Benchmarks To evaluate the performance more comprehensively, we compare the autoregressive Transformer with DIFFGLAT on 10 Benchmarks, and list the results in Table 4. We conduct experiments for both the *base* and the *big* setting. Our results show that DIFFGLAT outperforms or achieves comparable results to the Transformer on the 10 benchmarks.

Paraphrasing and Image Captioning Besides machine translation, we also conduct experiments for two other text generation tasks: paraphrase generation and image captioning. For paraphrase generation, DIFFGLAT can outperform the Transformer with about 1.7 BLEU scores and 2 ROUGE scores. As for image captioning, DIFFGLAT greatly improves over the DAT baseline and achieves slightly better results on the 4 metrics compared with the Transformer. Results on paraphrasing and image captioning shows that DIFFGLAT generalizes well for text generation tasks.

4.3 Ablation Study

Comparison of Different Diffusion Processes We substitute the modality process in Section 3.1

	WMT14 En-De	WMT14 De-En
DAT	25.96	30.02
+D3PM absorbing	26.84	30.70
+D3PM uniform	27.36	28.14
+Modality Diffusion	27.91	31.55

Table 6: Comparison of different diffusion processes. All the models use the residual glancing training.

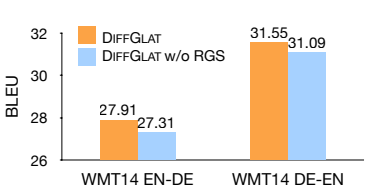

Figure 6: Ablation of the residual glancing strategy

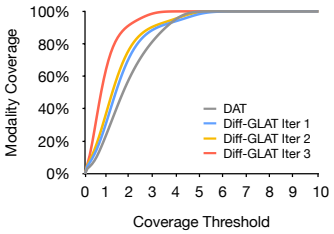

Figure 7: The modality coverage curves on WMT14 En-De.

to compare the performance of different diffusion processes. For comparison, we use two typical discrete diffusion processes proposed in (Austin et al., 2021): D3PM absorbing and D3PM uniform. In each forward step, the D3PM absorbing process masks each token of $\boldsymbol{y}_{t-1}$ with some given probabilities β_t, while the D3PM uniform process substitutes tokens in $\boldsymbol{y}_{t-1}$ with any other tokens uniformly. In our experiment, β_t is set to $1/(T-t+1)$. From the results in Table 6, we can find that our modality gains improvement over the absorbing and uniform process.

Effectiveness of Residual Glancing Training To verify the effectiveness of the residual glancing, we remove our proposed modification for the glancing training for comparison. The results are shown in Figure 6, note that the result of DIFFGLAT without RGS uses the original glancing training. We can find that removing residual glancing causes a performance decline, indicating that our residual glancing improves the original glancing training.

4.4 Analysis

Progressive Modality Capturing To measure the performance of modality capturing, we compute the modality coverage percentage under the corresponding thresholds, and the coverage curve is presented in Figure 7. Since the exact modalities for real world data is unavailable, we compute the coverage for the data point $(\boldsymbol{x}, \boldsymbol{y})$ as the modality is covered when the related data points are covered. Specifically, we compute the normalized loss for each data point with the trained model: $L_\theta(\boldsymbol{x}, \boldsymbol{y}) = -\log p_\theta(\boldsymbol{y}|\boldsymbol{x})/|\boldsymbol{y}|$. And the data point is considered covered by the model if $L_\theta(\boldsymbol{x}, \boldsymbol{y}) \leq \tau$, where $\tau \in \mathbb{R}$ is the threshold. For models with DAT, we compute $L_\theta(\boldsymbol{x}, \boldsymbol{y})$ with the best aligned path $\boldsymbol{a}^* = \arg\max_{\boldsymbol{a}\in\Gamma} P(\boldsymbol{y}, \boldsymbol{a}|\boldsymbol{x})$. As depicted in Figure 7, the value of τ ranges from 0 to 10, and each point represents the percentage of data points with $L_\theta(\boldsymbol{x}, \boldsymbol{y})$ less than or equal to τ.

In the figure, we can find that the curves of DIFFGLAT is overall on top of the DAT curve, indicating our model captures more modalities. And the three curves of DIFFGLAT shows that DIFFGLAT captures more modalities as the model performs more denoising steps.

Analysis for Decoding Iterations and Layers We also conduct experiments to analyze the performance with different decoding iterations and the effect of layer numbers for each denoising transition. The results are provided in Appendix F.

5 Related Work

The work for non-autoregressive neural machine translation can be divided into non-iterative and iterative methods. Previous non-iterative methods attempt to enhance the generation quality by earning from autoregressive models (Wei et al., 2019; Guo et al., 2020), incorporating light sequential decoding (Sun et al., 2019; Huang et al., 2022b), and introducing latent variables (Bao et al., 2019; Ma et al., 2019; Song et al., 2021b; Bao et al., 2022). Besides, models with iterative decoding have also been developed (Lee et al., 2018; Ghazvininejad et al., 2019; Gu et al., 2019; Huang et al., 2022a; Saharia et al., 2020; Savinov et al., 2022; Huang et al., 2022c).

DIFFGLAT also has connections with diffusion models for discrete data but differs from them in both probabilistic modelling and design motivation. More discussions can be found in Appendix I.

6 Conclusion

In this work, we propose DIFFGLAT, a parallel sequence generation model trained with the modality diffusion process and residual glancing sampling. By smoothing the learning of modalities in the diffusion model framework, DIFFGLAT greatly improves the generation quality of parallel generation. Compared with the autoregressive Transformer, DIFFGLAT achieves superior performance in both accuracy and efficiency for multiple se-

quence generation tasks, demonstrating the potential of the parallel generation paradigm.

Acknowledgements

This work is supported by the National Key R&D Program of China (2022ZD0160501), Natural Science Foundation of China (62376133). Hao Zhou is the corresponding author of this paper.

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993. Curran Associates, Inc.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Yu Bao, Hao Zhou, Jiangtao Feng, Mingxuan Wang, Shujian Huang, Jiajun Chen, and Lei Li. 2019. Non-autoregressive transformer by position learning. *CoRR*, abs/1911.10677.

Yu Bao, Hao Zhou, Shujian Huang, Dongqi Wang, Lihua Qian, Xinyu Dai, Jiajun Chen, and Lei Li. 2022. latent-GLAT: Glancing at latent variables for parallel text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8398–8409, Dublin, Ireland. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. 2022. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*.

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. 2021. Order-agnostic cross entropy for non-autoregressive machine translation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2849–2859. PMLR.

Zhujin Gao, Junliang Guo, Xu Tan, Yongxin Zhu, Fang Zhang, Jiang Bian, and Linli Xu. 2022. Difformer: Empowering diffusion model on embedding space for text generation. *arXiv preprint arXiv:2212.09412*.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020a. Aligned cross entropy for non-autoregressive machine translation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3515–3523. PMLR.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6111–6120. Association for Computational Linguistics.

Marjan Ghazvininejad, Omer Levy, and Luke Zettlemoyer. 2020b. Semi-autoregressive training improves mask-predict decoding. *CoRR*, abs/2001.08785.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 369–376. ACM.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*

2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 11179–11189.

Junliang Guo, Xu Tan, Linli Xu, Tao Qin, Enhong Chen, and Tie-Yan Liu. 2020. Fine-tuning by curriculum learning for non-autoregressive neural machine translation. In *Proceedings of the aaai conference on artificial intelligence*, volume 34, pages 7839–7846.

Zhengfu He, Tianxiang Sun, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. 2022. Diffusionbert: Improving generative masked language models with diffusion models. *arXiv preprint arXiv:2211.15029*.

Jindřich Helcl, Barry Haddow, and Alexandra Birch. 2022. Non-autoregressive machine translation: It's not as fast as it seems. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1780–1790, Seattle, United States. Association for Computational Linguistics.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*, volume 34, pages 12454–12465. Curran Associates, Inc.

Chenyang Huang, Hao Zhou, Osmar R. Zaïane, Lili Mou, and Lei Li. 2022a. Non-autoregressive translation with layer-wise prediction and deep supervision. *The Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022*.

Fei Huang, Hao Zhou, Yang Liu, Hang Li, and Minlie Huang. 2022b. Directed acyclic transformer for non-autoregressive machine translation. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9410–9428. PMLR.

Xiao Shi Huang, Felipe Perez, and Maksims Volkovs. 2022c. Improving non-autoregressive translation models without distillation. In *International Conference on Learning Representations*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 2020. Non-autoregressive machine translation with disentangled context transformer. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5144–5155. PMLR.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah A. Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1317–1327. The Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1173–1182. Association for Computational Linguistics.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *arXiv preprint arXiv:2205.14217*.

Jindrich Libovický and Jindrich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3016–3021. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard H. Hovy. 2019. Flowseq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4281–4291. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for

sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 48–53. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. 2021. Glancing transformer for non-autoregressive neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1993–2003. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1098–1108. Association for Computational Linguistics.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. 2022. Step-unrolled denoising autoencoders for text generation. In *International Conference on Learning Representations*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Chenze Shao, Zhengrui Ma, and Yang Feng. 2022. Viterbi decoding of directed acyclic transformer for non-autoregressive machine translation. In *Findings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022*.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France. PMLR.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021a. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

Jongyoon Song, Sungwon Kim, and Sungroh Yoon. 2021b. AligNART: Non-autoregressive neural machine translation by jointly learning to estimate alignment and translate. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1–14, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhi-Hong Deng. 2019. Fast structured decoding for sequence models. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3011–3020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. 2019. Imitation learning for non-autoregressive neural machine translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1304–1312. Association for Computational Linguistics.

Jiasheng Ye, Zaixiang Zheng, Yu Bao, Lihua Qian, and Mingxuan Wang. 2023. Dinoiser: Diffused conditional sequence learning by manipulating noises. *arXiv preprint arXiv:2302.10025*.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. 2023. A reparameterized discrete diffusion model for text generation. *arXiv preprint arXiv:2302.05737*.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A Settings of the Synthetic Experiment

Data In order to investigate the effect of modalities in learning, we create source-target pairs with known modalities for training. Suppose we have 4 modalities in the synthetic data, the source sequences can be transformed into target sequences with the rule of modality. We generate source sequences of length 32 with numbers uniformly chosen from $1 \sim 5000$, and randomly choose only one modality for each source sequence to create the target sequences. In total, we generate 100000 sequence pairs for training, and 5000 pairs each for validation and test.

Model Setup We train AR and NAR models on the synthetic data, with all models built with 4 encoder layers and 4 decoder layers. To learn all the modalities gradually, we train a NAR model with a modality growing process. Specifically, besides the original NAR training loss that learns 4 modalities, we also train the NAR model to learn 2 modalities with the middle decoder layer. To capture only 2 modalities in the middle, we merge modalities by transforming the targets of modality I to modality II and the targets of modality of III to modality IV. With the merged source-target pairs with only 2 modalities, we train the middle decoder layer to fit modality II and IV.

Evaluation For output quality measurement, we obtain the model outputs $\hat{\boldsymbol{y}}$, and compare them with the closest targets $\boldsymbol{y}$ in the 4 modalities for all the outputs. The number accuracy is the percentage of $\hat{y}_i = y_i$ and the sequence accuracy is the percentage of $\hat{\boldsymbol{y}} = \boldsymbol{y}$. To visualize the distribution of different modalities, we use the modality of the closest target as that of the output, and report the proportion for different modalities.

B Proof for Definition Consistency of the Modality Diffusion Process

Lemma 1. With $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$ defined in the modality diffusion process q^{MDP}, for $t \geq 1$, we have:

$$q(\boldsymbol{y}_t|\boldsymbol{y}_0) = (\gamma_t \mathcal{P}_{t-1} + (1-\gamma_t)\mathcal{P}_{t-1} \odot \boldsymbol{y}_0)/Z_t \quad (15)$$

Proof. According to the definition in Section 3.1, we have

$$q(\boldsymbol{y}_t|\boldsymbol{y}_0) := (\gamma_t \mathcal{P}_{t-1} + (1-\gamma_t)\mathcal{P}_{t-1} \odot \boldsymbol{y}_0)/Z_t \quad (16)$$

$$\begin{aligned} q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0) := \gamma_{t-1}\mathcal{P}_{t-2} + ((1-\gamma_{t-1})\mathcal{P}_{t-2} \\ - \omega_t \mathcal{P}_{t-1}) \odot \boldsymbol{y}_0 + \omega_t Z_t \boldsymbol{y}_0 \odot \boldsymbol{y}_t \end{aligned} \quad (17)$$

Since our models predcit the tokens in the sequence independently, the posterior $q(\boldsymbol{y}_t|\boldsymbol{y}_0)$ can also be decomposed into the form of independent token distributions:

$$\begin{aligned} q(\boldsymbol{y}_t|\boldsymbol{y}_0) &= \prod_{i=1}^{n} q(y_t^i|\boldsymbol{y}_0) \quad where \\ q(y_t^i|\boldsymbol{y}_0) &= \begin{cases} \gamma_t \mathcal{P}_{t-1}^i / Z_t & y_t^i \neq y_0^i \\ \mathcal{P}_{t-1}^i / Z_t & y_t^i = y_0^i \end{cases} \end{aligned} \quad (18)$$

Here, y_0^i is the ith token of $\boldsymbol{y}_0$, and $\mathcal{P}_{t-1}^i$ is the ith token output distribution of $\mathcal{P}_{t-1}$. In the same way, we can rewrite $q(y_{t-1}^i|\boldsymbol{y}_t, \boldsymbol{y}_0)$ as:

$$\begin{cases} \gamma_{t-1}\mathcal{P}_{t-2}^i & y_{t-1}^i \neq y_0^i \\ \mathcal{P}_{t-2}^i - \omega_t(\mathcal{P}_{t-1}^i - Z_t \mathbb{1}(y_0^i == y_t^i)) & y_{t-1}^i = y_0^i \end{cases} \quad (19)$$

For any $t \geq 2$, we can compute $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_0)$ by:

$$q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_0) = \int_{\boldsymbol{y}_t} q(\boldsymbol{y}_t|\boldsymbol{y}_0) q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0) \mathrm{d}\boldsymbol{y}_t \quad (20)$$

Thus,

$$\begin{aligned} & q(y_{t-1}^i = y_0^i|\boldsymbol{y}_0) \\ =& (1 - q(y_t^i = y_0^i|\boldsymbol{y}_0)) q(y_{t-1}^i|y_t^i \neq y_0^i, \boldsymbol{y}_0) \\ & + q(y_t^i = y_0^i|\boldsymbol{y}_0) q(y_{t-1}^i|y_t^i = y_0^i, \boldsymbol{y}_0) \\ =& (1 - \mathcal{P}_{t-1}^i/Z_t)(\mathcal{P}_{t-2}^i - \omega_t \mathcal{P}_{t-1}^i) \\ & + (\mathcal{P}_{t-1}^i/Z_t)(\mathcal{P}_{t-2}^i - \omega_t(\mathcal{P}_{t-1}^i - Z_t)) \\ =& \mathcal{P}_{t-2}^i - \omega_t \mathcal{P}_{t-1}^i + \mathcal{P}_{t-1}^i/Z_t \cdot \omega_t Z_t \\ =& \mathcal{P}_{t-2}^i \end{aligned} \quad (21)$$

And from Eq.19, we can derive:

$$q(y_{t-1}^i|\boldsymbol{y}_0) = \gamma_{t-1}\mathcal{P}_{t-2}^i \quad \text{for } y_{t-1}^i \neq y_0^i \quad (22)$$

Therefore, after normalization, we have:

$$\begin{aligned} q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_0) &= \prod_{i=1}^{n} q(y_{t-1}^i|\boldsymbol{y}_0) \quad where \\ q(y_{t-1}^i|\boldsymbol{y}_0) &= \begin{cases} \gamma_t \mathcal{P}_{t-2}^i / Z_{t-1} & y_{t-1}^i \neq y_0^i \\ \mathcal{P}_{t-2}^i / Z_{t-1} & y_{t-1}^i = y_0^i \end{cases} \end{aligned} \quad (23)$$

Similarly, we can prove that Eq.16 holds for $t \geq 1$.

C Additional Results on Machine Translation

For reference and further comparison, we also report the tokenized BLEU scores and the results based on CTC (Graves et al., 2006) rather DAT (Huang et al., 2022b).

Results with Tokenized BLEU As some previous work reports tokenized BLEU scores on the machine translation benchmarks, we also provide the results for tokenized BLEU scores for direct comparison.

Combination with CTC We also conduct experiments for DIFFGLAT with CTC, and the results are presented in Table 8.

The experimental results show that DIFFGLAT achieves improvements of 1∼3 BLEU scores over GLAT+CTC, demonstrating the effectiveness of DIFFGLAT. DIFFGLAT can easily combine with various existing methods for parallel generation because DIFFGLAT maintains the decoding process or simply adds more iterations. Specifically, DIFFGLAT keeps the original inference process in the non-iterative setting or forwards the decoder multiple times without intermediate decoding in the iterative setting.

D Inference Time Comparison

To provide a more comprehensive comparison of the decoding speedup, as discussed in (Helcl et al., 2022), we measure the inference latency on the WMT14 test set with 1 Nvidia-V100 GPU, and report the inference latency with batch size 1 in Table 9. Following the setting in Kasai et al. (2021), we also measure the inference latency of a Transformer with 12 encoder layers and 1 decoder layer.

Comparing with the Transformer (12-1), DIFFGLAT with 3 decoding iterations still has a 2.6× speedup. Although the Transformer with deep encoder and shallow decoder can achieve faster inference, the depth of decoder is important for capability of decoder-only models (Brown et al., 2020; OpenAI, 2023). Thus, the comparison for models with 6 decoder layers is also useful.

E Hyper-parameters for Experiments

We train our models with fairseq (Ott et al., 2019)[4]. For machine translation, the dropout is set to 0.1 except En-Ro/Ro-En and the Transformer *big* setting, where the dropout is 0.3. For paraphrasing and image captioning, we use the dropout of 0.3. Since our modality diffusion process requires the model to capture part of the modality, for the first 100k steps, we train the model to predict the target $\boldsymbol{y}_0$ at all steps t. For the subsequent training transitions, we train the model with the modality diffusion process. For DAT, the decoding length is set to be 8 times the source length for non-iterative models and 4 times for iterative models.

For the hyper-parameters of the modality diffusion in Eq. 9, we use a simplified implementation with the interpolation between $\mathcal{P}_{t-2}$ and $\boldsymbol{y}_0$:

$$q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0) = \gamma_{t-1}\mathcal{P}_{t-2} + (1-\gamma_{t-1})\mathcal{P}_{t-2} \odot \boldsymbol{y}_0 \quad (24)$$

Here, we set $\gamma_{t-1} \in (0, 1]$ to be selected from a predefined set of values. To achieve a similar effect for interpolating with $\boldsymbol{y}_0 \odot \boldsymbol{y}_t$, we choose γ_{t-1} as the maximum value that preserves 90% of the tokens in $\boldsymbol{y}_0 \odot \boldsymbol{y}_t$. With such interpolation, $\boldsymbol{y}_{t-1}$ includes most of the target tokens that is correctly predicted in the previous $\boldsymbol{y}_t$. Thus, the difference between $\boldsymbol{y}_t$ and $\boldsymbol{y}_0$ gradually reduces as t decreases.

In terms of residual glancing training, we set the hyper-parameter of glancing schedule $\mu = 1/T$. And the hyper-parameter for computing sampling numbers α decrease from 0.5 to 0.1 periodically for iterative DIFFGLAT.

For paraphrase generation, we utilize a Transformer architecture with 4 encoder layers and 4 decoder layers, with a hidden dimension of 256. For image captioning, we use the image features extracted by fast R-CNN [5] with a Transformer *base* setting. The maximum training steps are set to 100k for both paraphrase generation and image captioning.

F Additional Analysis

To further study our approach, we analyze the impact of the number of decoding iterations, the number of layers for one reverse transition, and the total diffusion steps in training.

Performance with Iterative Decoding The BLEU scores for different decoding iterations are presented in Figure 8. Here, the evaluated models are trained with generative processes of 3 iterations. Our findings indicate that the BLEU scores increases as the number of iterations increases to 3, and the scores do not increase when the iterations exceeds 3. Note that the decoding iterations in inference can be different from that in training. Since the model is trained using a generation process of 3 iterations, it also achieves the best scores

[4] https://github.com/facebookresearch/fairseq

[5] https://github.com/peteanderson80/bottom-up-attention

Model		Iter	WMT14 En-De	WMT14 De-En	WMT17 En-Zh	WMT17 Zh-En	Average Gap	Speedup
AR	Transformer (Vaswani et al., 2017)	M	27.6	31.4	34.3	23.7	-0.35	1.0x
	Transformer *base* (Ours)	M	27.81*	31.96*	34.65*	23.98*	0	1.0x
Iterative Models	CMLM (Ghazvininejad et al., 2019)	10	24.61	29.40	-	-	-2.88$^\diamondsuit$	2.2x
	Imputer (Saharia et al., 2020)	8	25.0	-	-	-	-2.96$^\diamondsuit$	2.7x
	SUNDAE (Savinov et al., 2022)	10	26.25	30.80	-	-	-1.36$^\diamondsuit$	2.2x
	CMLMC (Huang et al., 2022c)	10	26.40	30.92	-	-	-1.23$^\diamondsuit$	1.7x
	DINOISER (Ye et al., 2023)	20	24.48	29.40	-	-	-2.68$^\diamondsuit$	--
Non-iterative Models	CTC (Libovický and Helcl, 2018)	1	18.42	23.65	26.84	12.23	-9.31	14.3x
	OaXE (Du et al., 2021)	1	22.4	26.8	-	-	-5.28$^\diamondsuit$	14.2x
	GLAT+CTC (Qian et al., 2021)	1	25.02	29.14	30.65	19.92	-3.42	14.3x
	DAT† (Huang et al., 2022b)	1	26.95*	30.73*	33.27*	23.60*	-0.96	13.0x
Ours	DIFFGLAT †	1	27.40	31.94	34.12	24.23	-0.18	13.0x
	DIFFGLAT †	3	28.57	32.08	35.09	24.86	+0.55	7.2x

Table 7: The tokenized BLEU scores on WMT14 En↔De and WMT17 Zh↔En. The average gap is computed against our Transformer implementation. * represents the results are obtained from our re-implementation, and $\diamondsuit$ indicates that the average gap is only computed with available results. For models with †, we use the Joint-Viterbi decoding proposed by Shao et al. (2022) for inference. The average gap is computed against the results of our implemented Transformer *base*.

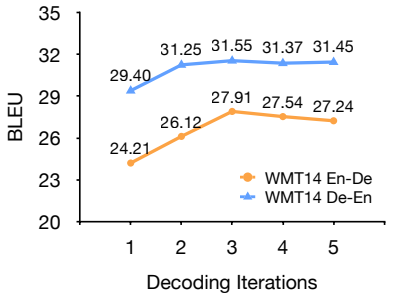

Figure 8: The BLEU score curves with iterative decoding.

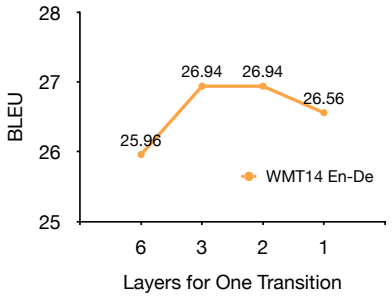

Figure 9: Comparison for the number of decoder layers in one denoising diffusion transition.

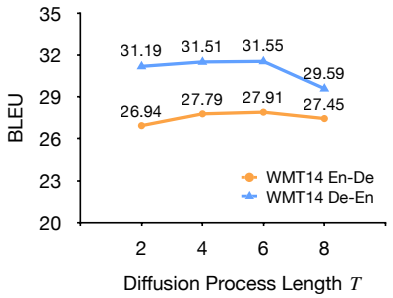

Figure 10: The effect of different diffusion process lengths.

	Iter	WMT14 En-De	WMT14 De-En	WMT17 En-Zh	WMT17 Zh-En
GLAT+CTC	1	24.85	28.37	30.20	17.57
DIFFGLAT (CTC)	1	**25.92**	**29.98**	**31.77**	**20.66**

Table 8: The results of DIFFGLAT based on CTC

	Transformer 6-6	Transformer 12-1	DIFFGLAT 6-6 (1 iteration)	DIFFGLAT 6-6 (3 iterations)
Latency	297.0ms	108.9ms	22.8ms	42.1ms

Table 9: The comparison of inference latency. A-B represents the model has A encoder layers and B decoder layers

with 3 decoding iterations, which is consistent with training.

The Number of Decoder Layers for each Transition We use 3 decoder layers to model one transition $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{x})$. Thus, one decoding iteration of the 6-layers decoder performs 2 denoising transitions. Without changing the total number of the decoder layers and iterative decoding, we conduct experiments to study the influence of the number of layers for one transition. The results in Figure. 9 show that the model achieves the best performance when using 2 or 3 layers for one transition. But the scores with only 1 layer for 1 transition decreases, which may caused by insufficient modelling capacity with only 1 layer.

The Effect of Diffusion Process Lengths We also investigate the effect of diffusion process steps T and the results are illustrated in Figure 10. Note that we perform each reverse denoising step with 3 decoder layers, so every iteration corresponds to 2 steps in the diffusion process. We find that the performance grows on WMT14 datasets until T reaches 6, and declines when the T is 8. We think the reason why the performance stops growing with the increasing iterations is that large iterations makes part of the denoising transitions too easy to learn. The easy learning task can lead to capability degradation, as the issues caused by adding

small noises in diffusion models for text (Li et al., 2022).

G The chrF and COMET Scores

	WMT14 En-De	WMT14 De-En	WMT17 En-Zh	WMT17 Zh-En
Transformer	0.5800	0.5832	0.3078	0.5230
DAT	0.5621	0.5669	0.2995	0.5090
DIFFGLAT	0.5790	0.5815	0.3136	0.5310

Table 10: The chrF scores on WMT14 En↔De and WMT17 En↔Zh

	WMT14 En-De	WMT14 De-En	WMT17 En-Zh	WMT17 Zh-En
Transformer	0.8623	0.8711	0.8599	0.8537
DAT	0.7966	0.8470	0.8265	0.8222
DIFFGLAT	0.8341	0.8600	0.8469	0.8366

Table 11: The COMET scores on WMT14 En↔De and WMT17 En↔Zh

Besides the commonly used BLEU (Papineni et al., 2002) metric, we compute the additional chrF (Popović, 2015) and COMET (Rei et al., 2020) scores for more comprehensive evaluation. We use the wmt22-comet-da for computing the COMET score.

DIFFGLAT achieves better chrF scores on WMT17 En↔Zh and comparable scores on WMT14 En↔De compared to the Transformer. And DIFFGLAT still outperforms DAT in terms of COMET but falls behind the Transformer baseline. The gap in COMET may caused by the distribution mismatch between the NAR outputs and the training data for COMET.

H Data Statistics and Evalution Metrics

The statistics of the data we used in experiments are listed as follow: WMT14 En↔De (4.5m), WMT17 En↔Zh (20m), WMT14 En↔fr (35m), WMT16 En↔Ro (0.6m) and WMT13 En↔Es(12m). For paraphrase generation and image caption, we use the Quora (145k) dataset and MS-COCO (113k) dataset respectively. For MS-COCO, we use the Karpathy split (Karpathy and Fei-Fei, 2015).

For paraphrase, we report tokenized BLEU and ROUGE-L (Lin, 2004)[6]. For image caption, we also report METEOR (Banerjee and Lavie, 2005) and CIDEr (Vedantam et al., 2015), and use the official evaluation tools for evaluation[7].

[6]The script is ROUGE-1.5.5.pl

[7]https://github.com/cocodataset/cocoapi

I Connections with Diffusion Models and Iterative NAR

Diffusion Models for Discrete Data Previous work studies continuous or discrete process for modelling discrete data with diffusion models. For continuous diffusion processes, a series of work explores adding Gaussian noise in the word embedding space (Li et al., 2022), converting the discrete data to 0/1 bits (Chen et al., 2022), the design space of diffusion models(Dieleman et al., 2022), and conditional text generation with continuous diffusion (Gong et al., 2022). For discrete diffusion processes, Hoogeboom et al. (2021) study a multinomial diffusion process where each state transits to other states uniformly, while Austin et al. (2021) explore more types of state transitions, including masking and increasing the transition frequency for similar states. He et al. (2022) propose a noise schedule based on token information. Zheng et al. (2023) derive a discrete diffusion framework with route mechanism via reparameterization.

Discussion for comparison with Step-Unrolled Models and Semi-Autoregressive Training Savinov et al. (2022) introduces a denoising procedure related to diffusion models but different from ours. Specifically, SUNDAE corrupts the target and uses the corrupted sequence as the input for learning denoising. In contrast, DIFFGLAT employs a diffusion process for dividing modalities, and uses the corrupted sequence as the intermediate training target. Besides, SUNDAE heavily relies on multiple decoding iterations, while our method can achieve competitive quality without iterative decoding.

Compared with SMART (Ghazvininejad et al., 2020b), our method learns to predict the intermediate targets rather than replacing decoder inputs. For SMART, the model first generates outputs with several iterations and uses the outputs as the decoding inputs to learn mistake correction. In contrast, DIFFGLAT uses the intermediate targets to capture the multi-modality in data gradually. Although both methods can be used for iterative refinement, they work on input and target, respectively

J Limitations

This work does not include large-scale pre-training experiments, which cannot be directly compared with the large autoregressive GPTs. The aim of this work is to establish the foundational framework

and potential of non-autoregressive text generation models, leaving the more detailed and expansive comparative study as a future endeavor.